افزایش سرعت نگهداری افزایشی دید با استفاده از الگوریتم فاخته

عفیفه کریمی مصدق[1] و نگین دانشپور[2]*

[1] دانشکده مهندسی کامپیوتر و فناوری اطلاعات، دانشگاه آزاد اسلامی‌قزوین، قزوین، ایران

[2] دانشکده مهندسی کامپیوتر، دانشگاه تربیت دبیر شهید رجایی، تهران، ایران

چکیده

پایگاه داده تحلیلی مخزنی از اطلاعات یکپارچه شده است که از منابع مختلف جمع‌آوری می‌شود. در پایگاه‌داده‌تحلیلی داده‌های استخراج‌شده از منابع مختلف، به فرم دید ذخیره می‌شوند؛ بنابراین دیدها باید نگهداری شوند و در هنگام تغییر منابع داده، دیدها نیز به‌روز شوند. از آن‌جایی که افزایش به‌روزرسانی‌ها ممکن است سربار و هزینه زیادی داشته باشد، ضروری است که به‌روزرسانی دیدها با دقت بالایی صورت گیرد. الگوریتمی که در این مقاله ارائه می‌شود، ترکیب یک روش گروه‌بندی، با الگوریتم فراابتکاری فاخته است که باعث کاهش زمان نگهداری دید و درنتیجه افزایش سرعت نگهداری دید افزایشی می‌شود. الگوریتم بهینه‌سازی فاخته با یک جمعیت اولیه آغاز می‌شود. تلاش برای زنده‌ماندن این فاخته‌ها اساس الگوریتم بهینه‌سازی است. نتایج پیاده‌سازی نشان می‌دهد که الگوریتم فاخته در مقایسه با روش‌های قبلی از سرعت بالاتری به‌منظور به‌روزرسانی دید افزایشی برخوردار است.

واژگان کلیدی: پایگاه‌داده‌تحلیلی، الگوریتم فاخته، جستجوی تصادفی، درخت دلتای بهینه، نگهداری افزایشی دید.

Increasing the Speed of Incremental View Maintenance Using the Cuckoo Algorithm

Afifeh Karimi Mosadegh[1] & Negin Daneshpour[2*]

[1]Faculty of Computer and Information Technology Engineering, Qazvin Branch, Islamic Azad University

[2]Faculty of Computer Engineering, Shahid Rajaee Teacher Training University, Tehran, Iran

Abstract

Data warehouse is a repository of integrated data that is collected from various sources. Data warehouse has a capability of maintaining data from various sources in its view form. So, the view should be maintained and updated during changes of sources. Since the increase in updates may cause costly overhead, it is necessary to update views with high accuracy. Optimal Delta Evaluation method is one of the incremental view maintenance method that can maintain materialized views efficiently in the data warehouse environment. This method is one of the incremental view maintenance grouping methods. In this method incremental maintenance expression is divided into groups, as a result access to some repeated relations is minimized. As a final result, Optimal Delta Evaluation method can minimize the total accesses to relations. The algorithm proposed in this paper, is the combination of optimal Delta Evaluation with Cuckoo heuristic Algorithm that reduces maintenance time of views and thus speeds up this process. Cuckoo optimization algorithm begins with an initial population. Trying to survive the Cuckoo makes the base to optimize the algorithm. The results show that the Cuckoo algorithm is faster in order to update its incremental views compared with previous methods.

Key words: data warehouse, Cuckoo algorithm, random search, optimization delta tree, incremental view maintenance.

* نویسندهٔ عهده‌دار مکاتبات

* Corresponding author

۱- مقدمه

امروزه با افزایش حجم داده‌ها، مسئله اصلی سازمان‌ها پردازش و نگهداری آن‌ها است. تسریع پردازش داده‌ها و نگهداری به‌موقع آن‌ها به تصمیم‌گیران و مدیران سازمان کمک می‌کند. اگرچه سازمان‌ها داده‌های زیادی در اختیار دارند؛ ولی داده‌های مناسب جهت اتخاذ تصمیماتی در حوزه طرح‌ریزی سیاست‌های آتی و طرح‌ریزی آینده در اختیار ندارند. راه‌حل این مشکل، پایگاه‌داده‌تحلیلی[1] است. پایگاه‌داده تحلیلی، داده‌های مورد نیاز را از منابع مختلف جمع‌آوری و یکپارچه کرده و آن را به شکل دید در خود ذخیره می‌کند. نگهداری پرس‌وجوهای از پیش محاسبه‌شده در قالب دیدهای ذخیره‌شده می‌تواند پاسخ به پرس‌وجوها را تسریع کند [1]. نظر به این‌که این دیدها جهت پاسخ‌گویی به پرس‌وجوهای تحلیلی مدیران استفاده می‌شوند، به‌روز بودن آن‌ها بسیار ضروری است و درصورت عدم به‌روزرسانی، پاسخ‌ها غیر معتبر بوده و منجر به تحلیل و تصمیم‌گیری نادرست خواهد شد. بنابراین به‌محض تغییر داده‌های پایه، دیدهای ذخیره‌شده نیز باید به‌روز شوند. به‌روزرسانی دید می‌تواند به‌صورت محاسبه مجدد رخ دهد؛ ولی محاسبه مجدد هزینه‌بر و نامطلوب است. برای این‌که به‌روزرسانی دیدها هزینه کمتری داشته باشد، ترجیح بر این است که عملیات بازسازی دید، محدود به محاسبه مجدد آن بخشی از پرس‌وجوی تعریف‌کنندهٔ دید باشد، که داده‌های مرتبط با آن بخش به‌روزرسانی شده‌اند. این روش بازسازی به نگهداری افزایشی دید[2](IVM) موسوم است[7]-[2].

روش‌های قدیمی نگهداری افزایشی دید زمان زیادی را صرف به‌روزکردن دیدها می‌کنند و این امر باعث کاهش سرعت نگهداری دید می‌شود. ازآن‌جایی که اگر به‌روزشدن دیدها به‌موقع انجام نشود، باعث می‌شود نتیجه یک پرس‌وجو اشتباه به‌دست آید، به‌روزکردن دیدها با سرعت بالا و به‌موقع ضرورت زیادی دارد. هدف از این مقاله، ارائه الگوریتمی است که سرعت نگهداری دید را افزایش دهد. این الگوریتم بر پایه محاسبات هوش مصنوعی است. در این مقاله از الگوریتم فراابتکاری فاخته[3]استفاده شده و زمان یافتن یک رابطهٔ بهینه برای نگهداری افزایشی دید کاهش یافته است. ساختار مقاله بدین شرح است: بخش دو، به بیان کارهای گذشته اختصاص داده شده است. در بخش سه، ابتدا عبارت نگهداری دید معرفی و سپس یک روش نگهداری دید افزایشی معرفی می‌شود که روش دلتای بهینه نام دارد. از معایب این روش این است که زمان زیادی را صرف پیدا کردن رابطه بهینه برای نگهداری دید افزایشی می‌کند. درنهایت در بخش چهارم روش پیشنهادی این مقاله مطرح می‌شود. در بخش پنجم نتایج شبیه‌سازی برای مقایسه روش‌ها با یکدیگر آورده شده است.

۲- معرفی الگوریتم‌های گذشته

این بخش دارای دو زیربخش است. بخش نخست معرفی کارهای گذشته در زمینه نگهداری افزایشی دید و بخش دوم معرفی الگوریتم های فراابتکاری است.

۲-۱- کارهای گذشته در زمینه نگهداری افزایشی دید

برای نگهداری افزایشی دید ایده‌ها و الگوریتم‌های مختلفی وجود دارد [23]-[8]، [7]، [2] که هدف آن‌ها کاهش هزینه یعنی کاهش محاسبات و افزایش سرعت نگهداری دید است.

یکی از روش‌ها، روش کمینه‌کردن اندازه افزایش است[24]. در این روش برای دیدی که از n رابطه تشکیل شده است، ابتدا اندازه تغییرات هر رابطه را حساب کرده سپس آن‌ها را به‌صورت صعودی در صفی قرار داده و آن‌ها را به‌ترتیب به‌روز می‌کند. این روش باعث می‌شود که افزایش داده روی رابطه‌های بعدی کمتر شود و درنتیجه هزینه نگهداری دید کاهش یابد و کارایی پایگاه‌داده بالا رود.

روش دیگر نگهداری دیدها به‌صورت موازی است [25]. این روش، یک روش نگهداری دید فوری است که از یک چارچوب چندعامله استفاده می‌کند. در سامانه‌های چندعامله، MAS[4] مانند خطی است که taskها را به‌صورت منظم از خود انتقال داده و عامل‌های مناسب به‌طور همزمان روی taskها عمل می‌کنند. استفاده از عامل‌ها باعث می‌شود پردازش به‌روزرسانی‌ها به‌صورت موازی انجام شود و این باعث می‌شود که سرعت به‌روزرسانی‌ها افزایش یابد.

