کاوش الگوهای متناوب از درخت FP با کمک ماتریس دوبعدی

بابک توسلی[۱]؛ مهرداد جلالی[۲]؛ مجید وفایی جهان[۳]

چکیده

کشف الگوهای پنهان و ارزشمند از درون حجم وسیعی از داده‌های خام، اخیراً توجه بسیاری از محققان را به خود جلب کرده‌است. اولین گام برای استخراج روابط بین اقلام موجود در داده‌ها، مشخص نمودن تعداد رخدادهای همزمان اقلام با یکدیگر است. الگوریتم‌های متعددی برای نیل به این هدف تاکنون ارائه شده‌اند اما می‌توان از نظر نحوه عملکرد آن‌ها را در دو دسته کلی قرار داد. در دسته نخست کاندیدهایی از مجموعه قلم‌داده تولید می‌شوند و سپس با پویش تراکنش‌ها، تکرار هر کدام به دست می‌آید و آنهایی که تکرارشان بیشتر از حدآستانه‌پشتیبانی‌کمینه تعیین شده باشد به عنوان الگوی متناوب معرفی می‌شوند. مشکل اصلی این روش هزینه بالای تولید کاندیدها و پویش‌های متعدد تراکنش‌هاست. در دسته دوم الگوریتم‌های رشد الگو قرار می‌گیرند که برای استخراج الگوهای متناوب نیازی به تولید کاندید ندارند. کارایی این الگوریتم‌ها در پایگاه‌های داده بزرگ، جایی که الگوریتم‌های دسته اول به خاطر تعداد زیاد کاندیدها بسیار ناکارآمد عمل می‌کنند، بسیار محسوس است.

در الگوریتم‌های رشد الگو، از ساختار داده‌ای به نام درخت الگوی متناوب استفاده میشود. با بررسی روش رشدالگوی‌متناوب مشخص گردیده‌است که حدود ۸۰٪ از زمان اجرای الگوریتم صرف پیمایش درخت الگوی متناوب می‌شود. از آنجاییکه پیمایش لیست پیوندی فرآیندی زمانبر است لذا در روش پیشنهادی از ساختارداده آرایه برای نگهداری درخت‌الگوی‌متناوب استفاده شده‌است تا سرعت پیمایش افزایش‌یابد. همچنین برای عملیات کاوش نیز دیگر نیازی به ساختن درخت‌های شرطی نیست و بر خلاف الگوریتم‌های فعلی که در هر بار پیمایش شاخه‌های درخت، الگوی جاری به طول n را تا $n+1$ گسترش می‌دهند، در روش پیشنهادی با کمک ماتریس دوبعدی، الگوی متناوب جاری به طول n تا حداکثر $n+2$ گسترش می‌یابد. نتایج تجربی نشان می‌دهد که تغییر ساختار نگهداری درخت و روش کاوش پیشنهادی در پایگاه‌های تراکنش متوسط تا بزرگ باعث بهبود در سرعت اجرای الگوریتم به خصوص در آستانه‌های کوچکتر می‌شود.

کلمات کلیدی

داده کاوی، مجموعه قلم‌داده، رشد الگو، درخت الگوی متناوب، ماتریس دوبعدی

[۱] دانشجوی کارشناسی ارشد کامپیوتر- نرم افزار ، دانشگاه آزاد اسلامی واحد مشهد، babak.tavassoli@gmail.com
[۲] استادیار گروه کامپیوتر دانشکده فنی مهندسی دانشگاه آزاد اسلامی واحد مشهد، mehrjalali@gmail.com
[۳] استادیار گروه کامپیوتر دانشکده فنی مهندسی دانشگاه آزاد اسلامی واحد مشهد، vafaeija@yahoo.com

Mining Frequent Patterns from FP-Tree By Using ۲D Matrix

Babak Tavassoli; Mehrdad Jalali; Majid Vafaei Jahan

Abstract

Discovering hidden pattern from large volume of raw data has recently attracted the attention of many researchers. The first step to extract relations among data items is specifying the number of simultaneous occurrence of items together. Several algorithms have been proposed to achieve this goal so far but they can be placed in two major categories based on their functionality. In the first category candidates generated from data items and their number of occurrence can be calculated after the first scan of database and those with frequency equal or greater than the minimum support are marked as frequent pattern. The main bottleneck of this method is high cost of generating candidates and multiple scans of database. In the second category there are pattern growth algorithms that for mining frequent patterns are not required to generate candidates. Performance of these algorithms in large databases where the first category algorithms are very inefficient because of large number of candidates is very noticeable. Pattern growth algorithms use a data structure called frequent pattern tree or FP-Tree. From numerous experiments, we found out that about ۸۰ percent of the CPU time was used for traversing FP-trees. Since linked list traversal is a time consuming process in the proposed algorithm we have used array structure for storing transactions in order to increase the speed. Constructing conditional fp-trees is no longer needed in the proposed method and unlike current pattern-growth algorithms which extend a frequent pattern with length n to n+۱ in each step of mining, in the proposed method each frequent pattern with length n can be extended to maximum n+۲. Experimental results show that changing the structure for constructing tree and new method of mining frequent patterns can improve speed of the algorithm for medium to large data bases especially in the smaller thresholds.

Keywords:

Data Mining; Itemset; Pattern Growth; Frequent Pattern Tree; ۲D Matrix

۱. مقدمه

الگوهای‌متناوب[۱] مجموعه قلم‌داده‌ها[۲]، توالی‌ها[۳] و یا ساختارهایی[۴] هستند که در یک مجموعه‌داده به تعداد دفعاتی بیشتر یا مساوی با آنچه توسط کاربر به عنوان حد آستانه تعیین شده است، تکرار شده باشند. به عنوان مثال مجموعه‌ای از چند قلم‌داده (مانند شیر و نان) که معمولا با هم در تراکنش‌های موجود در پایگاه تراکنش یک فروشگاه به کار می‌روند، یک مجموعه‌قلم‌داده‌متناوب[۵] هستند. در یک توالی علاوه بر قلم‌داده‌ها، ترتیب زمانی به کاررفتن و یا روی‌دادن آنها در یک تراکنش مهم است. به عنوان مثال خرید یک کامپیوتر و پس از آن میز کامپیوتر و سپس سی‌دی‌های بازی، یک توالی در پایگاه تراکنش فروشگاه محسوب می‌شود. یک ساختار می‌تواند اشکال مختلف ساختارها از قبیل گراف، درخت و شبکه را در برگیرد. اگر ساختاری در یک پایگاه گرافیکی بیش از حد مشخصی تکرار شود آن را الگوی ساختاری متناوب[۶] می‌نامند. یافتن الگوی متناوب، نقشی کلیدی در کاوش قوانین وابستگی، همبستگی‌ها و بسیاری از روابط جالب دیگر در بین داده‌ها ایفا می‌نماید. علاوه بر آن در اندیس نمودن داده‌ها، طبقه بندی، خوشه بندی و سایر عملیات داده‌کاوی نیز کارآیی دارد.

