# بهبود عبارت نگهداری دید در پایگاه داده تحلیلی

سید مهدی قریشی[۱] و نگین دانشپور[۲*]

**اطلاعات مقاله**

دریافت مقاله: ۱۳۹۴/۰۹/۲۹

پذیرش مقاله: ۱۳۹۶/۰۷/۲۲

**واژگان کلیدی:**

نگهداری افزایشی دید،

پایگاه داده تحلیلی،

عبارت نگهدای دید،

دید ذخیره شـــده در پایگاه داده تحلیلی.

**چکیده**

در پایگاه داده تحلیلی برای پاسخگویی سریع به پرس و جوهای تحلیلی کاربران، پاسخ تعدادی از پرس و جوها را ذخیره می‌نمایند. پاسخ پرس و جوهایی که در پایگاه داده تحلیلی ذخیره می‌شوند، منجر به تولید دیدهای ذخیره شده می‌گردد. مساله بروز رسانی و نگهداری از دیدها در پایگاه داده تحلیلی زمانی مطرح می‌گردد که داده‌ها در منابع داده پایه بوسیله تراکنش‌های مختلف مورد تغییر قرار گیرند. برای نگهداری از دیدها در پایگاه داده تحلیلی، عبارت‌های نگهداری دید متفاوتی ارائه شده است. در این مقاله الگوریتمی برای تولید عبارت نگهداری دید بهبود یافته‌ای ارائه می‌شود. الگوریتم ارائه شده از دو بخش تشکیل شده است. در بخش اول از الگوریتم پیشنهادی، درختی تشکیل می‌گردد که در آن تعداد دسترسی به منابع داده‌ای بزرگ (منابعی با تعداد رکورد و حجم بالا)، کاهش داده می‌شود. در بخش دوم بر اساس درخت تولید شده در بخش اول و اندازه تغییرات منابع داده(منابع داده پایه و گره‌های میانی درخت)، عبارت نگهداری دید بهبود یافته‌ای برای بروز رسانی پایگاه داده تحلیلی ارائه می‌گردد. نتایج آزمایش‌ها نشان می‌دهد که عبارت نگهداری دید ارائه شده، زمان و هزینه نگهداری از پایگاه داده تحلیلی را در مقایسه با روش‌های ارائه شده قبلی کاهش داده است.

## ۱- مقدمه

پایگاه داده تحلیلی[۱] شامل مجموعه‌ای از داده‌ها می‌باشد که از منابع داده مستقل و ناهمگن استخراج شده است. منابع داده پایه مدام در حال تغییر و بروز رسانی هستند، به همین دلیل برای پاسخ‌گویی مناسب به پرس و جوهای تحلیلی کاربران دیدهای ذخیره شده در پایگاه داده تحلیلی[۲] باید بروز رسانی و نگهداری گردند[۱]. هنگامی که منابع داده پایه دستخوش تغییر می‌گردند، دیدهای ذخیره شده در پایگاه داده تحلیلی نیز باید بروز رسانی و نگهداری گردد. پایگاه داده تحلیلی در هر لحظه میزبان تعداد زیادی از پرس و جوهای تحلیلی کاربران می‌باشد. بروز بودن پایگاه داده تحلیلی نقش زیادی در پاسخگویی درست به پرس و جوهای تحلیلی کاربران دارد، به نحوی که بروز نبودن آن نه تنها موجب پاسخگویی صحیح به پرس و جوها نمی‌گردد بلکه گمراه کننده نیز می‌تواند باشد. به دلیل اینکه حجم داده‌ها در پایگاه داده تحلیلی بسیار زیاد می‌باشد، بنابراین زمان زیادی صرف نگهداری از پایگاه داده تحلیلی می‌گردد. برای بروز رسانی پایگاه داده تحلیلی می‌توان دیدهای ذخیره شده‌ی قبلی را حذف نمود و دیدهای مورد نیاز را بر اساس منابع داده پایه دوباره محاسبه و ذخیره نمود[۱]. علاوه بر این، می‌توان به همراه دیدها منابع داده کمکی را ذخیره نمود که بروز رسانی دیدها به صورت خود نگهدار[۳] باشد[۱۷-۱۹].

در نگهداری دید به صورت افزایشی[۴]، تغییرات منابع داده

*پست الکترونیک نویسنده مسئول: ndaneshpour@sru.ac.ir

۱. کارشناسی ارشد، دانشکده مهندسی کامپیوتر، دانشگاه تربیت دبیر شهیدرجایی.

۲. استادیار، دانشکده مهندسی کامپیوتر، دانشگاه تربیت دبیر شهیدرجایی.

[1] -Data Warehouse
[2] -Materialized view
[3] Self-Maintenance
[4] Incremental View Maintenance

ایه محاسبه می‌گردد و عمل بروز رسانی پایگاه داده تحلیلی از طریق انتشار این تغییرات بر روی دیدها انجام می-شود.[۲-۱۶].

در روش دوباره سازی دیده‌ها [۱] به دلیل حجم زیاد داده‌ها در منابع داده پایه و صرف زمان زیاد برای ساختن آنها، استفاده از این روش مناسب نمی‌باشد. در نگهداری از دیدها به صورت خود نگهدار[۱۷-۱۹]، علاوه بر بروز رسانی دیدها باید منابع داده کمکی را نیز بروز رسانی نمود. از طرفی با توجه به حجم بالای تغییرات در منابع داده پایه بروز رسانی این منابع داده کمکی نیز خود مستلزم صرف زمان زیادی می‌باشد، بنابراین نگهداری دید به صورت افزایشی[۲-۱۶] به دلیل عملکرد مطلوب و کاهش زمان بروز رسانی بیشتر مورد توجه بوده است. در این مقاله نیز با استفاده از روش نگهداری دید افزایشی، عبارت نگهداری دید بهبود یافته‌ای ارائه می‌گردد که زمان و هزینه نگهداری از پایگاه داده تحلیلی را کاهش دهد.

این مقاله شامل پنج بخش می‌باشد. در بخش دوم کارهای مرتبط انجام شده در زمینه تولید عبارت نگهداری دید ارائه می‌گردد. در بخش سوم عبارت نگهداری دید بهبود یافته‌ای ارائه می‌شود که زمان و هزینه نگهداری از پایگاه داده تحلیلی را در مقایسه با کارهای انجام شده قبلی کاهش می‌دهد. بخش چهارم نتیجه‌ی مقایسه عبارت نگهداری دید ارائه شده با سایر عبارتهای نگهداری دید و همچنین تابع هزینه را بیان می‌کند و در بخش پنجم نتیجه‌گیری کار ارائه می‌گردد.