از دیگر روش‌های نگهداری دید، روش‌های مبتنی بر جبران هستند[29]-[26]. تمامی روش‌های مبتنی بر جبران با ارسال پرس‌وجوهای جبران خطاهایی را که ممکن است در به‌روزرسانی همزمان دیدها رخ دهد از بین می‌برند. ECA

[1] Data Warehouse
[2] Incremental view maintenance
[3] Cuckoo
[4] Multi agent system

[26] یکی از الگوریتم‌های مبتنی بر جبران است. این الگوریتم به‌ازای تمامی پرس‌وجوهای پاسخ داده نشده که برای پایگاه‌داده تحلیلی وجود دارد یک پرس‌وجوی جبران ارسال می‌کند. از معایب این الگوریتم این است که تنها برای یک منبع داده[3] عملکرد مناسبی دارد. یکی دیگر از الگوریتم‌های مبتنی بر جبران الگوریتم Strobe [27] نام دارد. این الگوریتم مشکل روش ECA را برطرف می‌کند. Strobe به‌روزرسانی‌ها را به همان ترتیبی که می‌رسند پردازش می‌کند. از معایب این الگوریتم این است که تا زمانی که به‌روزرسانی‌های سامانه تمام‌نشده باشد، دید ذخیره‌شده به‌روز نمی‌شود. درواقع این الگوریتم منتظر می‌ماند تا همه به‌روزرسانی‌ها دریافت شوند و فقط آن زمان است که نتایج تمام به‌روزرسانی‌ها را در دید ادغام می‌کند. پس ممکن است اگر به‌روزرسانی‌ها ادامه داشته باشند دید ذخیره‌شده به‌روز نشود.

الگوریتمSWEEP یکی دیگر از روش‌های مبتنی بر جبران است [28]. این روش پرس‌وجوهای جبران را برای هر رابطه منبع می‌فرستد حتی اگر چند منبع در پایگاه‌داده یکسان موجود باشند. این امر باعث افزایش تعداد پرس‌وجوی جبران به منابع و افزایش هزینه نهایی آن می‌شود. روش DSCM [29] یکی دیگر از روش‌های مبتنی بر جبران است. این روش از همکاری Online پایگاه‌داده تحلیلی و منابع داده استفاده می‌کند و باعث کاهش رفت‌وبرگشت بین پایگاه‌داده تحلیلی و منابع داده می‌شود و اندازه پرس‌وجوهای جبران را کاهش می‌دهد و از این طریق سربار نگهداری دید افزایشی را کاهش می‌دهد. روش دیگری که برای نگهداری دید افزایشی مطرح شده استفاده از روش‌های گروه‌بندی است[32]-[30]. این روش‌ها با دسته‌بندی‌کردن عبارت نگهداری افزایشی دید، دفعات دسترسی به داده‌ها را کاهش داده و هزینه نگهداری دید را کاهش می‌دهند.

یکی از روش‌های گروه‌بندی روش دلتای بهینه است[30]. این روش برای یافتن عبارت نگهداری دیدی که کمترین هزینه را دارد از درخت ارزیابی دلتا استفاده می‌کند [30]، [33]. این روش برای یافتن عبارتی با کمترین هزینه، زمان زیادی را صرف می‌کند. این روش با جزییات بیشتر در بخش چهار ارائه می‌شود. از آن‌جایی که اگر زمان به‌روزرسانی دیدها کاهش یابد، باعث می‌شود که دیدهای بیشتری در زمان مشخص‌شده به‌روز شوند، در این مقاله از الگوریتم‌های فراابتکاری که برای بهینه‌سازی مورد استفاده قرار می‌گیرند، استفاده شده است. الگوریتم فراابتکاری مورد استفاده در این مقاله، الگوریتم فاخته است، که منجر به بهبود سرعت نگهداری دید افزایشی می‌شود. در جدول (۱) برخی از جدیدترین روش‌های نگهداری افزایشی دید که مرتبط با کار انجام‌شده در این مقاله می‌باشد، آورده شده است.

(جدول -۱): برخی از جدیدترین روش‌های نگهداری افزایشی دید
(table-1):Some of the newest methods for incremental view maintenance

الگوریتم	معایب	هزینه	سرعت
ECA	Singel Sourceاست	افزایش هزینه	افزایش سرعت
Strobe	برای به‌روزرسانی دید نیاز به وضعیت سکون سیستم دارد و ممکن است دید اصلا بروز نشود	افزایش هزینه	کاهش سرعت
SWEEP	پرس‌وجوهای نگهداری را به تمام روابط موجود می‌فرستد حتی اگر روابط در یک منبع باشند و سرعت را پایین می‌آورد.	افزایش هزینه	کاهش سرعت
DSCM	در به‌روزرسانی یک دید کاربرد دارد	افزایش هزینه	افزایش سرعت
روش چند عامله	این روش برای جبران به‌روزرسانیهای همزمان از compensation الگوریتم های قدیمی مثل sweep استفاده می‌کند.	افزایش هزینه	افزایش سرعت
روش دلتای بهینه	عبارت ارزیابی را به گروه‌هایی تقسیم کرده و تقسیم‌بندی را انتخاب کرده که کمترین دسترسی به داده را داشته باشد	کاهش هزینه	کاهش سرعت

۲-۲- معرفی الگوریتم‌های فراابتکاری

روش‌ها و الگوریتم‌های بهینه‌سازی به دو دسته الگوریتم‌های دقیق و الگوریتم‌های تقریبی تقسیم‌بندی می‌شوند. الگوریتم‌های دقیق قادر به یافتن جواب بهینه به‌صورت دقیق هستند؛ اما در مورد مسائل بهینه‌سازی سخت کارایی ندارند.

الگوریتم‌های تقریبی قادر به یافتن جواب‌های خوب (نزدیک به بهینه) در زمان حل کوتاه برای مسائل بهینه‌سازی سخت هستند. الگوریتم‌های تقریبی نیز به سه دسته الگوریتم‌های ابتکاری و فراابتکاری و فوق ابتکاری بخش‌بندی می‌شوند. دو مشکل اصلی الگوریتم‌های ابتکاری، قرارگرفتن آن‌ها در بهینه‌های محلی، و ناتوانی آن‌ها برای کاربرد در مسائل گوناگون است. الگوریتم‌های فراابتکاری برای حل این مشکلات الگوریتم‌های ابتکاری ارائه شده‌اند. درواقع الگوریتم‌های فراابتکاری، یکی از انواع الگوریتم‌های بهینه‌سازی تقریبی هستند که دارای راه‌کارهای برونرفت از بهینه محلی می‌باشند و قابل کاربرد در طیف گسترده‌ای از مسائل هستند.

۱-۲-۲- الگوریتم فاخته

الگوریتم بهینه‌سازی فاخته COA [7]، [35]-[34] یکی از جدیدترین و قوی‌ترین روش‌های بهینه‌سازی تکاملی می‌باشد که تاکنون معرفی شده است.

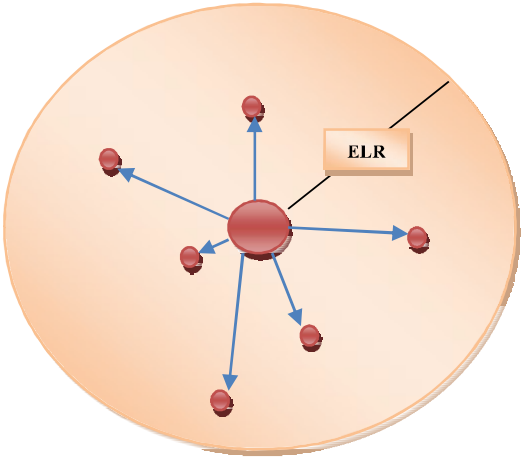
این الگوریتم از روش زندگی پرنده‌ای به نام فاخته الهام گفته است. مشابه دیگر روش‌های تکاملی، الگوریتم تکاملی فاخته نیز با یک جمعیت اولیه کار خود را شروع می‌کند که این جمعیت متشکل از فاخته‌ها و تخم‌های فاخته است.

جمعیت فاخته‌ها شروع به تخم‌گذاری در لانه پرندگان دیگر خواهند کرد. تخم‌هایی که شباهت بیشتری به تخم‌های پرنده میزبان دارند زنده مانده و فاخته بالغ خواهند شد. سایر تخم‌ها توسط پرنده میزبان شناسایی شده و از بین می‌روند. هرچه تخم‌های بیشتری در یک ناحیه زنده بمانند و رشد کنند به همان اندازه آن ناحیه مناسب‌تر بوده و سود بیشتری به آن منطقه اختصاص می‌یابد.

فاخته‌ها برای نجات تخم‌های خود دنبال بهترین منطقه می‌شوند. بهترین منطقه درواقع منطقه‌ای است که شباهت بیشتری به تخم‌هایشان دارد. پس از آن‌که جوجه‌ها از تخم در آمدند و تبدیل به فاخته بالغ شدند، جوامع و گروه‌هایی تشکیل می‌دهند. تمام گروه‌ها به سمت بهترین منطقه موجود فعلی مهاجرت می‌کنند. هر گروه در منطقه‌ای نزدیک بهترین موقعیت فعلی ساکن می‌شود. با درنظرگرفتن تعداد تخم‌هایی که هر فاخته خواهد گذاشت و همچنین فاصله فاخته‌ها از منطقه بهینه فعلی برای سکونت، تعدادی شعاع تخم‌گذاری محاسبه شده و شکل می‌گیرد.