۱,۱ قوانین وابستگی

با درنظر گرفتن یک مجموعه از قلم‌داده‌ها و یک پایگاه‌تراکنش متشکل از این مجموعه، با کمک قوانین‌وابستگی می‌توان وابستگی میان قلم‌داده‌ها را پیدانمود. این وابستگی‌ها بیان می‌کنند که حضور هر قلم‌داده و یا مجموعه‌قلم‌داده‌ها در یک تراکنش موجب حضور برخی دیگر از قلم‌داده‌ها و یا مجموعه‌قلم‌داده‌ها می‌شود. مثال متداول در رابطه با کشف قوانین وابستگی، «تحلیل سبد خرید» است[۱]. در این فرآیند با توجه به اقلام مختلفی که مشتریان در سبد خریدشان قرار می‌دهند، عادات و رفتار خرید آن‌ها مورد تحلیل قرار می‌گیرد. هدف در این فرآیند پیداکردن خودکار قوانینی مثل «۶۰٪ افرادی که نان خریداری می‌کنند شیر هم می‌خرند» می‌باشد که البته برای قابل قبول بودن قوانین معیارهایی نیز وجود

دارد. به طور کلی هدف اصلی در کاوش قوانین وابستگی پیداکردن تمامی قوانینی است که آستانه‌پشتیبان‌کمینه[۷] و ضریب‌اطمینان‌کمینه[۸] مدنظر کاربر را رعایت کرده باشند. مسئله کشف قوانین وابستگی را می‌توان به دو زیر مساله تقسیم کرد[۲]:

۱.پیدا کردن تمام مجموعه قلم داده‌هایی که حداقل آستانه پشتیبان را داشته باشند. به مجموعه قلم داده‌هایی که حداقل آستانه پشتیبان را داشته باشند مجموعه قلم داده‌های متناوب گفته‌می‌شود و مجموعه قلم داده‌هایی که این حداقل آستانه را نداشته باشند، مجموعه قلم داده‌های غیرمتناوب نامیده می‌شوند.
۲.استفاده از مجموعه قلم داده‌های متناوب جهت تولید قوانین وابستگی.

از آنجایی که زیر مساله اول؛ یعنی کاوش مجموعه قلم‌داده‌های متناوب مساله زمانبرتری‌است؛ لذا در سال‌های اخیر بررسی‌ها و پژوهش‌های بیشتری در این زمینه صورت‌گرفته‌است.

۲,۱ کاوش قوانین وابستگی[۹]

کاوش قوانین وابستگی یکی از مهمترین تکنیک‌های داده‌کاوی است که هدف اصلی آن استخراج ارتباطات پنهان و وابستگی‌های جالب بین قلم‌داده‌های موجود در یک پایگاه‌تراکنش است. عملیات یافتن قوانین وابستگی را کاوش قوانین وابستگی گویند. این عملیات شامل دو مرحله اساسی‌است: مرحله اول کاوش مجموعه قلم‌داده‌های متناوب و مرحله دوم تولید قوانین وابستگی با استفاده از مجوعه قلم‌داده‌های متناوب. بار اصلی این عملیات بر دوش مرحله اول یعنی پیداکردن مجموعه قلم‌داده‌های متناوب است. کاوش مجموعه قلم‌داده‌های متناوب نخستین بار در سال ۱۹۹۳ توسط آگراوال[۱۰] برای تحلیل سبد خرید و در قالب کاوش قوانین وابستگی ارائه شد[۳]. نتیجه چنین تحلیل‌هایی نشان می‌داد مشتریانی که کالای خاصی را می‌خرند به همراه این کالا به خرید چه نوع کالاهای دیگری رغبت نشان می‌دهند. بر اساس این نتایج صاحبان فروشگاه‌ها می‌توانند نحوه چینش کالاها در قفسه‌ها را بر اساس نیاز مشتریان تنظیم نمایند.

۳,۱ روش های یافتن مجموعه قلم داده‌های متناوب

تاکنون روش‌های متعددی برای یافتن مجموعه قلم‌داده‌های متناوب پیشنهاد شده‌اند که به در دو دسته کلی طبقه بندی میشوند؛ روش‌های مبتنی بر تولید کاندید[۱۱] و روش‌های مبتنی بر رشد الگو[۱۲]. برای نخستین بار آگراوال در سال ۱۹۹۴ خاصیت جالبی به نام Apriori را در مجموعه آیتم‌های متناوب مشاهده نمود[۴]. بر اساس این خاصیت، «یک مجموعه k- قلم‌داده‌ای فقط وقتی متناوب است که همه زیر مجموعه قلم‌داده‌های آن متناوب باشند». نتیجه اصلی این خاصیت، «متناوب نبودن ابرالگوهای یک الگوی نامتناوب» است که به الگوریتم کاوش امکان کنار گذاشتن مجموعه‌های k- قلم‌داده‌ای نامتناوب در کاوش مجموعه‌های (k+۱)- قلم‌داده‌ای را خواهد داد. بر این اساس برای یافتن مجموعه کامل همه مجموعه قلم‌داده‌های متناوب موجود در یک‌پایگاه تراکنش، باید ابتدا با یکبار پویش [۱۳]پایگاه تراکنش، همه مجموعه‌های ۱- قلم‌داده‌ای متناوب (قلم‌داده‌های متناوب) موجود در آن پیدا شده و سپس تمامی مجموعه‌های متناوب ۲- قلم‌داده‌ای قابل ساخت از این قلم‌داده‌های متناوب به عنوان کاندیدای متناوب بودن تولید شوند. هر یک از این مجموعه‌های ۲- قلم‌داده‌ای، چون تنها از قلم‌داده‌های متناوب‌تشکیل شده‌اند (همه زیرمجموعه قلم‌داده‌های آنها متناوب هستند)، به صورت بالقوه امکان متناوب بودن در پایگاه تراکنش را دارند. برای اطمینان از این تناوب، لازم است بار دیگر پایگاه تراکنش را پویش نموده و تناوب یا عدم‌تناوب هریک از کاندیداهای ۲- قلم‌داده‌ای مشخص شود. پس از تعیین مجموعه‌های متناوب ۲- قلم‌داده‌ای، به صورت مشابه از آنها برای ساخت کاندیداهای ۳- قلم‌داده‌ای و بررسی تناوب یا عدم تناوب آنها با پویش مجدد پایگاه‌تراکنش استفاده‌می‌شود. روند استفاده از مجموعه‌های متناوب k-قلم‌داده‌ای در تولید کاندیداهای (k+۱)-قلم‌داده‌ای و بررسی تناوب یا عدم تناوب آنها با یک پویش کامل پایگاه‌تراکنش، تا استخراج همه مجموعه قلم‌داده‌های متناوب موجود درپایگاه‌تراکنش ادامه می‌یابد. اگرچه در بسیاری از مسائل و نمونه‌ها، روش‌های مبتنی بر تولید کاندید با استفاده از اصل «عدم تناوب ابرالگوهای یک نامتناوب»، سبب کاهش قابل ملاحظه‌ای در اندازه مجموعه کاندیدهای تولیدشده برای حل مساله می‌گردند اما همواره دو اشکال اصلی بر این روش‌ها وارد است. نخستین اشکال تولید تعداد فوق العاده زیادی کاندیدا در قالب مجموعه‌های‌کاندید است که با افزایش تعداد قلم‌داده‌ها بسیار محسوس میشود. دومین مشکل اساسی پویش مکرر و چندین باره پایگاه‌تراکنش برای مطابقت کاندیداهای تولیدشده با تراکنش‌ها به‌منظور تشخیص تناوب یا عدم تناوب آنهاست.