## ۲- کارهای مرتبط انجام شده

برای کاهش زمان بروز رسانی و نگهداری در پایگاه داده تحلیلی روش‌ها و الگوریتم‌های مختلفی ارائه گردیده است. روش پایه برای بروز رسانی پایگاه داده تحلیلی محاسبه دوباره دید بر اساس منابع داده پایه می‌باشد[۲۱-۲۰]. در روش پایه به جهت اینکه حجم منابع داده در پایگاه داده تحلیلی بسیار زیاد می‌باشد، بنابراین زمان زیادی برای نگهداری دیدها در پایگاه داده تحلیلی نیاز داریم. از روشهای مناسب برای نگهداری دید می‌توان به نگهداری دید به صورت افزایشی اشاره کرد که کارهای مختلفی در این زمینه برای بروز رسانی پایگاه داده تحلیلی انجام شده است[۱۶-۲]. در زمینه نگهداری افزایشی دید کارهای مختلفی انجام شده است. تعدادی از کارها بر روی دیدهایی با قالب دیدهای مجتمع[5] انجام شده است. [۶, ۷]. تعدادی از کارها بر روی دیدهای Select-Project-Join (SPJ) ارائه شده است [۱۲,۱۱,۸,۴] و همچنین نیز تعدادی بر روی دیدهای با قالب outer join ارائه گردیده است[۹, ۱۰].

در این مقاله از دیدهای با قالب SPJ استفاده می‌گردد که یکی از دیدهای پرکاربرد در پایگاه داده تحلیلی می‌باشد. یکی از این روش‌ها در حوزه نگهداری دید افزایشی، تولید عبارت نگهداری دیدی است که زمان نگهداری(اعمال تغییرات منابع داده پایه بر روی پایگاه داده تحلیلی) را کاهش دهد[۱۱-۱۳]. عبارت‌های نگهداری دید متفاوت، دارای تعداد جملات متفاوتی نیز هستند. در روش[۱۴] برای یک دید با $m$ منبع داده پایه همانند $V = R_1 \infty R_2 \infty \ldots \infty R_m$، عبارت نگهداری دید به صورت رابطه (۱) ارائه می‌گردد.

$$\begin{aligned}\Delta V = & \left(\Delta R_1 \infty R_2 \infty \ldots \infty R_m\right) \\ & U\left(R_1 \infty \Delta R_2 \infty \ldots \infty R_m\right) \\ & \cdot \\ & \cdot \\ & \cdot \\ & U\left(\Delta R_1 \infty \Delta R_2 \infty \ldots \infty \Delta R_m\right)\end{aligned} \qquad (۱)$$

عبارت $\Delta R_j$ در رابطه (۱) بیانگر تغییرات منبع داده $R_j$ می‌باشد. در روش [۱۴] برای یک دید با $m$ منبع داده پایه، تعداد جملات تشکیل دهنده عبارت نگهداری دید برابر با $2^m - 1$ می‌باشد. در این روش با توجه به اینکه تعداد جملات در عبارت نگهداری دید زیاد می‌باشد لذا زمان زیادی برای بروز رسانی پایگاه داده تحلیلی صرف خواهد شد. تعداد جملات یک عبارت نگهداری دید، تاثیر زیادی در بهینه بودن زمان نگهداری دارد. به طور کلی به هر میزان که تعداد جملات یک عبارت نگهداری دید کاهش یابد، زمان نگهداری دیدها نیز کاهش خواهد یافت. برای کاهش تعداد جملات عبارت نگهداری دید روش جدیدی در مرجع [۱۵] ارائه گردید. در روش [۱۵] برای یک دید با $m$ منبع داده پایه، تعداد جملات تشکیل دهنده‌ی عبارت نگهداری دید برابر با $m$ می‌باشد. در این روش برای دیدی همانند $V = R_1 \infty R_2 \infty R_3 \infty R_4$ که از چهار منبع داده پایه تشکیل شده است عبارت نگهداری دیدی همانند رابطه (۲) ارائه

[5] Aggregate View

می‌گردد.

$$\Delta V = \begin{array}{l}(\Delta R_1 \infty R_2 \infty R_3 \infty R_4) \\ U\ \left(R_1^{'} \infty \Delta R_2 \infty R_3 \infty R_4\right) \\ U\ \left(R_1^{'} \infty R_2^{'} \infty \Delta R_3 \infty R_4\right) \\ U\ \left(R_1^{'} \infty R_2^{'} \infty R_3^{'} \infty \Delta R_4\right)\end{array} \qquad (۲)$$

عبارت $R_1^{'}$ $(R_1^{'} = \Delta R_1 U R_1)$ در رابطه (۲) بیانگر اعمال تغییرات بر روی منبع داده $R_1$ می‌باشد. در رابطه (۲) تعداد دسترسی‌ها به هر کدام از منابع داده پایه برابر با سه می‌باشد. در صورت داشتن اطلاعاتی در مورد حجم داده‌ای منابع داده پایه می‌توان رابطه (۲) را بهبود بخشید. در روش ارائه شده در مرجع [۱۵] توجهی به اندازه دسترسی ها به منابع داده‌ای با حجم بالا در عبارت نگهداری دید نشده است، این در حالی است که کاهش دسترسی‌ها به منابع داده‌ای با حجم بالا تاثیر قابل توجهی در کاهش زمان نگهداری دیدها در پایگاه داده تحلیلی خواهد داشت. به عنوان مثال اگر اندازه منبع داده‌ی $R_1$ و $R_2$ در مقایسه با منابع داده دیگر بزرگتر باشد، آنگاه با تغییر رابطه (۲) به صورت رابطه (۳) می‌توان تعداد دسترسی‌ها به منابع $R_1$ و $R_2$ را کاهش داد. در رابطه (۳) تعداد دسترسی‌ها به منابع داده‌ای $R_2$ و $R_1$ به ترتیب برابر با دو و یک مرتبه می‌باشد. این در حالی است که در رابطه (۲) تعداد دسترسی‌ها به منابع داده‌ای $R_2$ و $R_1$ سه مرتبه می‌باشد. در عبارت نگهداری دید، با کاهش دسترسی‌ها به منابع داده‌ای بزرگ می‌توان زمان و هزینه نگهداری از پایگاه داده تحلیلی را کاهش داد[۱۲].

$$\Delta V = \begin{array}{l}\left(\begin{array}{l}\left(\begin{array}{l}\left(\left(\left(\left(\Delta R_3 \infty R_4\right)\ U\ \left(R_3^{'} \infty \Delta R_4\right)\right) \infty R_2\right) U \\ \left(R_3^{'} \infty R_4^{'} \infty \Delta R_2\right)\end{array}\right) \infty R_1\end{array}\right) \\ U\ \left(R_3^{'} \infty R_4^{'} \infty R_2^{'} \infty \Delta R_1\right)\end{array} \qquad (۳)$$

به عبارتی رابطه (۳) نتیجه‌ی افراز منابع داده‌ای به صورت $T_1 = \{R_2, R_3, R_4\}$ و $T_2 = \{R_1\}$ می‌باشد. افراز $T_1 = \{R_2, R_3, R_4\}$ نیز از دو افراز دیگر به صورت $T_1 = \{R_3, R_4\}$ و $T_2 = \{R_2\}$ تشکیل می‌گردد. بنابراین رابطه (۳) را می‌توان به صورت رابطه (۴) بیان نمود.