فاخته‌ها شروع به تخم‌گذاری تصادفی در لانه‌هایی داخل شعاع تخم‌گذاری خود که ELR[1] نام دارد، می‌کنند. این چرخه تا رسیدن به بهترین محل برای تخم‌گذاری (منطقه با بیشترین سود) ادامه می‌یابد. این محل بهینه جایی است که بیشترین تعداد فاخته‌ها در آن گرد می‌آیند. شکل (۱) تخم‌گذاری فاخته را در محدوده ELR خودش نشان می‌دهد. پس از چند تکرار تمام جمعیت فاخته‌ها به یک نقطه بهینه با بیشینهٔ شباهت تخم‌ها به تخم‌های پرندگان میزبان و همچنین به محل بیشترین منابع غذایی می‌رسند. این محل بیشترین سود کلی را خواهد داشت و در آن کمترین تعداد تخم‌ها از بین خواهند رفت.

از آن‌جایی که نتایج با الگوریتم باکتریال و زنبور عسل نیز مقایسه می‌شوند، در ادامه الگوریتم باکتریال و زنبور عسل توصیف می‌شود.



(شکل-۱): تخم‌گذاری تصادفی فاخته با ۶ تخم در محدوده ELR
(Figure-1):Random laying cuckoo with 6 eggs in ELR range

۲-۲-۲- الگوریتم باکتریال

الگوریتم باکتریال در بسیاری از مسایل بهینه‌سازی استفاده می‌شود [36]. در این الگوریتم ابتدا لازم است تا یک جمعیت اولیه تولید شود. جمعیت اولیه الگوریتم باکتریال شامل ترکیب‌های مختلف از عبارت‌های نگهداری دید هستند که به‌صورت تصادفی تولید شده‌اند. هر کدام از این عبارت‌ها را می‌توان به‌صورت درخت ارزیابی دلتا نمایش داد. به هر یک از اعضای این مجموعه یک باکتری گفته می‌شود. این باکتری‌ها چند نسل را می‌گذرانند و در هر نسل چندین مرحله را طی می‌کنند و در هر مرحله تغییرمی‌کنند تا به درخت ارزیابی دلتای بهینه تبدیل شوند و عبارت نگهداری دید کم‌هزینه پیدا شود.

[1] Egg Laying Radius

در هر مرحله بهترین عبارت نگهداری دید انتخاب و با مرحله قبل مقایسه می‌شود تا درنهایت چند نسل طی و بهترین و کم‌هزینه‌ترین عبارت نگهداری دید انتخاب شود. آزمایش‌ها نشان می‌دهد که الگوریتم باکتریال سرعت بالاتری نسبت به روش ارزیابی دلتای بهینه دارد. حال با توجه به نتایج آزمایش‌ها مشاهده می‌شود که الگوریتم فاخته سرعت بالاتری نسبت به آن نیز دارد.

۳-۲-۲-الگوریتم زنبور عسل

الگوریتم زنبور عسل [37] نیز مانند تمام الگوریتم‌های فراابتکاری با یک جمعیت اولیه آغاز می‌شود. جمعیت اولیه شامل ترکیب‌های مختلف قابل ایجاد برای یک عبارت نگهداری دید است که به‌صورت درخت ارزیابی دلتا نمایش داده می‌شود. جمعیت اولیه همان زنبورهای دیده‌بان هستند. تمام زنبورهای دیده‌بان گلزارهای اطراف را جستجو کرده و به‌دنبال منطقه‌ای با بیشترین گرده می‌گردند. اگر منطقه‌ای که یافته شد، گرده بیشتری نسبت به منطقه قبل داشت، منطقه قبل را فراموش می‌کنند و منطقه جدید را به‌خاطر می‌سپارند؛ در غیر این‌صورت همان منطقه قبلی را به‌خاطر می‌سپارند. در انتها بقیه زنبورها به سمت بهترین منطقه حرکت می‌کنند. درواقع هر کدام از درخت‌ها، درختان اطراف خود را بررسی می‌کند و به‌دنبال بهترین درخت می‌گردند. هر درخت از میان درختان اطرافش درختی با کمترین هزینه را انتخاب کرده و به خاطر می‌سپارد. از میان درختان جمعیت اولیه، درختی با کمترین هزینه انتخاب شده و بقیه درختان سعی می‌کنند به این درخت نزدیک شوند. چندین نسل می‌گذرد و در هر نسل هر درخت عبارت ارزیابی دلتا سعی می‌کند درخت بهینه را به خاطر بسپارد و بقیه درخت‌ها نیز به سمت درخت بهینه حرکت می‌کنند، تا درنهایت بهترین منطقه یعنی درختی با کمترین هزینه پیدا شود.

۳- تعاریف اولیه

هدف بسیاری از روش‌ها و الگوریتم‌های نگهداری دید، رسیدن به کاهش هزینه و افزایش سرعت نگهداری دید است. یکی از روش‌هایی که باعث کاهش هزینه نگهداری دید می‌شود، روش‌های دسته‌بندی و گروه‌بندی است. این روش‌ها با دسته‌بندی عبارت نگهداری دید، دسترسی به بعضی روابط را کاهش می‌دهند و باعث کاهش هزینه نگهداری دید می‌شوند. در این بخش عبارت نگهداری دید افزایشی معرفی می‌شود [33].

تغییرات منابع داده در پایگاه‌داده تحلیلی به روش‌های مختلفی پردازش می‌شود. فرض می‌کنیم که دید ذخیره‌شده روی Select-Project-Join(SPJ) تعریف شده است [29]،[39]،[38] برای دید $V=R1\infty R2\infty.....\infty Rn$ که شامل دیدی با n رابطه است، عبارت نگهداری (۱) برای محاسبه تغییرات دید استفاده می‌شود.

$$\Delta v = (\Delta R1\infty R2\infty \dots \infty Rn \quad (۱)$$
$$U(\Delta R1\infty \Delta R2\infty \dots \infty Rn) U$$
$$(\Delta R1\infty \Delta R2\infty \dots \infty \Delta Rn)$$

در جمله نخست الحاق تغییرات رابطه R1 با بقیه روابط صورت گرفته است. در جمله دوم الحاق تغییرات رابطه R1 و R2 با بقیه روابط صورت گرفته است، و در جمله آخر الحاق تغییرات روابط R1 و R2 و ... و Rn صورت گرفته است. از آن‌جایی که عبارت ۱ نیازمند $(2^n-۱)$ جمله برای تغییرات دید است و باعث افزایش هزینه نگهداری دید می‌شود، یک عبارت نگهداری جدید بررسی می‌شود که فقط شامل **n** جمله است و آن را به‌عنوان عبارت نگهداری دید افزایشی در نظر می‌گیریم [40]. این عبارت هزینه کل رابطه را کاهش می‌دهد[30]،[41].

$$\Delta v = (\Delta R1\infty R2\infty.. \dots \infty Rn) \quad (۲)$$
$$U(R1'\infty \Delta R2\infty.. \dots Rn)$$
$$U(R1'\infty R2'\infty \dots .. \infty \Delta Rn)$$

در عبارت ۲، R_i ها اندازه هر رابطه است ΔR_i ها تغییرات مرتبط با هر R_i است و R_i'ها تغییر یافته هر R_i است. هزینه نهایی با استفاده از عبارت سه محاسبه می‌شود.

$$Cost_{total} = Cost_{Calc\_Time} \times Cost_{run} \quad (۳)$$

که در آن $Cost_{run}$ هزینه اجرای رابطه و $Cost_{Calc\_Time}$ مدت زمانی است که طول می‌کشد تا کم‌هزینه‌ترین رابطه یافته شود.

۴- معرفی روش ارزیابی دلتای بهینه

یکی از روش‌های نگهداری دید افزایشی، روش ارزیابی دلتای بهینه است. پیش از آن‌که روش پیشنهادی این مقاله معرفی شود، به توضیح روش ارزیابی دلتای بهینه می‌پردازیم. در بخش ششم، با بهره‌گیری از این روش، الگوریتم پیشنهادی معرفی می‌شود و با استفاده از آن سرعت نگهداری دید افزایشی افزایش می‌یابد.

روش ارزیابی دلتای بهینه یکی از روش‌های گروه‌بندی نگهداری افزایشی دید است. در این روش [30] عبارت نگهداری دید به گروه‌هایی تقسیم می‌شود و از این طریق دسترسی به برخی روابط تکراری به کمینه می‌رسد. کاهش دسترسی به برخی روابط باعث کاهش هزینه اجرای یک رابطه نگهداری دید و درنهایت باعث کاهش هزینه نگهداری دید افزایشی می‌شود. این روش از میان تمام عبارت‌های نگهداری برای یک دید، عبارت نگهداری دیدی با کمترین هزینه را پیدا می‌کند. در این مقاله هدف افزایش سرعت نگهداری دید و یافتن عبارت نگهداری دید با هزینه کمتر و در زمان کوتاه‌تری است. هر عبارت نگهداری دید را به شکل‌های مختلفی می‌توان گروه‌بندی کرد. برای مثال فرض می‌شود که یک رابطه با n=۴ داریم. برای آن ΔV به‌صورت رابطه (۴) محاسبه می‌شود.