هان[۱۴] در سال ۲۰۰۰ روشی به نام رشد الگوی متناوب[۱۵] را ابداع نمود که مجموعه کامل همه مجموعه‌قلم‌داده‌های‌متناوب را بدون تولید کاندیداها کاوش می‌نمود. این روش مبتنی بر تقسیم وحل بوده و با یک بار پویش پایگاه‌تراکش، همه قلم‌داده‌های متناوب موجود در پایگاه تراکنش را یافته و بر اساس‌تعداد تکرارشان در پایگاه‌تراکنش (میزان تناوب)، به صورت نزولی مرتب می‌کند. بر مبنای لیست مرتب قلم‌داده‌های‌متناوب، پایگاه‌تراکنش در قالب یک درخت‌الگوی‌متناوب[۱۶] فشرده‌سازی می‌شود که این درخت شامل همه اطلاعات موجود در پایگاه‌تراکنش است و می‌تواند در فرآیند کاوش مورد استفاده قرارگیرد. کاوش درخت‌الگوی‌متناوب با شروع از هر قلم‌داده متناوب (به عنوان الگوی پسوندی اولیه)

آغازمی‌گردد و پایگاه الگوهای‌شرطی آن قلم‌داده‌ها ساخته می‌شود. پایگاه الگوهای شرطی یک الگو شامل مجموعه پیشوندی مسیرهایی از درخت الگوی متناوب است که الگوی مورد نظربه عنوان پسوند در آن مسیرها به کار رفته باشد. پس از این مرحله، درخت الگوی متناوب‌شرطی الگوی مزبور ساخته می‌شود و عملیات کاوش به صورت بازگشتی برروی این درخت اعمال می‌گردد. رشد الگوها با اتصال الگوهای پسوندی به الگوهای متناوب کاوش شده از درخت‌های‌الگوی‌متناوب‌شرطی، ساخته می‌شوند. الگوریتم رشدالگوی‌متناوب مساله یافتن الگوهای متناوب بزرگ را به مساله جستجوی الگوهای متناوب کوچکتر به صورت بازگشتی تبدیل می‌نماید و سپس پسوندها را به الگوهای یافته شده متصل می‌کند. این روش با انتخاب درست قلم‌داده‌ها می‌تواند کمترین قلم‌داده‌ها را به عنوان پسوند در نظر بگیرد. مطالعات انجام شده در ارتباط با کارآیی این‌الگوریتم نشان داده‌است که روش مزبور، زمان جستجو را به شکل قابل ملاحظه‌ای کاهش می‌دهد[۵].

۴,۱ کارهای مرتبط

تاکنون پیشنهادهای زیادی برای بهبود روش رشد الگوهای متناوب ارائه‌شده‌است. در روش رشد الگوی متناوب یکبار در ابتدای پویش پایگاه‌تراکنش درخت الگوی متناوب ساخته شده و در ضمن انجام فرآیند کاوش، به دفعات، درخت‌های‌شرطی به منظور رشدالگو تولید و پیمایش می‌شوند. یکی از راه‌کارهای سریعتر کردن روش رشد الگوی متناوب به نام رشدالگوی‌متناوب‌استار[۱۷] در سال ۲۰۰۵ توسط گراهنه و ژو[۱۸] ارائه شده است[۶]. در این روش نشان داده شده است که بیش از ۸۰٪ زمان پردازنده صرف ساخت و پیمایش درخت های‌الگوی‌متناوب می‌شود. تولید هر درخت شرطی جدید نیازمند دو بار پویش درخت جاری است. بار اول تعداد تکرارهای هر قلم‌داده به دست می‌آید و در بار دوم تراکنش‌های دربردارنده الگوی جاری از شاخه‌های درخت خوانده شده و پس از اصلاح و حذف قلم‌داده‌های کم‌تکرار، در درخت شرطی جدید قرار می‌گیرند. با بکارگیری ساختار داده ماتریس می‌توان با یکبار پیمایش درخت جاری، درخت شرطی جدید را ساخت. در این روش در هنگام تولید درخت شرطی جدید دیگر نیازی به پیمایش درخت جاری برای یافتن تعداد تکرار قلم‌داده‌ها نیست؛ بلکه با مراجعه به سطر مربوطه می‌توان جدول سرآیند[۱۹]درخت شرطی جدید را بدست آورد و تنها با یکبار پیمایش درخت جاری به درخت شرطی رسید. بررسی نتایج و نمودارهای مربوط به روش استار نشان می‌دهد که زمان اجرای الگوریتم کاهش چشمگیری پیدا می‌کند. با این وجود الگوریتم همچنان برای نگهداری تراکنش‌ها از ساختار لیست پیوندی استفاده میکند که پیمایش آن زمانبر بوده و نیز برای تولید الگوهای متناوب در هر مرحله از عملیات کاوش باید درخت‌های‌شرطی را تولید و پیمایش نماید که با افزایش عمق درخت اصلی باعث کاهش کارایی الگوریتم میشود.