$$\begin{array}{l}\Delta(V) = \left(\Delta\left(R_2 \infty R_3 \infty R_4\right) \infty R_1\right) \cup \left(R_2^{'} \infty R_3^{'} \infty R_4^{'} \infty \Delta R_1\right) \qquad (۴) \\ \Delta\left(R_2 \infty R_3 \infty R_4\right) = \left(\Delta(R_3 \infty R_4) \infty R_2\right) \cup \left(R_3^{'} \infty R_4^{'} \infty \Delta(R_2)\right) \\ \Delta\left(R_3 \infty R_4\right) = \left(\Delta(R_3) \infty R_4\right) \cup \left(R_3^{'} \infty \Delta(R_4)\right)\end{array}$$

رابطه (۲) را از جنبه ترتیب اعمال تغییرات منابع داده پایه نیز می‌توان بهبود بخشید. به عنوان مثال اگر در رابطه (۲) اندازه تغییرات روابط پایه به صورت $\left|R_2^{'} - R_2\right| < \left|R_1^{'} - R_1\right| < \left|R_3^{'} - R_3\right|$ باشد، در این صورت برای کاهش هزینه و زمان نگهداری در پایگاه داده تحلیلی، بهتر است که ابتدا تغییرات منبع داده $R_2$ سپس $R_1$ و در نهایت $R_3$ اعمال گردد[۱۳]. در روش [۱۲] تعداد دسترسی‌ها به منابع داده‌ای با حجم بالا کاهش داده شد، اما توجهی به اندازه تغییرات منابع داده پایه نشده است. در واقع با توجه به متفاوت بودن ویژگی‌ها و کاربردهای منابع داده پایه در پایگاه داده تحلیلی، تغییرات این منابع داده نیز متفاوت خواهد بود. به عبارت دیگر، اعمال ترتیب مناسب با توجه به حجم و اندازه تغییرات منابع داده پایه نیز تاثیر قابل توجهی در کاهش زمان و هزینه عبارت نگهداری دید خواهد داشت. رابطه (۳) به گونه‌ای است که دسترسی به منابع داده بزرگ را کاهش می‌دهد، اما توجهی به ترتیب اعمال تغییرات بر روی منابع داده پایه ندارد. بنابراین در این مقاله عبارت نگهداری دید بهبود یافته‌ای ارائه می‌گردد که علاوه بر کاهش دسترسی به منابع داده بزرگ، ترتیب مناسبی برای اعمال تغییرات منابع داده پایه و گره‌های میانی بر روی دید نهایی داشته باشد.

## ۳- الگوریتم پیشنهادی برای تولید عبارت نگهداری دید بهبود یافته

در این مقاله عملیات نگهداری بر روی دیدهایی با قالب SPJ اعمال می‌گردد. در الگوریتم پیشنهادی عبارت نگهداری دید بهبود یافته‌ای ارائه می‌گردد که زمان و هزینه نگهداری از پایگاه داده تحلیلی را در مقایسه با کارهای انجام شده قبلی کاهش دهد[۱۱-۱۳]. عبارت‌های نگهداری دید بر اساس تعداد منابع داده‌ای و تعداد دسترسی‌ها به منابع داده‌ای، دارای هزینه نگهداری متفاوتی می‌باشند. بنابراین ما به عبارت نگهداری دید بهبود یافته‌ای نیاز داریم، که زمان و هزینه نگهداری از دیدها را کاهش دهد. عبارت نگهداری دید ارائه شده به‌گونه‌ای است که در آن تعداد دسترسی‌ها به منابع داده بزرگ در حد امکان کاهش می‌یابد و اعمال تغییرات منابع داده‌ای با ترتیب مناسبی انجام می‌شود. الگوریتم پیشنهادی برای تولید عبارت نگهداری دید بهبود یافته دارای دو بخش می‌باشد. در بخش ۳-۱ درختی برای کاهش دسترسی به منابع داده‌ای بزرگ ارائه

می‌گردد و در بخش ۳-۳ از الگوریتم پیشنهادی دنباله‌ای مناسب برای اعمال تغییرات منابع داده‌ای و تولید عبارت نگهداری دید بهبود یافته ارائه می‌گردد. در بخش ۳-۲ تحلیل الگوریتم ارائه شده در بخش ۳-۱ ارائه می‌گردد.

### ۳-۱- ساخت درخت بهینه

همانطور که گفته شد برای یک دید در پایگاه داده تحلیلی عبارت‌های نگهداری دید متفاوتی ارائه شده است[۱۱-۱۳]. روش [۱۵] به دلیل داشتن تعداد جملات کم در عبارت نگهداری دید، از روش‌های مناسب برای بروز رسانی پایگاه داده تحلیلی می‌باشد. در روش [۱۵] بین منابع داده‌ای با اندازه‌های مختلف تفاوتی قائل نمی‌گردد. این در حالی است که کاهش دسترسی به منابع داده بزرگ نقش زیادی در کاهش زمان نگهداری دیدها دارد[۱۲]. بنابراین در رابطه (۱) برای مطلوبتر شدن عبارت نگهداری دید، تلاش می‌گردد تا تعداد دسترسی به منابع داده بزرگ را کاهش دهیم. همانطور که در رابطه (۳) مشاهده می‌شود با عمل فاکتور-گیری تعداد دسترسی‌ها به منابع داده بزرگ $(R_1, R_2)$ کاهش داده شده است.

افرازهای متفاوت بر روی منابع داده پایه در عبارت نگهداری دید رابطه (۴) موجب گردید، تا تعداد دسترسی به منابع داده بزرگ $(R_1, R_2)$ کاهش داده شود.. به عبارتی با استفاده از افرازهای متفاوت بر روی منابع داده پایه، درخت نگهداری ارائه می‌گردد که در آن زمان و هزینه نگهداری کاهش - یابد[۱۲]. در حالت کلی اگر افرازهای انجام شده بر روی دیدی با $n$ منبع داده پایه همانند $V = R_1 \infty R_2 \infty \ldots \infty R_n$ به صورت $T = \{T_1, T_2, \ldots, T_m\}$ باشد، آنگاه اعضای مجموعه $T$ باید دارای شرایط زیر باشد:

۱- اجتماع همه افرازها برابر با مجموعه منابع داده پایه باشد.

۲- افرازها با یکدیگر اشتراک نداشته باشند.

۳- هیچکدام از افرازها تهی نباشند.

بنابراین با توجه به شرایط مذکور عبارت نگهداری دید به صورت رابطه (۶) تعریف می‌گردد.

$$\begin{aligned}\Delta V = &\left(\Delta(T_1) \infty (T_2) \infty \ldots \infty (T_m)\right)\\ &U\left((T_1)' \infty \Delta(T_2) \infty \ldots \infty (T_m)\right)\\ &\cdot\\ &\cdot\\ &U\left((T_1)' \infty (T_2)' \infty \ldots \infty \Delta(T_m)\right)\end{aligned} \qquad (۶)$$

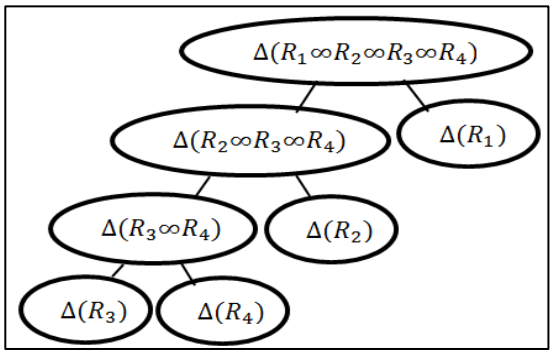
در رابطه (۵) عبارت‌های $T_i$ و $T_i'$ هر کدام بیانگر زیر مجموعه‌ای از مجموعه منابع داده پایه می‌باشند که بر روی افراز $T_i = \{R_i, R_j, \ldots, R_k\}$ تعریف شده است(i,k ثابت عددی است و k>i). با توجه به مطالب ذکر شده، برای افرازهای متفاوت از منابع داده پایه، می‌توان عبارت‌های نگهداری دید متفاوتی بدست آورد. بنابراین درخت‌های بهینه‌ای که بر اساس افرازهای مختلف ساخته می‌شود هر کدام دارای هزینه‌ای متفاوت خواهند بود. برای درک بهتر مساله افرازهای متفاوت بر روی منابع داده پایه، مثال (۱) ارائه شده است.