(۴)

$$\begin{aligned}\Delta V = (&\Delta R_1 \bowtie R_2 \bowtie R_3 \bowtie R_4)\\ &\cup (R_1' \bowtie \Delta R_2 \bowtie R_3 \bowtie R_4)\\ &\cup (R_1' \bowtie R_2' \bowtie \Delta R_3 \bowtie R_4)\\ &\cup (R_1' \bowtie\bowtie R_3' \bowtie \Delta R_4)\end{aligned}$$

هزینه عبارت ۴ به‌صورت رابطه (۵) محاسبه می‌شود.

$$\begin{aligned}Cost(\Delta V) = Cost(&\Delta R1 \infty \bowtie R2 \bowtie R3 \bowtie R_4) + Cost(R'1 \bowtie \Delta R2 \bowtie R3 \bowtie R_4) + Cost(R'1 \bowtie R'2 \bowtie R_4) + Cost(R'1 \bowtie R'2 \bowtie R'3 \bowtie \Delta R_4)\\ = Cost(&|\Delta R1| + |R2| + |R3| + |R4|\\ &+ Cost(|R'1| + |\Delta R2| + |R3| + |R4|)\\ &+ Cost(|R'1| + |R'2| + |\Delta R3| + |R4|)\\ &+ Cost(|R'1| + |R'2| + |R'3| + |\Delta R4|)\end{aligned} \qquad (۵)$$

عبارت (۴) را به شکل‌های مختلفی می‌توان نوشت. در عبارت (۴)، $R_3 \bowtie R_4$ در جملات نخست و دوم و $R_1' \bowtie R_2'$ در جملات سوم و چهارم تکرار شده‌اند. برای کاهش تعداد دسترسی از هر دو رابطه جداگانه می‌توان فاکتور گرفت. عبارت (۴) را به‌صورت عبارت (۶) می‌توان بازنویسی کرد.

$$\begin{aligned}\Delta V = ((&\Delta R_1 \bowtie R_2 \cup) \bowtie R_3 \bowtie R_4) \cup (R_1' \bowtie R_2' \bowtie (\Delta R_3 \bowtie R_4 \cup R_3' \bowtie \Delta R_4))\end{aligned} \qquad (۶)$$

که از عبارت (۶) استنتاج می‌شود، تعداد دسترسی به داده‌ها نسبت به عبارت (۴) کاهش پیدا کرده است. در عبارت (۶) عبارت $\Delta R_1 \bowtie R_2 \cup R_1' \bowtie \Delta R_2$ را به‌منظور خلاصه‌نویسی می‌توان به‌صورت $\Delta (R_1 \bowtie R_2)$ تعریف و همچنین عبارت $\Delta R_3 \bowtie R_4 \cup R_3' \bowtie \Delta R_4$ را به‌منظور خلاصه‌نویسی می‌توان به‌صورت $\Delta (R_3 \bowtie R_4)$ تعریف کرد. عبارت (۶) قابل بازنویسی به‌صورت عبارت (۷) است.

$$\begin{aligned}\Delta V = (&\Delta (R_1 \bowtie R_2) \bowtie R_3 \bowtie R_4) \cup (R_1' \bowtie R_2' \bowtie \Delta (R_3 \bowtie R_4))\end{aligned} \qquad (۷)$$

عبارت (۴) را می‌توان به فرم‌های دیگری نیز نوشت. که عبارت (۸) نمونه‌ای از آن است.

$$\begin{aligned}\Delta V = (&\Delta (R_1 \bowtie R_2 \bowtie R_3) \bowtie R_4) \cup (R_1' \bowtie R_2' \bowtie R_3' \bowtie \Delta (R_4))\end{aligned} \qquad (۸)$$

همان‌طور که مشاهده می‌شود R_4 در عبارت (۴) سه بار و در عبارت (۸) یکبار تکرار شده است اگر اندازه R_4 بزرگ باشد، هزینه نگهداری افزایشی دید را به‌صورت قابل ملاحظه‌ای می‌توان کاهش داد. عبارت شماره (۴) را به شکل‌های دیگری نیز می‌توان بازنویسی کرد. عبارت‌های (۴)، (۷)، و (۸) همگی نگهداری دیدی را نشان می دهند که از ۴ رابطه تشکیل شده است و تنها در میزان هزینه با یکدیگر متقاوت هستند.

در روش ارزیابی دلتای بهینه هدف این است که با گروه‌بندی عبارت نگهداری دید، دسترسی به رابطه‌های با اندازه بزرگ‌تر کاهش داده شود و درنهایت هزینه نگهداری کاهش یابد. به‌عنوان مثال برای دیدی با سه رابطه پایه، عبارت نگهداری دید را به شکل‌های مختلف می‌توان نوشت. عبارت عبارت نگهداری دیدی که بر اساس عبارت دو نوشته شود به‌صورت عبارت نه است.

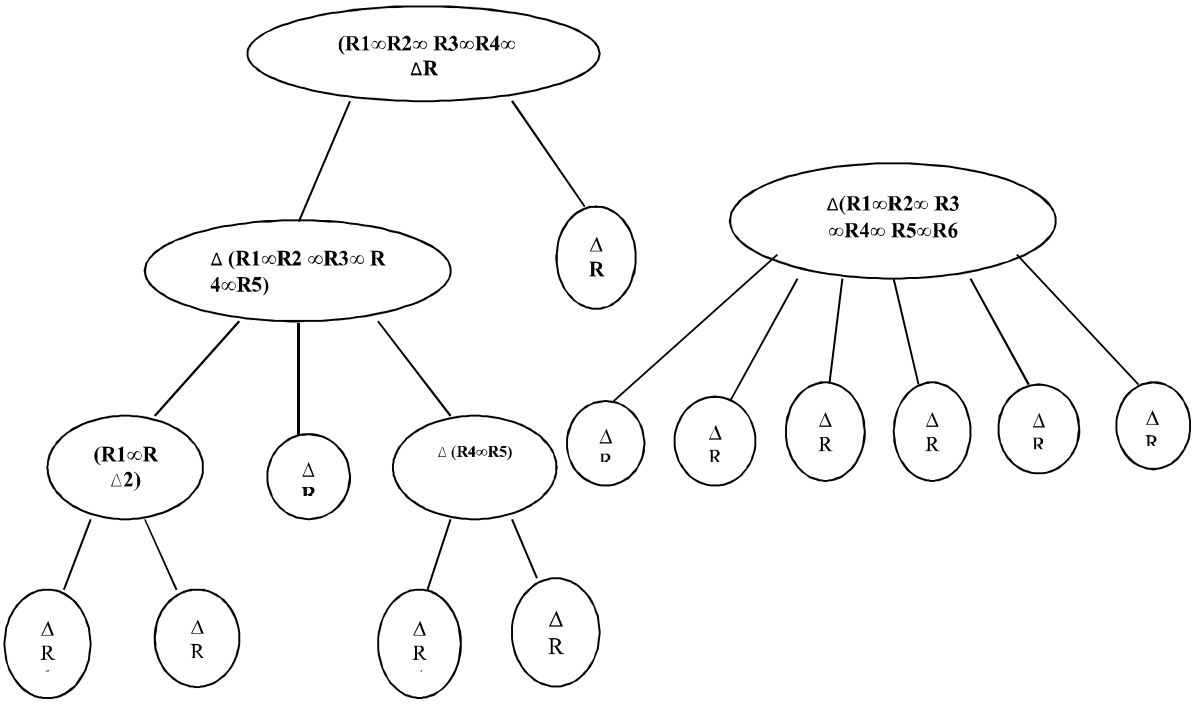
$$\begin{aligned}\Delta v = (&\Delta R1 \infty R2 \infty R3) \cup (R1' \infty \Delta R2 \infty R3) \cup (R1' \infty R2' \infty \Delta R3)\end{aligned} \qquad (۹)$$

این عبارت را به‌صورت عبارت ۱۰ نیز می‌توان نوشت.

$$\begin{aligned}\Delta v = (&((\Delta R1 \infty R2) \cup (R1' \infty \Delta R2)) \infty R3) \cup (R1' \infty R2' \infty \Delta R3)\end{aligned} \qquad (۱۰)$$

همان‌طورکه مشاهده می‌شود R_3 در عبارت ۹ دو بار تکرار شده، ولی در عبارت ۱۰ یکبار وجود دارد. اگر اندازه R_3 خیلی بزرگ باشد اندازه محاسبات را به‌صورت قابل ملاحظه‌ای می‌توان کاهش داد. در روش ارزیابی دلتای بهینه

برای پیداکردن کم‌هزینه‌ترین رابطه، لازم است تمام روابط ممکن نوشته و در انتها رابطه‌ای که کمترین مقدار را دارد، از بین آن‌ها انتخاب شود. برای پیداکردن یک رابطه با کمترین مقدار ممکن است زمان زیادی صرف شود. روش ارزیابی دلتای بهینه از یک الگوریتم پویا استفاده کرده و از میان عبارت‌های نگهداری افزایشی دید عبارت نگهداری دیدی را پیدا می‌کند که کمترین هزینه را در بین عبارت‌های دیگر دارد. این روش برای یافتن عبارتی با کمترین هزینه از درخت ارزیابی دلتا استفاده و درخت ارزیابی دلتای بهینه را از میان تمام درختان، انتخاب می‌کند. شکل (۲) چند نمونه از درخت ارزیابی دلتا را نمایش می‌دهد[30].