در سال ۲۰۱۱ کی-چانگ[۲۰] و همکاران پیشنهاد دیگری برای بهبود روش رشد الگوی متناوب ارائه کردند[۷]. بر اساس تحقیقات آن‌ها عملیات درج تراکنش‌ها به منظور ساختن درخت‌الگوی‌متناوب بخشی از زمان اجرای الگوریتم را به خود اختصاص می‌دهد. اگرچه این زمان زیاد نیست اما مشکلی که باعث این اتلاف وقت می‌شود در مراحل کاوش درخت، خود را نشان می‌دهد. مشکل بدین صورت است که نگهداری فرزندان هر گره در یک لیست باعث می‌شود تا هنگام پیداکردن شاخه‌های درخت، زمان محسوسی به خاطر جستجو در این لیست از دست داده‌شود. کی-چانگ برای حل این مشکل ساختارجدیدی به نام جدول‌آدرس[۲۱] را پیشنهاد کرده‌است. این ساختار آدرس هر گره فرزند را در یک جدول‌درهم نگهداری می‌کند و در هنگام مراجعه به هر گره به راحتی می‌توان فرزندان را پیداکرد. برای حل مشکل ساختن متوالی درخت‌های شرطی، گروه کی-چانگ روش جدیدی برای کاوش درخت پیشنهاد کرده‌اند. این روش بیان می‌کند که برای قلم‌داده‌های پرتکرارتر بهتر است درخت‌های‌شرطی بر روی درخت اصلی تشکیل شوند. با این کار علاوه بر صرفه‌جویی در حافظه، سرعت عملیات کاوش نیز بهبود می‌یابد. پارامتر سطح‌درخت[۲۲]، قلم‌داده‌هایی را مشخص می‌کند که درخت‌های شرطی آنها باید بر روی درخت اصلی ساخته شود. با توجه به آزمایشات فراوانی که بر روی پایگاه‌های‌تراکنش انجام گرفته است، مقدار ۵۰٪ به عنوان مقدار پایه‌ای برای این پارامتر ارائه شده‌است؛ یعنی اینکه اگر برای قلم‌داده‌های نیمه بالایی جدول‌سرآیند درخت‌های شرطی را بر روی درخت اصلی و برای قلم‌داده‌های نیمه پایینی جدول‌سرآیند همانند روش رشد الگوی متناوب عمل کنیم به بهترین زمان‌های اجرا دست می‌یابیم. اگرچه در این روش میزان نیاز به تولید درخت‌های شرطی تا ۵۰٪ کاهش می‌یابد اما مشکل درخت‌های شرطی همچنان در آن وجود دارد. از طرف دیگر تمامی پایگاه‌های تراکنش با مقدار ۵۰٪ برای پارامتر سطح‌درخت سازگار نیستند و لازم است تا برای مقداردهی این پارامتر از روش‌های بهتری استفاده شود. مشکل مشترک در هر دو روش ذکر شده این است که همچنان برای عملیات کاوش به درخت‌های‌شرطی نیاز دارند؛ هرچند که در روش کی-چانگ این نیاز بسیار کمتراست. در الگوریتم پیشنهادی ما سعی بر این بوده‌است تا با تغییر ساختار نگهداری درخت از لیست پیوندی به ترکیبی از جداول‌درهم و آرایه زمان پیمایش شاخه‌ها را کاهش داده و با تغییر روش کاوش از پیمایش درخت دربردارنده تراکنش‌ها و تولید درختان شرطی به یک روش کاملا مبتنی بر ماتریس دوبعدی، ضمن افزایش سرعت عملیات کاوش، تعداد بیشتری الگوی متناوب در هر مرحله کاوش تولید نماییم.

۲. روش پیشنهادی

این بخش به بررسی قسمت‌های مختلف الگوریتم پیشنهادی بر روی یک پایگاه‌تراکنش نمونه می‌پردازد. در شکل‌های (۱) و (۲) کلیات الگوریتم پیشنهادی آورده شده است.

۱.۲ فرضیات

در کاوش قوانین وابستگی رخ دادن مجموعه قلم‌داده‌ها مورد توجه است و تعداد دفعات تکرار هر قلم‌داده تاثیری در ارتباط بین آن‌ها ندارد. با توجه به این فرض پایگاه تراکنش مانند D را بر روی مجموعه قلم‌داده L={۱،۲،۳،۴،۵،۶،۷،۸،۹،۱۰،۱۱،۱۲،۱۳،۱۴،۱۵،۱۶،۱۸} که در جدول (۱) آمده است را در نظر بگیرید. این پایگاه‌داده شامل تراکنش‌هایی است که هر کدام از آن‌ها شامل چند قلم‌داده می‌باشند.

جدول (۱) پایگاه تراکنش D

تراکنش	مجموعه قلم‌داده‌ها
۱	۱۶،۳،۴،۶،۹،۷،۱۳،۱
۲	۶،۲،۱۳،۳،۱،۱۲،۱۵
۳	۲،۶،۱۵،۱۰،۸
۴	۳،۱۱،۲،۱۸،۱۶
۵	۱۴،۳،۵،۱۶،۱۲،۱۳،۱،۶

همانطور که مشاهده می‌شود برای راحتی از اعداد به عنوان نماد قلم‌داده‌ها استفاده شده است. با یکبار پیمایش پایگاه‌تراکنش تعداد تکرار هر قلم‌داده مشخص می‌شود. در گام بعدی و با در نظر داشتن حد آستانه پشتیبان کمینه[۲۳]، قلم‌داده‌های کم‌تکرار هر تراکنش حذف شده و سایر قلم‌داده‌ها بر اساس میزان تکرار به طور نزولی مرتب می‌شوند. در الگوریتم پیشنهادی پایگاه‌تراکنش دو بار پویش می‌شود. در پویش نخست قلم‌داده‌های متناوب [۲۴]شناسایی می‌شوند و در پویش دوم با حذف قلم‌داده‌های کم‌تکرار، با درنظر گرفتن حدآستانه پشتیبان کمینه، تراکنش‌ها اصلاح می‌شوند و در شاخه‌های درخت قرار می‌گیرند. حد آستانه پشتیبان کمینه برابر با سه، ۳ = δ، فرض شده است. در جدول (۲) قلم‌داده‌های متناوب بر اساس میزان تکرار به صورت نزولی نشان داده شده‌اند. در صورت تکرار مساوی برای چند قلم‌داده، ارزش عددی آن‌ها به صورت نزولی، مد نظر قرار گرفته‌است.

جدول (۲) قلم‌داده‌های متناوب

قلم داده	۳	۶	۱	۲	۱۳	۱۶
تکرار	۴	۴	۳	۳	۳	۳

۲،۲ فاز اول: پیش پردازش

همانطور که در شکل (۱) مشاهده می‌شود پایگاه‌تراکنش‌ها ابتدا یکبار مورد پویش قرار می‌گیرد و با توجه به حد آستانه قلم‌داده‌های متناوب پیدامی‌شوند. سپس در پویش بعدی قلم‌داده‌های غیرمتناوب از تراکنش‌ها حذف شده و قلم‌داده‌های متناوب در تراکنش‌ها بر اساس میزان تکرارشان به طور نزولی قرار می‌گیرند. در اینجا مرحله پیش پردازش به اتمام رسیده و الگوریتم وارد فاز کاوش می‌شود. در جدول (۳) پایگاه‌تراکنش بعد از اتمام مرحله پیش‌پردازش نشان داده شده‌است.