مثال ۱) دیدی با چهار منبع داده پایه را در نظر بگیرید. برای این دید بر روی مجموعه منابع داده پایه $R = \{R_1, R_2, R_3, R_4\}$ افرازهای زیر انجام می‌شود.

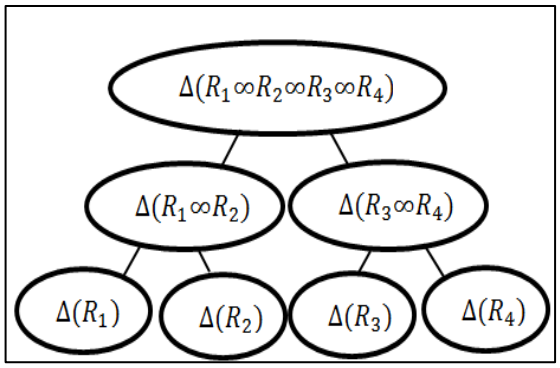
الف) $T_1 = \{R_2, R_3, R_4\}\ ,\ T_2 = \{R_1\}$

ب) $T_1 = \{R_1, R_2\}\ ,\ T_2 = \{R_3, R_4\}$

برای هر کدام از افرازهای بالا درخت‌های نگهداری متفاوتی ارائه می‌گردد. شکل (۱) نشان دهنده‌ی درخت نگهداری ارائه شده برای افراز حالت (الف) می‌باشد. شکل (۲) نشان دهنده‌ی درخت نگهداری ارائه شده برای افراز حالت (ب) می‌باشد.




شکل ۱- درخت نگهداری ارائه شده برای افراز حالت (الف)




شکل ۲- درخت نگهداری ارائه شده برای افراز حالت (ب)

در شکل (۱) تعداد دسترسی‌ها به منابع داده پایه $R_1$، $R_2$، $R_3$ و $R_4$ به ترتیب برابر با ۳،۳،۲،۱ می‌باشد و در شکل (۲) تعداد دسترسی‌ها به هر کدام از منابع داده پایه برابر با ۲ می‌باشد. بنابراین اندازه تابع هزینه خطی برای درخت‌های ساخته شده در شکل (۱) و شکل (۲) متفاوت خواهد بود. اگرمنبع داده‌ای $R_1$ حجم و اندازه بالایی نسبت به منابع دیگر داشته باشد، با توجه به اینکه در درخت شکل (۱) تعداد دسترسی‌ها به منبع داده‌ای $R_1$ کمتر می‌باشد، بنابراین هزینه عبارت نگهداری دید معادل با افراز انجام شده در حالت (الف) کمتر خواهد بود. در حالت کلی اگر افرازها به گونه‌ای انجام گردد که تعداد دسترسی‌ها به منابع داده‌ای با حجم بالا کمتر شود، می‌توان هزینه عبارت نگهداری دید را کاهش داد.

در حالت کلی برای دیدی که از $m$ منبع داده پایه تشکیل شده است، تعداد افرازهایی ممکن بر روی مجموعه منابع داده پایه برابر با $m^m$ می‌باشد[۱۶]. برای هر افرازی می‌توان درخت نگهداری متفاوتی ارائه نمود، بنابراین لازم است از بین افرازهای موجود بهترین افراز یا بهترین درخت برای کاهش زمان و هزینه نگهداری ارائه شود. در شکل (۳) الگوریتمی ارائه می‌گردد که در آن از بین درخت‌های نگهداری ارائه شده، درختی که دارای کمترین هزینه نگهداری است، انتخاب می‌گردد.

Calculate incremental view maintenance with view maintenance expression
Input:
Materialized View V= $R_1 \infty R_2 \infty \ldots \infty R_n$.
Basic relation $\{R_1 \infty R_2 \infty \ldots \infty R_n\}$
Output:
The lowest access to large resource with the optimal tree.
// Initialized
Construct all subset of base data source set R= $\{R_1, R_2, R_3 \ldots\ldots R_n\}$ with binary value.
Insert the subsets to *all_subset set.*
Begin
- Initialized j=2;
- While (j < n)
  - For (each subset as S of R with cardinality j)
    Find the subset as $S_j$ with cardinality j from All_subset which minimize the cost function (linear work metric);
  - End for
  - Construct the optimal tree for $S_j$ using optimal tree for $(S_1),(S_2),\ldots,(S_m)$; (Note that the optimal tree for $S_m$ $(1 \le m \le j-1)$ is already obtained)
  - Compute $|S_j|$, $|S_j'|$ and $|\Delta S_j|$ from $|S_i|$, $|S_i'|$ and $|\Delta S_i|$ where $(1 \le i < j)$;
  - Append $|\Delta S_i|$ to "change message queue";
  - j = j+1;
- End while
//Function innovation with message queue argument
  - Minimum calculation maintenance(change message queue);

End

شکل ۳- الگوریتم ساخت درخت نگهداری دید

روند کار در الگوریتم ارائه شده در شکل (۳) بدین گونه است که، برای دید ذخیره شده‌ای همانند $V=R_1 \infty R_2 \infty \ldots \infty R_n$ با n منبع داده پایه ابتدا همه زیر مجموعه‌هایی از مجموعه منابع داده پایه تشکیل می‌گردد. در واقع در مرحله اول از الگوریتم پیشنهادی برگ‌های درخت زیر مجموعه با منابع داده پایه مقداردهی می‌گردد(تعداد برگ‌های درخت برابر با تعداد منابع داده پایه دید خواهد بود). مرحله دوم ساخت گره‌های میانی درخت زیر مجموعه می‌باشد که این کار توسط دو حلقه تو در تو انجام می‌شود. حلقه اول برای ساخت گره‌هایی از درجه j

می‌باشد(j=2,3, … ,n). حلقه تکرار دوم بر روی کل زیرمجموعه‌های تشکیل شده در مرحله اول حرکت می‌نماید و به دنبال زیرمجموعه یا گره‌ای از درجه j می‌باشد که دارای کمترین مقدار زمان و هزینه نگهداری باشد. به همین ترتیب مراحل کار ادامه می‌یابد تا به گره‌ای از درجه j=n برسیم که در واقع همان ریشه درخت یا دید ذخیره شده مورد نظر می‌باشد. لازم به ذکر است که در هنگام ساخت گره‌های درخت در هر مرحله‌ای مانند j از گره‌های ساخته شده در مرحله m استفاده می‌گردد(2≤m≤i-1).