(شکل-۲): نمونه‌هایی از درخت ارزیابی دلتا[29]
(figure-2):Examples of Delta Evaluation Tree

۵- معرفی الگوریتم پیشنهادی

در این مقاله هدف این است که با استفاده از الگوریتم فاخته و روش دلتای بهینه رابطه بهینه نگهداری دید افزایشی در کوتاه‌ترین زمان یافته شود.

بهینه‌سازی یک مساله با مقادیر حقیقی به کمک روش‌های برنامه‌ریزی خطی قابل اجرا می‌باشد اما اگر با مساله‌ای پیچیده مواجه شویم، نظیر مسایلNP-Hard، روش‌های کلاسیک به علت خصوصیت خطی بودن کارا نمی‌باشند. یکی از روش‌هایی که امروزه برای رفع این مشکل‌ها پیشنهاد می‌شود استفاده از الگوریتم‌های هوش مصنوعی است. در این مقاله، استفاده ازالگوریتم تکاملی فاخته که برای مسائل بهینه‌سازی غیرخطی پیوسته استفاده می‌شود، پیشنهاد شده است. با استفاده از این الگوریتم بهترین عبارت نگهداری دید را در زمان کوتاه‌تری می‌توان پیدا کرده و سرعت نگهداری دید افزایشی را افزایش داد.

هدف از این مقاله این است که در زمان کمتری عبارت نگهداری دید با هزینه مناسب یافته شود، و درواقع سرعت نگهداری دید افزایشی، افزایش یابد. روش ارزیابی دلتای بهینه[30] بهترین عبارت نگهداری دید را پیدا اما زمان زیادی صرف می‌کند. برای کاهش زمان در این مقاله روش ارزیابی دلتای بهینه با الگوریتم فاخته ترکیب شده است. در ابتدا لازم است تا جمعیت اولیه الگوریتم فاخته تولید شود. جمعیت اولیه الگوریتم فاخته شامل ترکیب‌های مختلف قابل ایجاد همانند روابط (۷) و (۸) هستند که به‌صورت تصادفی تولید شده‌اند. به هر کدام از این عبارت‌ها یک فاخته می‌گوییم. هر کدام از این فاخته‌ها به دنبال بهترین محل برای تخم‌گذاری می‌گردند. در مرحله نخست، فاخته‌ها شروع به تخم‌گذاری می‌کنند. تعدادی از تخم‌ها از بین رفته و تعدادی از تخم‌ها زنده می‌مانند و رشد می‌کنند. وقتی فاخته‌ها به فاخته‌های بالغ تبدیل شوند باز به دنبال محلی بهتر برای تخم‌گذاری گشته و شروع به تخم‌گذاری

می‌کنند. دوباره تعدادی از تخم‌ها از بین رفته و یک تعداد زنده مانده و رشد می‌کنند تا درنهایت در آخر به مناسب‌ترین منطقه می‌رسند. در ابتدا فاخته‌ها مقداردهی اولیه می‌شوند و درواقع هزینه عبارت‌های نگهداری دید محاسبه می‌شود. در هر مرحله هر کدام از این فاخته‌ها شروع به تخم‌گذاری می‌کنند، درواقع هر کدام از درختان از یک تا ده درخت جدید تولید می‌کنند. در هر نسل تعدادی از فاخته‌ها از بین می‌روند و تعدادی باقی می‌مانند. درواقع عبارت‌های نگهداری دیدی که دارای هزینه زیادی هستند، از بین می‌روند و عبارت‌های با هزینه کمتر باقی می‌مانند و هزینه تمام آن‌ها محاسبه می‌شود. در مرحله بعد دوباره هر کدام از فاخته‌های باقی‌مانده شروع به تخم‌گذاری کرده و سعی می‌کنند به منطقه بهینه نزدیک شوند. درواقع هر کدام از درختان شروع به تولید درختان جدید کرده و سعی می‌کنند به درخت بهینه نزدیک شوند؛ درنهایت با گذشت چند مرحله عبارت نگهداری دید با کمترین هزینه را در زمان کوتاه‌تری می‌توان به‌دست آورد. شکل (۳) شبه کد الگوریتم فاخته را نشان می‌دهد.

```
Initial Cuckoo _Pop_Base_model1
Initial best habitat
While Condition_Algorithm
Output=Cuckoo(CuckooPop_Base_model1);
Dedicate Some Egg To Each Cuckoo
Output = SortMin(Output)
Best habitat = Output(0);
   Cuckoo_Pop_Base_model1=Generate_Pop(best
                habitat);
End While
```

(شکل-۳): شبه‌کد الگوریتم فاخته

(figure-3): Cuckoo algorithm pseudo code

برای حل یک مسئله بهینه‌سازی لازم است تا مقادیر متغیرهای مسئله به‌شکل یک آرایه شکل گیرند. در الگوریتم ژنتیک این آرایه‌ها با نام‌های «کروموزوم» و در الگوریتم بهنیه‌سازی فاخته به این آرایه habitat یا «محل سکونت» می‌گوییم؛ سپس به هر کدام از این محل‌ها تعدادی تصادفی تخم در فضای ELR آن‌ها تخصیص می‌یابد. تعدادی از این تخم‌ها توسط پرنده میزبان از بین رفته و تعدادی زنده می‌مانند و رشد می‌کنند. تخم‌های زنده‌مانده ارزیابی شده و بهترین آن‌ها به‌عنوان منطقه بعدی سکونت انتخاب شده و بقیه فاخته‌ها به آن‌جا مهاجرت می‌کنند؛ این روند ادامه پیدا می‌کند تا درنهایت بهترین منطقه شناسایی شود. هدف پیداکردن بهترین منطقه برای تخم‌گذاری است. هر چه تعداد تخم‌هایی که زنده مانده و به جوجه تبدیل می‌شوند در یک منطقه بیشتر باشد، آن منطقه بهتر است. تابع برازش تعداد تخم‌هایی است که احتمال می‌رود در یک منطقه زنده بمانند. با توجه به شبیه‌سازی‌های انجام‌شده، به این نتیجه رسیدیم که حداکثر صد نسل برای این منظور کافی است. در شکل (۴) شبه کد الگوریتم پیشنهادی آورده شده است.

```
Initial DeltaCuckoo _Pop_Base_model2
Initial best Tree
While Condition_Algorithm
Output=DeltaCuckoo(CuckooPop_Base_model2);
Create number of trees by each  trees
Output = SortMin(Output);
    Best Tree = Output()
DeltaCuckoo_Pop_Base_model2=
Generate_Pop(best Tree);
End While
```

(شکل-۴): شبه‌کد الگوریتم پیشنهادی

(figure-4): Proposed algorithm pseudo code

در ابتدا یک مجموعه از درختان برای یک عبارت نگهداری دید ایجاد می‌شود که همان فاخته‌ها هستند؛ سپس هر کدام از درختان در محدوده ELR خود شروع به تولید درختان جدید می‌کنند. درختان با هزینه بالا از بین می‌روند و درختان با هزینه کم باقی می‌مانند. این روند چند نسل ادامه پیدا می‌کند و در هر نسل درختان به درخت بهینه نزدیک می‌شوند تا درنهایت درخت بهینه به‌دست می‌آید. در تمامی روش‌های دسته‌بندی و گروه‌بندی، سعی می‌شود عبارت نگهداری دید به‌گونه‌ای نوشته شود که دسترسی به برخی روابط تکراری کاهش و با کاهش میزان محاسبات، هزینه نگهداری دید کاهش یابد.

محاسبه هزینه برای یک رابطه در الگوریتم پیشنهادی همانند روش دلتای بهینه است با این تفاوت که با توجه به شبیه‌سازی‌های انجام‌شده، یک رابطه با کمترین میزان هزینه در کمترین زمان ممکن محاسبه می‌شود.

۶- نتایج و شبیه‌سازی

روش دلتای بهینه و الگوریتم پیشنهادی معرفی‌شده در این مقاله در شرایط یکسان با استفاده از C# با سیستمی با مشخصات Microsoft Windows 7 Ultimate، RAM4GB، Hard Disk 750GB، Intel(R) Core(TM) i5-4200U CPU 1.60GHz, 2301 Mhz, 2 Core(s), 4 Logical پیاده‌سازی شده است و نتایج آن‌ها برای دیدهای مختلف که از تعداد روابط پایه مختلف تشکیل شده‌اند، با هم مورد مقایسه قرار

گرفته است. کارهای قبلی نیز برای مقایساتشان از همین تعداد روابط استفاده کرده‌اند. حاصل نتایج برای بررسی بیشتر با الگوریتم ژنتیک و باکتریال نیز مقایسه شده است. الگورتیم ژنتیک یک الگوریتم پرکاربرد برای هر زمینه کاری است، و الگوریتم باکتریال پیشنهاد قبلی نویسندگان بود [42] که نتایج خوبی داشت. در شروع پژوهش مورد نظر، ابتدا سرعت روش ارزیابی دلتای بهینه با استفاده از الگوریتم باکتریال بهبود داده شد [42]. در این مقاله سرعت روش ارزیابی دلتای بهینه، با استفاده از الگوریتم فاخته بهبود داده شد که افزایش سرعت بیشتری نیز در مقایسه با باکتریال به همراه داشت. آزمایش‌ها نشان می‌دهد الگوریتم پیشنهادی این مقاله، از الگوریتم‌های ژنتیک نیز بهتر عمل می‌کند.