جدول (۳) پایگاه تراکنش D بعد از اصلاح تراکنش‌ها

تراکنش	مجموعه قلم‌داده‌ها
۱	۳،۶،۱،۱۳،۱۶
۲	۳،۱،۱۶
۳	۳،۶،۱،۲،۱۳
۴	۶،۲
۵	۳،۶،۱،۱۳،۱۶

برای مواردی که تعداد تکرار قلم‌داده‌ها یکسان است، اولویت با عدد کمتر است. برای مثال هر دو قلم‌داده ۳ و ۶ به تعداد یکسان تکرار شده‌اند لذا اگر هر دو در یک تراکنش با هم بیایند، ابتدا قلم‌داده ۳ و سپس ۶ قرار می‌گیرد.

۱,۲,۲ حد آستانه

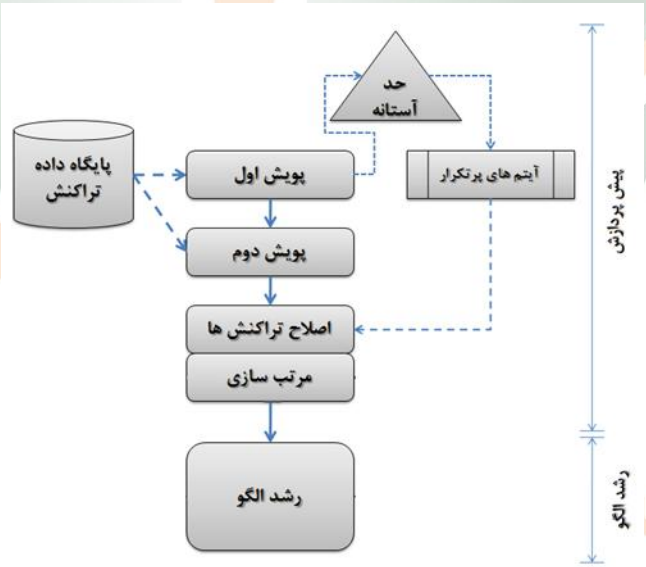
حد آستانه یک مقدار عددی است که گاه به صورت مطلق و گاه به صورت درصد توسط کاربر تعیین می‌شود. این عدد تعداد تکرار لازم برای متناوب بودن یک قلم‌داده را مشخص می‌کند و با علامت δ نمایش داده می‌شود.

۲. ۳ فاز دوم: رشد الگو

این مرحله از الگوریتم به سه بخش کلی تقسیم می‌شود که در ادامه مورد بررسی قرارمی‌گیرند

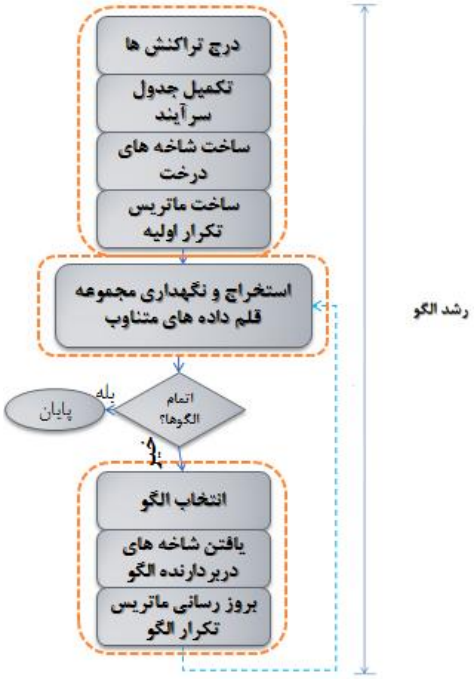
۱,۳,۲ آماده سازی

دو هدف اصلی این بخش به ترتیب عبارتند از خواندن تراکنش‌های اصلاح‌شده و قراردادن آن‌ها در شاخه‌های درخت و تشکیل ماتریس تکرار اولیه. هنگام خواندن هر تراکنش با مراجعه به جدول راهنما وجود شاخه‌ای با قلم‌داده آغازین مشابه به قلم‌داده آغازین تراکنش جاری مورد بررسی قرار می‌گیرد. در صورتیکه شاخه‌ای وجود داشته باشد تراکنش جاری در آن قرار گرفته و در صورت عدم وجود، شاخه جدیدی ایجاد شده و تراکنش به‌طور کامل در آن درج می‌شود.



شکل (۱) شماتیک الگوریتم پیشنهادی - پیش پردازش

بعد از اتمام درج با توجه به قلم‌داده‌های بکار رفته در تراکنش، جدول راهنما بروز می‌شود و موقعیت قلم‌داده‌ها در بخش موقعیت جدول قرارمی‌گیرند. با اتمام خواندن هر تراکنش ماتریس تکرار اولیه بروز می‌شود. تعداد سطرها و ستون‌های این ماتریس به تعداد قلم‌داده‌های متناوب درون پایگاه‌تراکنش است. اگر قلم‌داده‌های‌متناوب بر اساس میزان تکرارشان به صورت نزولی مرتب شده و به صورت $\{F_1, F_2, \dots, F_n\}$ نشان داده شوند، برای $n = 5$ ، ماتریس تکرار اولیه ساختاری شبیه جدول (۴) خواهد داشت.



شکل (۲) شماتیک الگوریتم پیشنهادی- کاوش

مقادیر درون خانه‌های ماتریس تکرار که به صورت V_{ij} نشان داده شده‌است، بیانگر میزان تکرار قلم‌داده‌های F_i و F_j با یکدیگر است. جدول (۵) ساختار شاخه‌های درخت را برای شاخه‌ای با $n=5$ قلم‌داده متناوب نشان می‌دهد. الگوریتم برای دسترسی و پیمایش شاخه‌های درخت از ساختاری به نام جدول راهنما استفاده می‌کند. در حقیقت این جدول بهبود یافته جدول‌سرآیند در روش رشد الگوی متناوب است که با کمک آن دیگر نیازی به استفاده از اشاره‌گرهای درونی برای یافتن موقعیت قلم‌داده‌ها در درخت نیست. جدول۶ نمونه‌ای از این ساختارداده را نشان می‌دهد.