در الگوریتم پیشنهادی هنگام ساخت درخت، اندازه تغییرات گره‌های میانی در یک صف به نام صف تغییرات[6] قرار می‌گیرد. پس از ساخت درخت و تشکیل صف تغییرات، بخش دوم از الگوریتم برای تولید عبارت نگهداری دید بهبود یافته فراخوانی می‌گردد. در بخش دوم از الگوریتم پیشنهادی بر اساس صف تغییرات و ساختار درخت، دنباله‌ای مناسب جهت اعمال تغییرات منابع داده (منابع داده پایه و گره‌های میانی) بر روی دید نهایی ارائه می‌گردد.

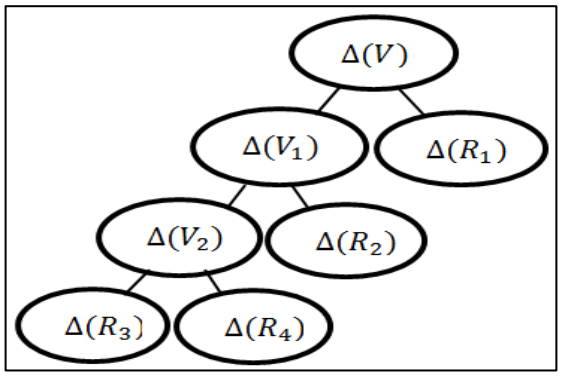
### ۳-۲- تحلیل الگوریتم پیشنهادی

دید $V=R_1 \infty R_2 \infty \dots \infty R_n$ با n منبع داده پایه را در نظر بگیرید. برای دید ذخیره شده‌ای با n منبع داده پایه، الگوریتم پیشنهادی برای تشکیل گره‌های میانی درخت بر روی همه‌ی زیرمجموعه‌ها از مجموعه منابع داده پایه دید عمل جستجو را انجام می‌دهد. تعداد حالات ممکن برای ساخت زیرمجموعه‌های یک مجموعه n عضوی برابر با $2^n$ می‌باشد. در هر مرحله از الگوریتم پیشنهادی برای ساخت یک گره میانی می‌بایست از طریق جستجو بر روی همه زیرمجموعه‌ها به دنبال گره یا زیرمجموعه‌ای باشیم که دارای کمترین میزان هزینه و زمان نگهداری باشد. مرتبه زمانی برای تشکیل یک گره در درخت زیرمجموعه برابر با $n * 2^n$ می‌باشد. بنابراین برای ساخت همه گره‌های میانی درخت زیرمجموعه‌ی متناظر با دید ذخیره شده‌ای که شامل n منبع داده پایه می‌باشد، به تعداد n بار عمل جستجو بر روی همه زیرمجموعه‌ها انجام می‌گردد. بنابراین مرتبه زمانی برای ساخت درخت زیرمجموعه برابر با $n^2 * 2^n$ می‌باشد. روش ارائه شده در روش [۱۲] یکی از روش‌های معروف در زمینه کاهش دسترس به منابع داده بزرگ می‌باشد. روش ارائه شده در [۱۲] برای دیدی با n منبع داده پایه از الگوریتمی با مرتبه زمانی $n^n * 2^n$ استفاده می‌نماید. این در حالی است که الگوریتم ارائه شده در شکل (3) برای کاهش دسترسی به منابع داده بزرگ دارای مرتبه زمانی $n^2 * 2^n$ می‌باشد که عملکرد و کارایی مناسبی نسبت به الگوریتم ارائه شده در [۱۲] را دارد.

### ۳-۳- ساخت عبارت نگهداری دید بهبود یافته

در این بخش از الگوریتم پیشنهادی، دنباله‌ی ترتیبی مناسبی بر اساس اندازه تغییرات منابع داده‌ای، برای نگهداری از دیدهای ذخیره شده در پایگاه داده تحلیلی ارائه می‌گردد. در عبارت نگهداری دید بهبود یافته، منابعی که دارای اندازه تغییرات کوچکتری هستند در اولویت قرار می‌گیرند و منابعی که اندازه تغییرات بزرگتری دارند در مراحل بعدی عبارت نگهداری دید ظاهر می‌شوند. لازم به ذکر است که عملیات بروز رسانی بوسیله عملگرهای درج و حذف انجام می‌شود. در این قسمت از مقاله منظور از منابع داده‌ای گره‌های میانی درخت و منابع داده پایه دید می‌باشد.

گره‌های میانی درخت زیرمجموعه‌ی ساخته شده در شکل (۱) $\Delta(R_3 \infty R_4)$ و $\Delta(R_2 \infty R_3 \infty R_4)$ با $V_2$ و $V_1$ نشان داده می‌شود. با توجه به تغییرات ذکر شده در شکل (۱)، درخت ساخته شده به صورت شکل (۴) ارائه می‌گردد.




شکل ۴- مثالی از درخت نگهداری برای دیدی با چهار منبع داده پایه

برای محاسبه و انتشار تغییرات منابع داده‌ای می‌توان از دنباله‌های متفاوتی استفاده نمود. دنباله اول می‌تواند به صورت رابطه (۷) باشد.

$$\Delta R_1 \rightarrow \Delta V,\ \Delta R_3 \rightarrow \Delta V_2,\ \Delta R_4 \rightarrow \Delta V_2,\ \Delta R_2 \rightarrow \Delta V_1,\ \Delta V_2 \rightarrow \Delta V_1, \Delta V_1 \rightarrow V \quad (۷)$$

در رابطه (۷) عبارت $\Delta R_i \rightarrow V$ بیانگر انتشار تغییرات منبع داده‌ی $R_i$ بر روی دید $V$ می‌باشد. علاوه بر این برای

[6] Change Message Queue

درخت ارائه شده در شکل (۴) می‌توان دنباله دیگری همانند رابطه (۸) را ارائه کرد.

$$\Delta R_3 \to \Delta V_2,\ \Delta R_4 \to \Delta V_2,\ \Delta V_2 \to \Delta V_1,\ \Delta R_2 \to \Delta V_1,\ \Delta V_1 \to \Delta V,\ \Delta R_1 \to \Delta V \quad (۸)$$

همانطور که گفته شد، برای یک درخت زیرمجموعه دنباله-های متفاوتی جهت اعمال تغییرات منابع داده‌ای بر روی دید ذخیره شده وجود دارد. در درخت ارائه شده در شکل (۴) اگر اندازه تغییرات منابع داده‌ای به صورت رابطه (۹) باشد، در این صورت برای کاهش زمان نگهداری دنباله‌ای به صورت رابطه (۱۰) ارائه می‌گردد.

$$\left|R_4'\right|-|R_4| < \left|R_3'\right|-|R_3| < \left|R_2'\right|-|R_2| < \left|V_2'\right|-|V_2| < \left|R_1'\right|-|R_1| < \left|V_1'\right|-|V_1|. \quad (۹)$$

$$\Delta R_4 \to \Delta V_2,\ \Delta R_3 \to \Delta V_2,\ \Delta R_2 \to \Delta V_1,\ \Delta V_2 \to \Delta V_1 \quad (۱۰)$$