این آزمایش‌ها روی دیدهای مختلف با تعداد روابط پایه مختلف انجام شده است. نتایج حاصل از شبیه‌سازی به همراه پارامترهای آن در جدول‌های (۲، ۴ و ۶) مشاهده می‌شوند.در این جدول‌ها اندازه هر یک از پارامترهای یک رابطه نگهداری دید ارائه شده است. R، R′، ΔR هر یک اجزای یک رابطه نگهداری دید هستند. به‌عنوان مثال اندازه رابطه پایه R_1 در ابتدا ۳۰ بوده بعد از تغییراتی شامل حذف و درج (به‌عنوان مثال: (ΔR_1=۱۰)به R'_1 تبدیل شده است. جدول (۲) دیدی را نشان می دهد که از چهار رابطه پایه تشکیل شده و اندازه روابط مشخص است.

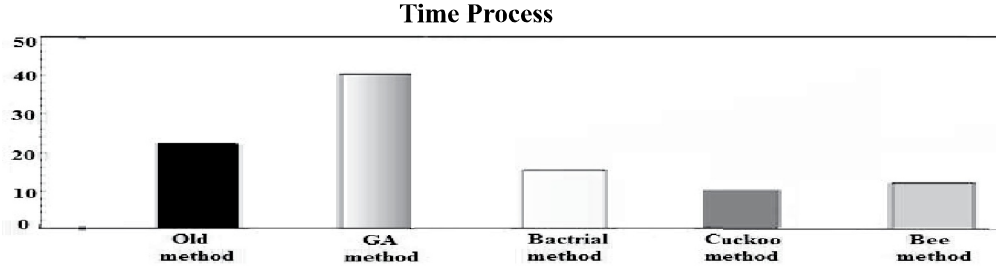
$$\Delta v = (\Delta R1 \infty R2 \infty R3 \infty R4)$$

(جدول-۲): جدول پارامترهای شبیه سازی برای دیدی با چهار رابطه پایه

(table-2): Table of simulation parameters for a view with 4 base relations

R_1	R_2	R_3	R_4
30	22	53	12
$R_1^/$	$R_2^/$	$R_3^/$	$R_4^/$
38	5	30	33
ΔR_1	ΔR_2	ΔR_3	ΔR_4
10	20	25	17

با توجه به جدول (۲)، در جدول (۳) هزینه نگهداری دید افزایشی الگوریتم فاخته نسبت به روش دلتای بهینه و الگوریتم‌های دیگر آورده شده است. با توجه به آزمایش‌های انجام‌شده در شکل (۵) مشاهده می‌کنیم که الگوریتم باکتریال حدود ۳۰ درصد، الگوریتم زنبور عسل حدود ۴۵ درصد و الگوریتم فاخته حدود ۵۰ درصد کاهشِ زمانِ اجرا دارند.



(شکل-۵): مقایسه زمان اجرای الگوریتم‌ها. از چپ: روش دلتای بهینه، ژنتیک، الگوریتم باکتریال، الگوریتم فاخته و الگوریتم زنبور عسل برای دیدی با ۴ رابطه پایه

(figure-5): Comparison of algorithms execution time. From Left: Optimal Delta Method, Genetics,Bacterial algorithm, Cuckoo algorithm and Bee algorithm for a view with 4 base relations

(جدول-۳): هزینه اجرای روش دلتای بهینه، ژنتیک، باکتریال، الگوریتم فاخته و زنبور عسل برای دیدی با چهار رابطه پایه

(table-3): Cost of executing Optimal Delta Method, Genetic algorithm,Bacterial algorithm, Cuckoo algorithm and Bee algorithm for a view with 4 base relations

Model (Old Method)	GA Method	Bacterial Method	Cuckoo Method	Bee Method
282	282	282	282	282

جدول (۴) دیدی را نشان می‌دهد که از شش رابطه پایه تشکیل شده است.

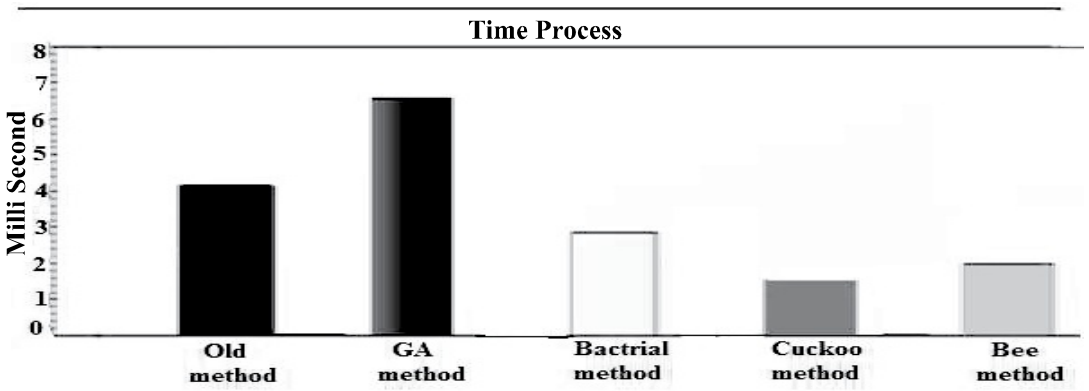
$$\Delta v = (\Delta R1 \infty R2 \infty R3 \infty R4 \infty R5 \infty R6)$$

در جدول (۵) و شکل (۶) هزینه نگهداری دید افزایشی و زمان اجرای الگوریتم فاخته نسبت به روش دلتای بهینه و الگوریتم‌های دیگر برای دیدی که از شش رابطه پایه تشکیل شده، نشان داده شده است.

(جدول-۴):جدول پارامترهای شبیه‌سازی برای دیدی با شش رابطه پایه

(table-4): Table of simulation parameters for a view with 6 base relations

R_1	R_2	R_3	R_4	R_5	R_6
20	45	13	25	12	50
$R_1^/$	$R_2^/$	$R_3^/$	$R_4^/$	$R_5^/$	$R_6^/$
23	10	8	32	13	43
ΔR_1	ΔR_2	ΔR_3	ΔR_4	ΔR_5	ΔR_6
5	30	5	7	4	20



(شکل-۶): مقایسه زمان اجرای الگوریتم‌ها. از چپ :روش دلتای بهینه، ژنتیک، الگوریتم باکتریال، الگوریتم فاخته و الگوریتم زنبور عسل برای دیدی با ۶ رابطه پایه

(figure-6): Comparison of algorithms execution time. From Left: Optimal Delta Method, Genetic algorithm,Bacterial algorithm, Cuckoo algorithm and Bee algorithm for a view with 6 base relations

با توجه به آزمایش‌های انجام شده، الگوریتم باکتریال حدود ۳۳ درصد و الگوریتم فاخته حدود ۵۵ درصد کاهش زمان اجرا دارند.

(جدول-۵): هزینه اجرای روش دلتای بهینه، ژنتیک، باکتریال، الگوریتم فاخته و زنبور عسل برای دیدی با شش رابطه پایه

(table-5): Cost of executing Optimal Delta Method, Genetics algorithm,Bacterial algorithm, Cuckoo algorithm and Bee algorithm for a view with 6 base relations

Model (Old Method)	GA Method	Bacterial Method	Cuckoo Method	Bee Method
503	503	522	522	522

با توجه به جدول (۵) و شکل (۶) مشاهده می‌شود که هزینه نگهداری دید افزایشی برای الگوریتم‌های باکتریال و فاخته حدود سه درصد نسبت به روش ارزیابی دلتای بهینه افزایش داشته است؛ درعوض الگوریتم باکتریال حدود سی و پنج، الگوریتم زنبور عسل حدود پنجاه درصد و الگوریتم فاخته حدود ۵۵ درصد زمان نگهداری دید را کاهش، درنتیجه سرعت را افزایش داده‌اند. همان‌طور که مشاهده می‌شود، با توجه به عبارت سه برایند هزینه برای الگوریتم باکتریال حدود ۳۰ درصد، برای زنبور عسل حدود ۴۵ در صد و برای الگوریتم فاخته حدود پنجاه درصد است.

جدول (۶) دیدی را نشان می‌دهد که از هشت رابطه پایه تشکیل شده است.