جدول (۴) ساختار ماتریس تکرار

F_1	V_{11}	-	-	-	-
F_2	V_{12}	V_{22}	-	-	-
F_3	V_{13}	V_{23}	V_{33}	-	-
F_4	V_{14}	V_{24}	V_{34}	V_{44}	-
F_5	V_{15}	V_{25}	V_{35}	V_{45}	V_{55}
	F_1	F_2	F_3	F_4	F_5

جدول (۵) ساختار شاخه درخت

قلم داده	F_1	F_2	...	F_n
تکرار	V_1	V_2	...	V_n

جدول (۶) جدول راهنما

قلم داده	شروع	موقعیت ها
F_1	S_1	$\{R_{1i},\ldots,R_{1k}\}$
F_2	S_2	$\{R_{2i},\ldots,R_{2p}\}$
...	...	...
F_n	S_n	$\{R_{ni},\ldots,R_{nq}\}$

ستون سمت چپ قلم‌داده‌های متناوب را نشان می‌دهد، ستون میانی (S_i) شاخه‌ای را نشان می‌دهد که با F_i شروع شده است و ستون سمت راست (R_i) دنباله‌ای از اعداد را شامل می‌شود که بیانگر شاخه‌هایی از درخت است که قلم‌داده F_i در آنها وجود دارد. برای روشن‌تر شدن مطلب تراکنش نخست از پایگاه‌تراکنش D را در نظر بگیرید. این تراکنش با قلم‌داده۳ آغاز می‌شود. از آنجاییکه در ابتدا جدول راهنما خالی‌است لذا با مراجعه به آن شاخه‌ای برای قلم‌داده۳ پیدا نمی‌شود و درنتیجه شاخه‌ای جدید ایجاد شده و قلم‌داده‌های این تراکنش در آن قرار می‌گیرند. تیجه این عملیات در شکل (۳) نشان داده شده است.

قلم داده	شروع	موقعیت ها
3	1	1
6	-	1
1	-	1
2	-	-
13	-	1
16	-	1

جدول راهنما

قلم داده	3	6	1	13	16
تکرار	1	1	1	1	1

شاخه های درخت

$T_1 = \{3, 6, 1, 13, 16\}$

شکل (۳) وضعیت شاخه‌های درخت بعد از ورود اولین تراکنش

در ادامه تراکنش دوم از پایگاه‌تراکنش خوانده می‌شود. این تراکنش نیز با قلم‌داده۳ آغاز می‌شود. این‌بار با مراجعه به جدول راهنما، شاخه شماره یک به عنوان شاخه مقصد بازگردانده می‌شود. مقدار عددی خانه تکرار برای این قلم‌داده در شاخه شماره یک، یک واحد افزایش می‌یابد. قلم‌داده دوم در این تراکنش قلم‌داده۲ می‌باشد و این درحالیست که قلم‌داده بعدی در شاخه شماره یک، قلم‌داده۶ است؛ یعنی اینکه برای درج ادامه تراکنش باید شاخه دیگری انتخاب شود. در جدول راهنما در حال حاضر شاخه‌ای برای قلم‌داده۲ ثبت نشده است و لذا برای ادامه درج باید شاخه جدیدی ساخته شود. نتیجه این عملیات در شکل (۴) نشان داده‌شده‌است.

قلم داده	شروع	موقعیت ها
3	1	1,2
6	-	1
1	-	1
2	-	2
13	-	1
16	-	1,2

جدول راهنما

قلم داده	3	6	1	13	16
تکرار	2	1	1	1	1

قلم داده	3	2	16
تکرار	۱	1	1

شاخه های درخت

$T_2 = \{3, 2, 16\}$

شکل (۴) وضعیت شاخه‌های درخت بعد از ورود دومین تراکنش

با ادامه این روند تمامی تراکنش‌ها در شاخه‌های مناسب قرارمی‌گیرند. شکل (۵) نتیجه نهایی این عملیات را نشان می‌دهد. بعد از درج شدن هر تراکنش، ماتریس تکرار نیز بروز می‌شود. تراکنش نخست شامل قلم‌داده‌های ۳،۶،۱،۱۳ و ۱۶ است. ستون مربوط به قلم‌داده۳ را در نظر بگیرید. از آنجاییکه این قلم‌داده با قلم‌داده‌های ۶،۱،۱۳ و ۱۶ تکرار شده‌است لذا سطرهای مربوط به این قلم‌داده‌ها برای ستون قلم‌داده۳ مقدار یک می‌گیرند. درصورتیکه این خانه‌ها از قبل مقدار داشته‌باشند آن مقدار یک واحد افزایش می‌یابد.

قلم داده	شروع	موقعیت ها
3	1	1,2,3
6	4	1,3,4
1	-	1,3
2	-	2,3,4
13	-	1,3
16	-	1,2

جدول راهنما

قلم داده	3	6	1	13	16
تکرار	4	3	3	2	2

قلم داده	3	2	16
تکرار	۱	1	1

قلم داده	3	6	1	2	13
تکرار	۱	۱	۱	1	1

قلم داده	6	2
تکرار	1	1

شاخه های درخت

شکل (۵) وضعیت نهایی بعد از ورود آخرین تراکنش

جدول (۷) وضعیت نهایی این ماتریس را بعد از ورود تمامی تراکنش‌ها نشان می‌دهد. قطر اصلی ماتریس در این مرحله تعداد تکرار قلم‌داده‌های متناوب پایگاه‌تراکنش را نشان می‌دهد. از آنجاییکه قلم‌داده‌ها در تراکنش‌ها بر اساس میزان تکرار مرتب شده‌اند لذا ماتریس تکرارمتقارن می‌شود و از نیمه بالایی آن استفاده نمی‌شود؛ به همین علت مقدار صفر برای خانه‌های بالای قطر اصلی در نظر گرفته‌شده‌است.