مثال (۱) نیز جهت وضوح اولویت‌بندی انتشار تغییرات منابع داده‌ی بر روی عبارت نگهداری دید ارائه می‌گردد. مثال ۱) رابطه (۴)، عبارت نگهداری یک دید با چهار منبع داده پایه می‌باشد. اگر اندازه تغییرات منابع داده‌ای به صورت رابطه (۹) باشد، در این حالت برای کاهش زمان و هزینه نگهداری بهتر است که به ترتیب تغییرات $R_1, V_2, R_2, R_3, R_4$ و $V_1$ بر روی دید ذخیره شده انتشار یابد. در نتیجه‌ی اعمال به ترتیب تغییرات منابع داده‌ای، ترتیب روابط دلتا بر روی عبارت نگهداری متفاوت خواهد بود که منجر به تولید عبارت نگهداری به صورت رابطه (۱۱) می‌گردد.

$$\begin{aligned}
&\Delta(V) = (\Delta R_1 \infty R_2 \infty R_3 \infty R_4) \cup \left(R_1' \infty \Delta(R_2 \infty R_3 \infty R_4)\right)\\
&\Delta(R_2 \infty R_3 \infty R_4) = (\Delta(R_2) \infty R_3 \infty R_4) \cup \left(R_2' \infty \Delta(R_3 \infty R_4)\right)\\
&\Delta(R_3 \infty R_4) = (\Delta(R_4) \infty R_3) \cup \left(R_4' \infty \Delta(R_3)\right)
\end{aligned} \quad (۱۱)$$

در الگوریتم شکل (۳) درختی برای کاهش میزان دسترسی به منابع داده‌ی بزرگ ایجاد گردید. برای درخت ارائه شده بر اساس بزرگی اندازه تغییرات منابع داده‌ای، عبارت نگهداری دید بهبود یافته‌ای ارائه می‌گردد. اندازه تغییرات منابع داده‌ای در داخل صفی به نام صف تغییرات قرار دارد(تغییرات گره‌های میانی در هنگام ساخت درخت محاسبه شده است). صف تغییرات بر اساس بزرگی اندازه تغییرات منابع داده(منابع داده پایه و گره‌های میانی) به صورت صعودی مرتب می‌گردد. بنابراین با توجه به ساختار درخت و ترتیب منابع در صف تغییرات، دنباله‌ای مناسب(عبارت نگهداری دید بهبود یافته) برای نگهداری از دیدها ارائه می‌گردد. علاوه بر این، دنباله‌ی تولید شده برای نگهداری از دید باید بدون چرخه باشد. به عبارتی در دنباله ارائه شده، تغییرات یک منبع داده‌ای خاص نباید بیش از یک بار محاسبه گردد. بر این اساس، در شکل (۵) الگوریتم تولید عبارت نگهداری دید بهبود یافته ارائه می‌گردد.

```
Minimum calculation maintenance
Input:
        Change Message Queue, materialized view V, base relations {R1, R2, R3... Rn}.
Output:
        Incremental view maintenance with Minimum tuple calculation of V.
Begin
        For all node of delta tree calculate the change with ΔRi = |Ri' |- |Ri | equation;
        Append ΔRi to Change Message Queue;
        Ascending sort of "Change Message Queue";
        According to Change message queue and level of delta tree construct the subsequence of data
        source change;
        Calculate the view V with (V= V U ΔV)
        Return (V);
End
```

شکل۵- الگوریتم تولید عبارت نگهداری دید بهبود یافته

## ۴- آزمایش‌ها

در این بخش عبارت نگهداری دید بهبود یافته را با الگوریتم ارائه شده در [۱۲] مورد مقایسه قرار می‌دهیم. در این بخش ابتدا به معرفی تابع هزینه می‌پردازیم سپس

آزمایشات انجام شده برای مقایسه الگوریتم ها ارائه می-گردد.

### ۴-۱- تابع هزینه

در این مقاله برای مقایسه عبارت‌های نگهداری دید از تابع هزینه خطی[۷] استفاده می‌گردد[۱۳]. در تابع هزینه خطی برای بدست آوردن هزینه یک عبارت نگهداری دید، ابتدا هزینه هرکدام از جملات موجود در عبارت نگهداری دید را به صورت مستقل محاسبه کرده و سپس هزینه همه واحدها با هم جمع می‌گردد. تابع هزینه خطی را بر روی عبارت نگهداری دید رابطه (۲) اعمال می‌نماییم. رابطه (۵) بیانگر نتیجه اعمال تابع هزینه خطی بر روی رابطه (۲) می‌باشد.

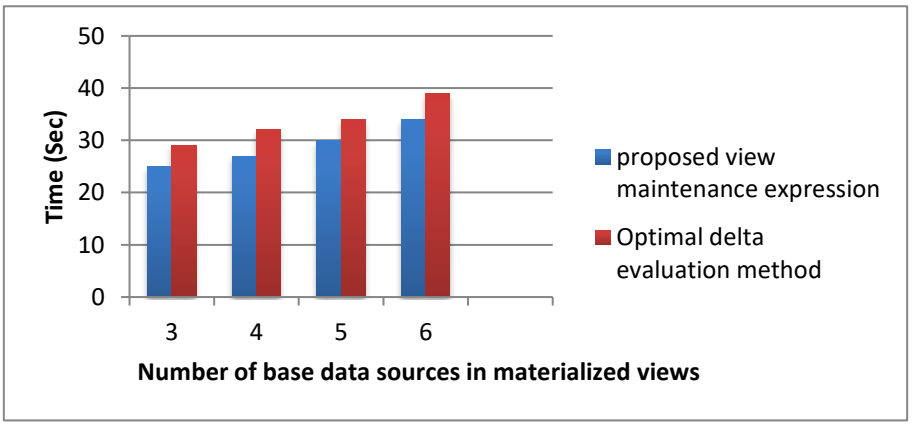
$$\begin{aligned} Cost(\Delta V) &= Cost\left((\Delta R_1 \infty R_2 \infty R_3 \infty R_4)\right) \\ &+ Cost\left((R_1' \infty \Delta R_2 \infty R_3 \infty R_4\right) \\ &+ Cost\left((R_1' \infty R_2' \infty \Delta R_3 \infty R_4\right) \\ &+ Cost\left((R_1' \infty R_2' \infty R_3' \infty \Delta R_4\right) \\ &= c.\left(|\Delta R_1| + |R_2| + |R_3| + |R_4|\right) \\ &+ c.\left(|R_1'| + |\Delta R_2| + |R_3| + |R_4|\right) \\ &+ c.\left(|R_1'| + |R_2'| + |\Delta R_3| + |R_4|\right) \\ &+ c.\left(|R_1'| + |R_2'| + |R_3'| + |\Delta R_4|\right) \end{aligned} \quad (۵)$$

در رابطه (۵) مقدار $c$ یک مقدار ثابت است و $|R_1|$ بیانگر اندازه منبع داده $R_1$ می‌باشد. در این تابع هزینه با توجه به اینکه مجموع هزینه جملات در هر عبارت نگهداری دید لحاظ می‌گردد، بنابراین کاهش جملات و کاهش منابع داده‌ای با حجم بالا تاثیر چشمگیری در کاهش هزینه عبارت نگهداری دید خواهد داشت. بهبود عبارت نگهداری دید (کاهش دسترسی به منابع داده بزرگ و ارائه دنباله مناسب جهت اعمال تغییرات) نقش اساسی در کاهش هزینه خواهد داشت.