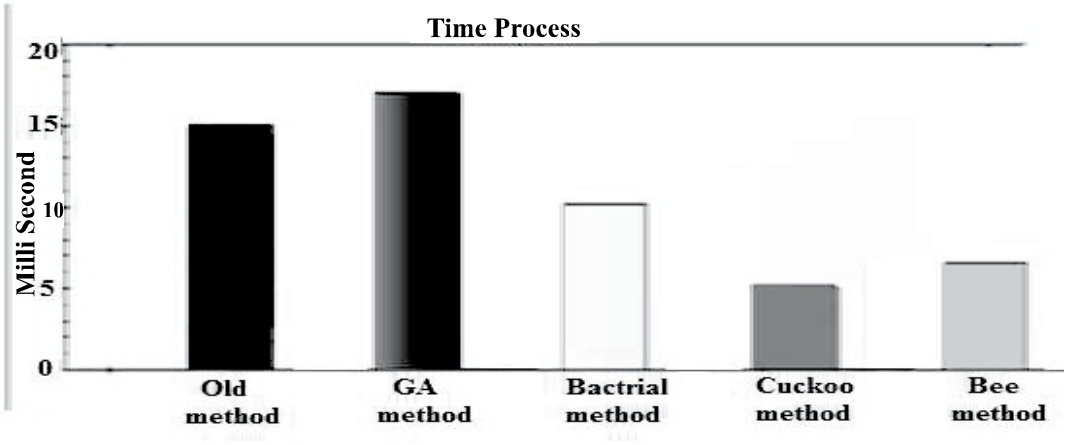
$$\Delta v = (\Delta R1 \infty R2 \infty R3 \infty R4 \infty R5 \infty\ R6 \infty R7 \infty R8)$$

(جدول-۶):جدول پارامترهای شبیه‌سازی برای دیدی با هشت رابطه پایه

(table-۶): Table of simulation parameters for a view with 8 base relations

R_1	R_2	R_3	R_4	R_5	R_6	R_7	R_8
46	2	10	25	62	50	4	32
$R_1^/$	$R_2^/$	$R_3^/$	$R_4^/$	$R_5^/$	$R_6^/$	$R_7^/$	$R_8^/$
35	11	30	10	48	43	1	28
ΔR_1	ΔR_2	ΔR_3	ΔR_4	ΔR_5	ΔR_6	ΔR_7	ΔR_8
10	9	25	10	23	20	2	5

در جدول (۷) و شکل (۷) هزینه نگهداری دید افزایشی و زمان اجرای الگوریتم فاخته نسبت به روش دلتای بهینه و الگوریتم‌های دیگر نشان داده شده است. با توجه به آزمایش‌های انجام‌شده، هزینه نگهداری دید با استفاده از الگوریتم باکتریال حدود هفت درصد، با استفاده از الگوریتم زنبور عسل حدود هشت در صد و با استفاده از الگوریتم فاخته حدود هشت درصد نسبت به روش ارزیابی دلتای بهینه افزایش داشته است؛ درعوض، الگوریتم باکتریال حدود ۳۷ درصد، الگوریتم زنبور عسل حدود ۵۵ در صد و الگوریتم فاخته حدود شصت درصد کاهش زمان اجرا دارند.



(شکل-۷): مقایسه زمان اجرای الگوریتم‌ها. از چپ: روش دلتای بهینه، ژنتیک، الگوریتم باکتریال، الگوریتم فاخته و الگوریتم زنبور عسل برای دیدی با هشت رابطه پایه

(figure-7): Comparison of algorithms execution time. From Left: Optimal Delta Method, Genetic algorithm,Bacterial algorithm, Cuckoo algorithm and Bee algorithm for a view with 8 base relations

برایند هزینه برای الگوریتم باکتریال حدود سی درصد، برای الگوریتم زنبور عسل حدود ۴۷ درصد و برای الگوریتم فاخته حدود پنجاه درصد کاهش داشته است.

(جدول-۷): هزینه اجرای روش دلتای بهینه، ژنتیک، باکتریال، الگوریتم فاخته و زنبور عسل برای دیدی با هشت رابطه پایه

(table-7): Cost of executing Optimal Delta Method, Genetic algorithm,Bacterial algorithm, Cuckoo algorithm and Bee algorithm for a view with 8 base relations

Model (Old Method)	GA Method	Bacterial Method	Cuckoo Method	Bee Method
894	894	966	977	977

همان‌طور که در نمودارها مشاهده می‌شود، با افزایش تعداد رابطه‌های پایه، عبارت نگهداری دید با کمترین هزینه در زمان کوتاه‌تری به‌دست می‌آید. با توجه به جدول (۷) و شکل (۷) مشاهده می‌شود که با افزایش تعداد رابطه‌ها هزینه نگهداری دید حدود هشت درصد افزایش داشته است؛ ولی عبارت نگهداری دید با شصت درصد کاهش زمان محاسبه شده است، و برایند آن‌ها حدود پنجاه درصد افزایش سرعت را نشان می‌دهد. به‌عبارتی می‌توان در بازهٔ زمان مشخص، تعداد دیدهای بیشتری را به‌روز و نگهداری کرد و کارایی پایگاه‌داده‌تحلیلی را بالا برد.

آزمایش‌ها با تعداد دیدهای متفاوت بررسی و جامعیت الگوریتم ثابت شد. هزینهٔ این الگوریتم نسبت به روش ارزیابی دلتای بهینه حدود هشت درصد بیشتر است؛ اما در عوض حدود پنجاه درصد افزایش سرعت دارد.

۷- نتیجه‌گیری نهایی

در این مقاله ابتدا به معرفی عبارت نگهداری دید پرداخته و سپس یکی از روش‌های نگهداری افزایشی دید به نام روش دلتای بهینه معرفی شد. این روش عبارت نگهداری دیدی را پیدا می‌کند که کمترین هزینه را دارد؛ اما زمان زیاد به‌منظور جستجوی رابطه بهینه صرف می‌کند که منجر به کاهش سرعت نگهداری دید در پایگاه‌داده تحلیلی می‌شود. برای رفع این مشکل استفاده از الگوریتم فاخته پیشنهاد شد. این الگوریتم نیز مانند دیگر الگوریتم‌های تکاملی از یک جمعیت اولیه استفاده می‌کند که شامل مجموعه‌ای از روابط برای نگهداری افزایشی دید است. در هر نسل عبارت‌های با هزینه زیاد از بین رفته و بهترین عبارت‌ها شروع به تولید عبارت‌های دیگر می‌کنند و سعی می‌کنند به بهترین عبارت

نزدیک شوند. پس از گذشت چند نسل، کم‌هزینه‌ترین عبارت نگهداری دید در زمان کوتاه‌تری به‌دست می‌آید.

با توجه به هزینه نهایی به‌دست‌آمده در بخش پیشین و مقایسه الگوریتم پیشنهادی با سایر الگوریتم‌ها و همچنین با استناد به نمودارها می‌توان نتیجه گرفت که الگوریتم فاخته و باکتریال زمان اجرای کوتاه‌تری نسبت به روش ارزیابی دلتای بهینه داشته و درنتیجه سرعت نگهداری دید بالاتری دارند. آزمایش‌ها بر روی دیدهای با تعداد منابع پایه مختلف انجام و مشاهده شد که با افزایش تعداد روابط پایه به‌روزرسانی دید با استفاده از الگوریتم فاخته با سرعت بالاتری انجام می‌شود. همان‌طور که مشاهده شد برآیند هزینه نهایی الگوریتم فاخته حدود پنجاه درصد نسبت به روش ارزیابی دلتای بهینه کاهش یافته است. در شکل‌های (۵، ۶ و ۷) سرعت اجرای الگوریتم‌ها در مقایسه با یکدیگر ملاحظه می‌شود. همان‌طور که مشاهده می‌شود، الگوریتم فاخته کمترین زمان اجرا و درنتیجه بیشترین سرعت نگهداری دید را در میان سایر الگوریتم‌ها دارد.

۸- مراجع 8-References

[۱] صباغ گل ریحانه ، دانشپور نگین ، "بهبود الگوریتم انتخاب دید در پایگاه‌داده‌تحلیلی با استفاده از یافتن پرس‌وجوهای پر تکرار"، فصل‌نامه علمی‌پژوهشی ((پردازش علائم و داده‌ها))، دوره ۱۴، شماره ۱، ۳-۱۳۹۶.

[1] Sabbagh Gol. R, and Daneshpour. N. An Improved View Selection Algorithm in Data Warehouses by Finding Frequent Queries. Journal of Signal and Data Processing, vol 14, no. 1, pp. 29-40, 1396.

[2] Almazyad A.S,.and siddiqui. m. k, Incremental View Maintenance: An Algorithmic Approach. International Journal of Electrical & Computer Sciences IJECS-IJENS 01/2010; Vol: 10:16.may 2014.

[3] Ghosh. P, Sen, S. Dynamic Incremental Maintenance Of Materialize View based on attribute affinity. International Conference on Data Science & Engineering (ICDSE), 2014.

[4] Mohapatra. A, Genesereth. M. Incremental Maintenance Of Aggregate View. Foundations of Information and Knowledge Systems. Volume 8367 of the series Lecture Notes in Computer Science, 2014, pp 399-414.

[5] Katsis. Y. Ong. K.W, Papakonstantinou. Y, Zhao. K, "Utilizing IDs to Accelerate Incremental View Maintenance", Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data.

[6] Nikolic. M. ElSeidy. M, Koch. C, "LINVIEW: incremental view maintenance for complex analytical queries", June 2014 SIGMOD '14: Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data.

[7] Yi. V.h, Xu. w.h, and taochen. Y, Novel Back Propagation Optimization by Cuckoo Search Algorithm .ScientificWorldJournal: 878262, 2014.