جدول (۷) ماتریس تکرار بعد از ورود تمام تراکنش‌ها

3	4	-	-	-	-	-
6	3	4	-	-	-	-
1	3	3	3	-	-	-
2	2	2	1	3	-	-
13	3	3	3	1	3	-
16	3	2	2	1	2	3
	3	6	1	2	13	16

۲,۳,۲ استخراج الگو

بعد از اتمام خواندن تراکنش‌ها و تکمیل ماتریس تکرار اولیه و یا بعد از مرحله بازسازی، الگوریتم وارد بخش استخراج الگو می‌شود. با مراجعه به ماتریس تکرار و در نظر گرفتن حد آستانه، مکان‌هایی از ماتریس را که مقدارشان بیشتر از حد آستانه است برای استخراج الگو در نظر گرفته می‌شوند. ماتریس تکرار جدول (۷) را در نظر بگیرید، اگر $V_{ij} \geq \delta$ باشد و الگوی متناوب جاری P_k باشد، الگو های جدید به صورت زیر تولید می‌شوند:

$$P_{k+۱} = P_k \wedge F_i \qquad (۱)$$

$$P_{k+۲} = P_k \wedge F_j \qquad (۲)$$

$$P_{k+۳} = P_k \wedge F_i \wedge F_j \qquad (۳)$$

در فرمول‌های (۱)، (۲) و (۳) نماد $\wedge$ بیانگر عملگر اتصال است و به صورت زیر تعریف می‌شود:

$$F_i \wedge F_j = \begin{cases} F_i F_j & i \neq j \\ F_i & i = j \end{cases} \qquad (۵)$$

باید توجه داشت که در اولین ورود به بخش کاوش، هیچ الگویی از قبل وجود ندارد و لذا داریم: $P_k = \phi$

برای مثال با بررسی ماتریس تکرار در این مرحله تمامی الگوهای متناوب به طول ۲ قابل استخراج هستند. نتیجه این عملیات در شکل ۶ نشان داده‌شده‌است.

3	4	0	0	0	0	0
6	3	4	0	0	0	0
1	3	3	3	0	0	0
2	2	2	1	3	0	0
13	3	3	3	1	3	0
16	3	2	2	1	2	3
	3	6	1	2	13	16

ماتریس تکرار

{3,6}
{3,1}
{3,13}
{3,16}
{6,1}
{6,13}
{1,13}

الگوهای متناوب به طول دو

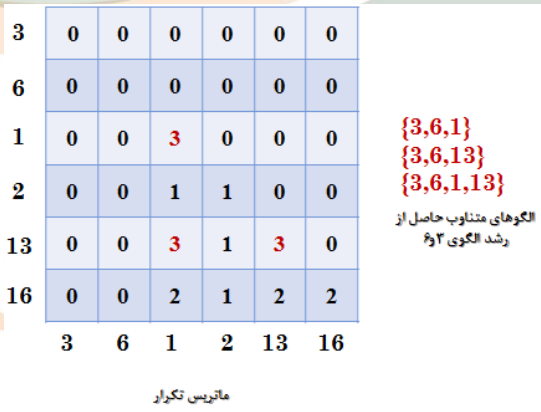
شکل(۶) استخراج الگوهای متناوب به طول دو

۳،۳،۲ بازسازی

در این مرحله یک الگوی متناوب از الگوهای استخراج شده در مرحله استخراج الگو انتخاب شده و رشد می‌یابد. فرآیند رشد بدین شکل است که ابتدا از جدول راهنما موقعیت‌های مربوط به قلم‌داده‌های درون الگوی انتخاب شده به‌دست آمده و سپس با اشتراک‌گیری از این موقعیت‌ها شاخه‌های در بردارنده الگوی انتخابی مشخص می‌شود. حال با مراجعه به این شاخه‌ها و خواندن قلم‌داده‌هایی که شامل می‌شوند الگوریتم مقادیر درون خانه‌های ماتریس تکرار را بروز می‌کند. در اینجا مرحله بازسازی به اتمام می‌رسد و الگوریتم دوباره وارد بخش استخراج الگو می‌شود؛ با این تفاوت که این بار الگوهای داری پیشوند الگوی انتخابی استخراج می‌شوند. در شکل (۷) و (۸) نتیجه عملیات بازسازی برای الگوی {۳،۶} نشان داده شده‌است.

شکل (۷) پیداکردن شاخه‌ها برای الگوی {۳،۶}

با بررسی ماتریس بازسازی شده سه الگوی جدید استخراج می‌شود؛ این الگوهای در سمت راست شکل ۳-۸ نشان داده‌شده‌اند. با ادامه این روند برای سایر الگوها تمامی مجوعه قلم‌داده‌های پایگاه‌تراکنش به‌دست می‌آیند.



شکل (۸) ماتریس تکرار بازسازی شده برای الگوی {۳،۶}

۳. نتایج تجربی

در این بخش الگوریتم پیشنهادی به همراه چند الگوریتم دیگر در شرایط یکسان پیاده سازی و مورد آزمایش قرار گرفته‌اند تا به طور عملی کارایی هر یک از آن‌ها مشخص شود. سعی بر این بوده است تا از کدهای استاندارد و موجود برای الگوریتم‌های ذکر شده استفاده شود. پیاده‌سازی الگوریتم‌ها با زبان سی‌شارپ نسخه ۴ و در محیط ویژوال استودیو ۲۰۱۰ انجام شده‌است. در جدول (۸) مشخصات سیستمی که آزمایشات روی آن انجام شده، آورده شده است.

جدول (۸) مشخصات سیستم مورد استفاده در آزمایشات

Processor	Intel Core 2 Quad 6600
Ram	2 GB
OS	MS Windows 7 SP1

۱,۳ الگوریتم‌های مورد آزمایش

الگوریتم پیشنهادی به همراه الگوریتم‌های رشدالگوی‌متناوب، FP-Growth-Star و IFP بر روی دیتاست‌های مختلف اجرا شده‌اند تا تاثیر پیشنهادات داده شده در عمل مورد بررسی قرار بگیرد.

۲,۳ دیتاست‌های استفاده شده در آزمایش

در جداول (۹) و (۱۰) اطلاعات کلیدی در مورد دیتاست‌های استانداردی که در آزمایش مورد استفاده قرار گرفته‌اند، به طور خلاصه آورده شده است.

جدول (۹) دیتاست T.I.D.K

نام دیتاست	T10I4D100K
تعداد تراکنش ها	100,000
تعداد آیتم ها	870
میانگین طول تراکنش	10.1

۳,۳ ارزیابی کارایی

تمامی الگوریتم‌هایی که در بخش ۱,۳ معرفی شدند توسط زبان سی‌شارپ پیاده‌سازی کامپایل شده‌اند و از ساختارهای داده استاندارد Framework ۴,۰ استفاده شده‌است. بستر امن[۲۵] انجام شده و در هیچ یک از آنها از تکنیک‌های موازی سازی استفاده نشده است. زمان‌های اندازه گیری شده از اولین نمونه‌سازی اشیاء مربوط به الگوریتم‌های مختلف تا زمان اتمام پروسه کاوش[۲۶] هر شیئ در نظر گرفته شده است. کلاس‌های مربوط به کار با پایگاه‌تراکنش‌ها در تمامی روش‌ها یکسان پیاده سازی شده است و تا جای ممکن برای عملیات یکسان، از کدها و ساختارهای داده‌ای مشابه استفاده شده است تا نتایج ارزیابی‌ها دقیق باشند.