### ۴-۲- بررسی کارایی الگوریتم‌ها

برای انجام آزمایشات و مقایسه الگوریتم‌ها با یکدیگر، از ابزارهای Microsoft SQL Server 2008 و Visual studio 2010 استفاده شده است. این ابزارها بر روی سیستمی با حافظه اصلی 4G اجرا گردیده است. محیط آزمایشی مورد استفاده، بر پایه منابع داده‌ای تولیدکنندگان خودرو می‌باشد. منبع داده‌ای CarFactor توصیف کننده میزان فروش خودروها در شعبه‌های مختلف می‌باشد.

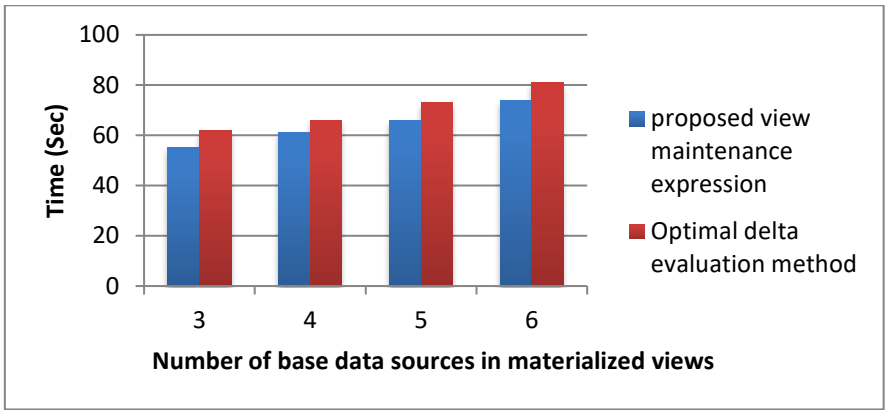
در کارهای مرتبط انجام شده [۱۱-۱۳] برای ارزیابی عبارت نگهداری دید ارائه شده، از دیدهایی با تعداد منابع داده متفاوت استفاده می‌شود. بنابراین در این مقاله نیز برای انجام آزمایش‌ها از چهار دید با تعداد منابع داده متفاوت استفاده می‌گردد. دیدهای ذکر شده را با $V_1$، $V_2$، $V_3$ و $V_4$ نشان می‌دهیم و به صورت روابط زیر تعریف می‌شود. در آزمایش‌ها، الگوریتم ارائه شده (عبارت نگهداری دید بهبود یافته) و الگوریتم ارزیابی دلتای بهینه [۱۲] بر روی دیدهای ذکر شده اجرا گردید. در آزمایش اول هر کدام از روش‌های نگهداری دید، بر روی $V_1$ با سه منبع داده پایه اعمال شده و زمان لازم برای نگهداری از $V_1$ مورد مقایسه قرار گرفت. در آزمایش‌های بعدی روشهای نگهداری دید ذکر شده را بر روی دیدهای $V_2$، $V_3$ و $V_4$ به ترتیب با ۴، ۵ و ۶ منبع داده پایه اعمال گردید. نتایج حاصل از مقایسه دو روش در شکل (۶) ارائه می‌شود. در شکل (۶) محور افقی تعداد منابع داده پایه برای هر کدام از دیدها را نشان می‌دهد و محور عمودی زمان لازم برای نگهداری از دیدها را نشان می‌دهد.




شکل ۶- مقایسه‌ی عبارت نگهداری دید بهبود یافته با درخت دلتای بهینه بر اساس دیدهایی با منابع داده‌ای متفاوت

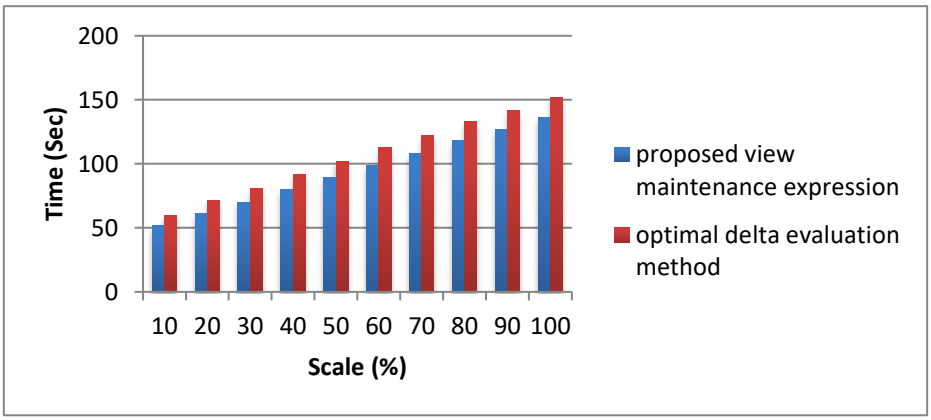
[۷] Linear Work Metric

ساختار تابع هزینه خطی به گونه‌ای است که اندازه منابع داده پایه تاثیر زیادی در عملکرد عبارت نگهداری دید بهبود یافته دارد. بنابراین در آزمایش دیگری داده‌های جدیدی به منابع داده پایه وارد گردید، به طوری که اندازه منابع داده پایه ۳۰ درصد افزایش داده شد و دوباره برای هر کدام از دیدهای $V_1$، $V_2$، $V_3$ و $V_4$ عبارت نگهداری دید بهبود یافته و روش ارزیابی دلتای بهینه [۱۲] اعمال گردید. شکل (۷) نشان می‌دهد که با وجود افزایش ۳۰ درصدی داده‌های منابع داده پایه، عملکرد عبارت نگهداری دید ارائه شده مطلوب‌تر از روش ارائه شده در [۱۲] می‌باشد.

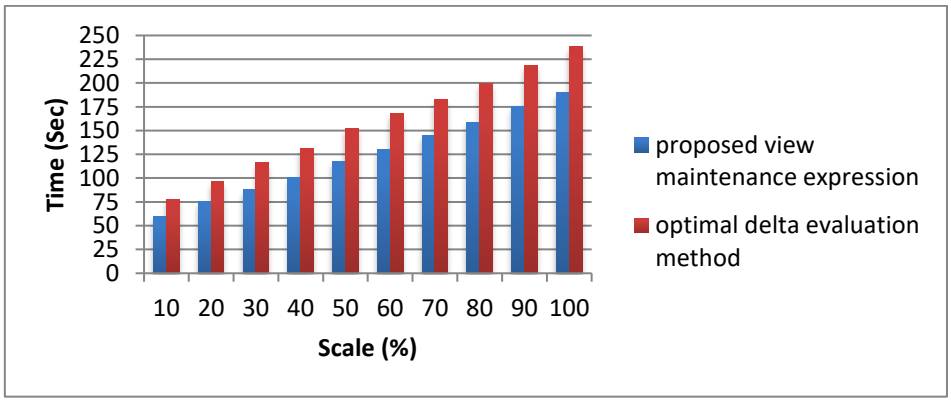



شکل ۷- افزایش ۳۰ درصدی داده‌های منابع داده پایه و مقایسه عبارت نگهداری دید بهبود یافته با درخت دلتای بهینه بر اساس دیدهایی با منابع داده‌ای متفاوت

در آزمایش دیگری جهت بررسی کارایی روش ارائه شده، اندازه تغییرات منابع داده پایه افزایش داده شد. در این آزمایش تغییرات منابع داده پایه از ۱۰ درصد تا ۱۰۰ درصد افزایش می‌یابد و کارایی روش ارائه شده در مقایسه با روش [۱۲] مورد ارزیابی قرار می‌گیرد. این آزمایش را بر روی دیدهای $V_3$ و $V_4$ انجام شده است که نتایج این آزمایش در شکل (۸) و (۹) ارائه شده است.