[8] Larson. P, Zhou. J, and Elmongui. H. G, Lazy Maintenance of Materialized Views. In Proceedings of the 33rd International conference on Very Large data bases Vienna,2007.

[9] Luo. G, Naughton. J.f, Ellmannand. C, Watzke. M, A comparison of three methods for join view maintenance in parallel RDBMS, Proceedings of ICDE Conference, pp. 177–188, 2003.

[10] Zhang.X, Ding. L, and Rundensteiner. A, Parallel multisource viewmaintenance. the VLDB Journal, 13(1):22-48, January 2004.

[11] Koch. M, An Applied Data Matching Methodology Master's Thesis, University of Kaiserslautern, December 2010.

[12] Palpanas. T, Sidle. R, Cochrane. R and Pirahesh. H, Incremental maintenance for non-distributive aggregate functions, pp. 802–813, Proceedings of VLDB Conference, 2002.

[13] Rundensteiner. E.A,and Chen. S GPIVOT: efficient incremental maintenance of compl-ex,pp, 552–563 ROLAP views, Proceedings of ICDE Conference, 2005.

[14] Wang. S Qin. B, and Du. X. Effective Maintenance of Materialized Views inPeerData Management Systems, Proceedings of the First International Conference on Semantics, Knowl-edge, 0-7695-2534-2/05 © IEEE.2006.

[15] Zhuge. Y, Molina. H. G, and Wiener. J Consistency algorithms for multi-source warehouse view maintenance. Journal of Distributed and Parallel Databases, pages 7–40,Jan. 2007.

[16] Ismail. R.M Maintenance of materialized views over peer-to-peer datawarehouse architecture, Computer Engineering & Systems (ICCES),.Page(s): 312 - 318 .IEEE Conference Publications, 2011.

[17] Jin. X, Liao. H, An incremental maintenance method for XQuery materialized view.Mecha-tronic Science, Electric Engineer-ing and Computer (MEC), International Conference on IEEE.2011.

[18] Huangand. X, Chen.Q, A maintainable model ofmaterialized view bas-ed on datawarehouse,Mechatronic Science, Electric Engineering and Computer (MEC), 2011 International Conference , Page(s): 1974 - 1977 .IEEE Conference Publications, 2011.

[19] Dattaand. S,.Chaki.D.N,An Architectureto Maintain Materialized View in Cloud Computing Environment for OLAP ProcessingComputing Sciences (ICCS), Page(s): 360 - 365 .IEEE Conference Publications.InternationalConferenceon2012.

[20] Jainand. H, and Gosain. A, A comprehensive study of view maintenance approaches in data warehousing evolution. ACM SIGSOFT Software Engineering Notes archive. Volume 37 Issue 5, September 2012

[21] Jörg. T and Behrend. A, Optimized Incremental ETL Jobs for Maintaining Data Warehouses In: Proc. IDEAS, pp. 216-224 2010.

[22] Gupta. A, Folkert. N, Witkowski. A, Subrmanian. S, Bellamkonda.S, Shankar.S, Bozgaya.T and Sheng.L, "Opt imisingRegresh Set of Materialized View". Proceedings of VLDB Conference, Norway, 2005.

[23] Almazyad A.S, siddiqui. m. k Ahmad. Y, Kan. Z, A Incremental view Maintenance Approach Using Version Store in Warehousing Environment. Computer Science and Engineering. WCSE 09 Second International Workshop on Volume :1.Dol: 10.1109/WCSE. 2009.

[24] Zhou. L and.Geng.Q. H, The minimum Incremental Maintenance of Materialize View in Data Warehouse2nd International Asia Conference on Informatics in Control. © IEEE, 2010.

[25] Yeung. H, Gary, C, and Gruver.W A, Multiagent Immediate Incremental View Maintenance for Data Warehouses. for Data Warehouses. IEEE TRANSACTIONS ON SYSTEMS, MARCH 2005.

[26] Zhuge.Y,Molina. H G, Hammer. J, and Widom. J, View maintenance in a warehousing environment. In Proceedings of SIGMOD, pages 316--327, May 1995.

[27] Zhuge. Y,.Molina. H. G, Hammer. J, and Wiener.J. L, The strobe algorithms for multisource warehouse consistency. In Proceedings of the Fourth International Conference on Parallel and Distributed Information Systems, pages 146- 157, December 1996.

[28] Agrawal. D, Abbadi. A. E, Singh. A, and Yurek. T, Efficient view maintenance at data warehouses. In Proceedings of SIGMOD, pages 417-427, May 1997.

[29] Zhang. X, Yangand. L, Wang. D, Incremental View Maintenance Based on Data Source Compensation in Data Warehouses. International Conference on Computer Application and System ModelingICCASM 2010.

[30] Lee. K. Y, Son. J. H, and Kim. M, "Reducing the cost of accessing relations in incremental view maintenance", Decision Support Systems 43 512–526, 2007.

[31] Liu.B, and Rundensteiner. E.A, Finkel.D, Maintaining large update batches by restructuring and grouping.Informatio Systems 32 621–639.www.elsevier.com /locate/infosys, 2007.

[32] He. H, Xie.J, Yangand. J, Yu. H, Asymmetric Batch Incremental View Maintenance.Browse Conference Publications. Data Engineering, ICDE .IEEE, 2005.

[33] Zhou. J, Larson.P, and Elmongui. H.G, Lazy Maintenance of Materialized Views. In Proceedings of the 33rd International conference on Very Large data bases,Vienna, Austria, 2007.

[34] Rajabioun.R, "Cuckoo Optimization Algorithm". Control and Intelligent Processing Center of Excellence (CIPCE), Scool of Electrical and Computer Engineering, University of Tehran, Tehran, Iran. Journalhome page : www.elsevier.com Applied Soft Computing 11 (2011) 5508-5518.

[35] Esmonde. W, G.kanagaraj. L and ponnambalam. S.G, PCB Drill Path Optimization by Combinatorial Cuckoo Search Algorithm .The Scientific World Journal,Volume2014.Data Science & Engineering (ICDSE).Incremental Conference on Dol:10.1109/ICDSE. 2014.

[36] Buruzs. A, Hatwagner. M F, and Pozna. R C, Advanced Learning of Fuzzy Cognitive Maps of Waste Management by Bacterial Algorithm. 978-1-4799-0348-1/13/$31.00 ©IEEE, 2013.

[37] Y. ujang, Yi . Renjie He.A Novel Artificial Bee Colony Algorithm Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2014 Sixth International Conference on Volume: 1 DOI: 10.1109/IHMSC.2014.73 Publication Year: 2014 .

[38] Yang. J, Yi. K, Yu, H, Xia. G, Chen. Y, Efficient maintenance of materialized top-k views, Proceedings of the ICDE Conference, 2003, pp. 189–200.

[39] Griffin. T, and Libkin.L, Incremental maintenance of views with duplicates. In Proc. SIGMOD, 2007.

[40] Gupta. A, Mumick.I. S, and Subrahmanian. V. S, Maintaining Views incrementally, Proceeding of ACM SIGMOD Conference, pp.157-166, 1993.

[41] Jörg. T and Dessloch.S, View Maintenance using Partial Deltas In: Proc. BTW, LNI P - 180, pp. 287-306 March 2011.

[۴۲] کریمی مصدق عفیفه ،. دانشپور نگین ، "کاهش هزینه نگهداری افزایشی دید پایگاه‌داده‌تحلیلی با استفاده از الگوریتم‌های فراابتکاری"، سومین کنفرانس بین المللی اطلاعات،حال و آینده ۲۰۱۴.

[42] Karimi Mosadegh A, and Daneshpour N. Incremental View Maintenance Cost Reduction in Data Warehouses using Meta Heuristic Algorithms. In Proceedings of the 3rd International Conference on Present and Future Information, 2014.

عفیفه کریمی مصدق مدارک کارشناسی خود را در رشته مهندسی کامپیوتر- نرم افزار در سال ۱۳۸۸ و در سال ۱۳۹۳ کارشناسی ارشد خود را در رشته مهندسی کامپیوتر- نرم‌افزار از دانشگاه آزاد قزوین اخذ کرده است. موضوع پایان نامه ایشان نگهداری دید در پایگاه داده تحلیلی بود.
نشانی رایانامه ایشان عبارت است از:

karimi.mosadegh@yahoo.Com

نگین دانشپور استادیاردانشکده مهندسی کامپیوتر دانشگاه تربیت دبیر شهیدرجایی است. نامبرده تحصیلات خود را در مقطع کارشناسی مهندسی کامپیوتر- سخت‌افزار در سال ۱۳۷۸ در دانشگاه شهیدبهشتی و کارشناسی ارشد مهندسی کامپیوتر نرم‌افزار را در سال ۱۳۸۱ در دانشگاه صنعتی امیرکبیر به پایان رسانده و در سال ۱۳۸۹ دکترای خود در رشته مهندسی کامپیوتر- نرم‌افزار را از دانشگاه صنعتی امیرکبیر اخذ کرده است. زمینه‌های پژوهشی مورد علاقه ایشان عبارتند از: پایگاه داده، پایگاه‌داده تحلیلی، سامانه‌های تصمیم یار و داده‌کاوی.
نشانی رایانامه ایشان عبارت است از:

ndaneshpour@srttu.edu