شکل ۸- افزایش اندازه‌ی تغییرات منابع داده دید $V_3$ و مقایسه عبارت نگهداری دید بهبود یافته با درخت دلتای بهینه




شکل ۹- افزایش اندازه‌ی تغییرات منابع داده دید $V_4$ و مقایسه عبارت نگهداری دید بهبود یافته با درخت دلتای بهینه

## ۵- نتیجه‌گیری

پایگاه داده تحلیلی برای آنالیز و کمک به تصمیم‌گیری در سطوح مختلف مدیریتی کاربرد دارد. بروز بودن داده‌ها در پایگاه داده تحلیل نقش مهمی در پاسخگویی درست و معتبر به پرس و جوهای تحلیلی و کمک به مدیران در تصمیم گیری‌ها خواهد داشت. منابع داده پایه مدام در حال تغییر و بروز رسانی هستند. به همین دلیل دیدهای ذخیره شده در پایگاه داده تحلیلی برای پاسخگویی مناسب به پرس و جوهای تحلیلی کاربران باید بروز رسانی و نگهداری گردند. با توجه به حجم بالای داده‌ها در پایگاه داده تحلیلی، بروز رسانی و نگهداری دیدهای ذخیره شده مستلزم زمان و هزینه زیادی می‌باشد. در واقع کارایی سیستم پردازش تحلیلی برخط[8] در ارائه نتایج درست تا حد زیادی به استفاده از روش و الگوریتم مناسب برای بروز کردن داده‌های پایگاه داده تحلیلی وابسته می‌باشد. در این مقاله عبارت نگهداری دید بهبود یافته‌ای برای نگهداری از دیدها در پایگاه داده تحلیلی ارائه شده است.

عبارت نگهداری دید ارائه شده از دو دیدگاه منجر به بهبود عملکرد نگهداری از پایگاه داده تحلیلی شده است. بدین صورت که در دیدگاه اول با استفاده از درخت زیر مجموعه، تعداد دسترسی به منابع داده بزرگ(منابعی با تعداد رکورد و حجم بالا) کاهش داده می‌شود، و در دیدگاه دوم با ارائه دنباله‌ای مناسب بر اساس اندازه تغییرات منابع داده(منابع داده پایه و گره‌های میانی درخت) عبارت نگهداری دید مناسب ارائه می‌گردد. آزمایش‌ها نشان می‌دهد که عبارت نگهداری دید اراده شده عملکرد مناسبی نسبت به عبارتهای نگهداری دید پیشین داشته است.

## مراجع


[1] Y.-S. Huang, D. Duy, and C.-C. Fang. (2014). "Efficient maintenance of basic statistical functions in data warehouses," *Decision Support Systems*, vol. 57, pp. 94-104.

[2] X. Huang and Q. Chen. (2011). "A maintainable model of materialized view based on data warehouse," in *Mechatronic Science, Electric Engineering and Computer (MEC), 2011 International Conference on*, pp. 1974-1977.

[3] A. S. Almazyad and M. K. Siddiqui. (2010). "Incremental view maintenance: an algorithmic approach," *International Journal of Electrical & Computer Sciences IJECSIJENS*, vol. 10.

[4] X. Zhang, L. Yang, and D. Wang. (2010). "Incremental view maintenance based on data source compensation in data warehouses," in *Computer Application and System Modeling (ICCASM), 2010 International Conference on*, pp. V2-287-V2-291.

[5] A. S. Almazyad, M. K. Siddiqui, Y. Ahmad, and Z. I. Khan. (2009). "An Incremental View Maintenance Approach Using Version Store in Warehousing Environment," in *Database Theory and Application*, ed: Springer, pp. 9-16.

[6] H. Gupta. (2009). "Incremental Maintenance of Views with Aggregates," in *Encyclopedia of Database Systems*, ed: Springer, 2009, pp. 1421-1425.

[7] T. Palpanas, R. Sidle, R. Cochrane, and H. Pirahesh.(2002). "Incremental maintenance for non-distributive aggregate functions," in *Proceedings of the 28th international conference on Very Large Data Bases*, pp. 802-813.

[8] H. Fan. (2005). "Using schema transformation pathways for incremental view maintenance," in *Data Warehousing and Knowledge Discovery*, ed: Springer, pp .126-135


[8] OnLine Analytical Process

[9] A. Nica.(2012). "Incremental maintenance of materialized views with outerjoins," *Information Systems,* vol. 37, pp. 430-442.

[10] H. Gupta and I. S. Mumick. (2006). "Incremental maintenance of aggregate and outerjoin expressions," *Information Systems* ,vol. 31, pp. 435-464.

[11] L. Zhou, Q. Shi, and H. Geng.(2010). "The minimum incremental maintenance of materialized views in data warehouse," in *Informatics in Control, Automation and Robotics (CAR), 2010 2nd International Asia Conference on*, pp. 22.۲۲۳-۰

[12] K. Y. Lee, J. H. Son, and M. H. Kim. (2007). "Reducing the cost of accessing relations in incremental view maintenance," *Decision support systems,* vol. 43, pp. 512-526.

[13] W. J. Labio, R. Yerneni, and H. Garcia-Molina. (1999). "Shrinking the warehouse update window," in *ACM SIGMOD Record*, pp. 383-394.

[14] T. Griffin and L. Libkin. (1995). "Incremental maintenance of views with duplicates," in *ACM SIGMOD Record*, pp. 328-339.

[15] A. Gupta, I. S. Mumick, and V. S. Subrahmanian. (1993). "Maintaining views incrementally," *ACM SIGMOD Record*, vol. 22, pp. 157-166.

[16] B. Niamir. (1978). "Attribute Partitioning in a Self-Adaptive Relational Data Base System," 1978

[17] Garcia, C. 2006. Real time self-maintenable data warehouse. *in Proceedings of the 44th annual Southeast regional conference*. pp. 518-524.

[18] Mohania, M., and Y. Kambayashi. 2000. Making aggregate views self-maintainable. *Data & Knowledge Engineering,* vol. 32. pp. 87-109.

[19] Samtani, S., V. Kumar, and M. Mohania. 1999. Self maintenance of multiple views in data warehousing. *in Proceedings of the eighth international conference on Information and knowledge management*. pp. 292-299.