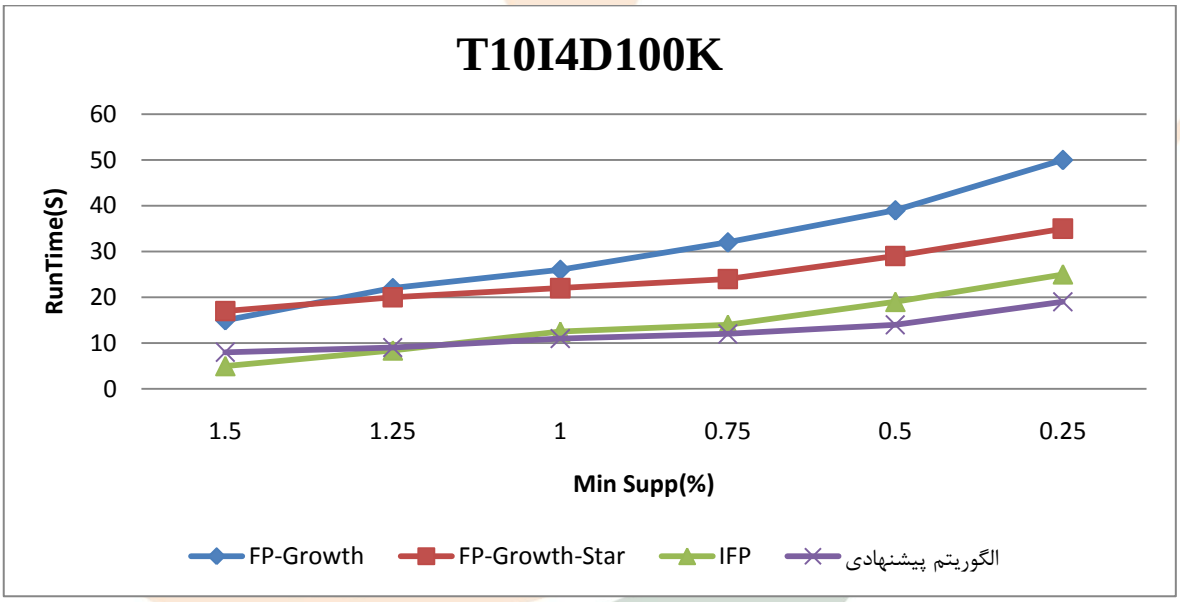
جدول (۱۰) دیتاست T.I.D.K

نام دیتاست	T40I10D100K
تعداد تراکنش ها	100,000
تعداد آیتم ها	942
میانگین طول تراکنش	39.6

۱,۳,۳ نتایج اجرا بر روی دیتاست T۱۰I۴D۱۰۰K

نتایج اجرای الگوریتم‌های FP-Growth، FP-Growth-Star، IFP و الگوریتم پیشنهادی بر روی این دیتاست در شکل (۹) آورده شده است. در این نمودار محور افقی بیانگر حد آستانه پشتیبانی کمینه و محور عمودی بیانگر مدت زمان اجرای الگوریتم بر حسب ثانیه است. الگوریتم استار با استفاده از تکنیک آرایه برای ساختن درخت‌های شرطی یک بار کمتر نسبت به الگوریتم الگوی‌متناوب مجبور به پیمایش درخت اصلی است و لذا بهتر از آن عمل می‌کند. هرچه حد آستانه کمتر باشد، درخت عریض‌تر و عمیق‌تر می‌شود در نتیجه الگوریتم استار کارایی خود را بهتر نشان

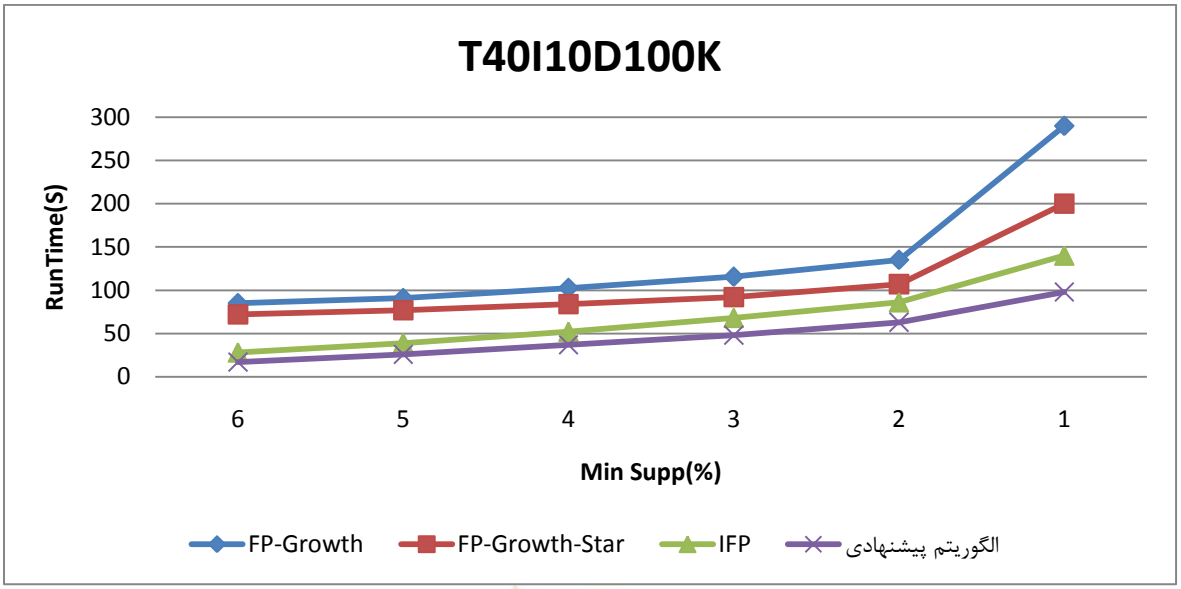
می‌دهد. با افزایش حد آستانه تعداد قلم‌داده‌های متناوب کمتر شده و در نتیجه آن، الگوهای متناوب نیزکمتر می‌شود. با کاهش تعداد الگوها، درخت کوچکتر و کوتاه‌تر می‌شود و در نتیجه الگوریتم رشد الگوی متناوب سریعتر آن‌را پیمایش می‌کند و چون درخت‌های شرطی نیز کمتر و کوتاه‌تر می‌شوند لذا سرعت اجرای الگوریتم بیشتر می‌شود. با افزایش حد آستانه و کوچک‌تر شدن درخت، سربار تولید ماتریس تکرار و پیمایش آن باعث کند شدن الگوریتم استار نسبت به الگوریتم الگوی متناوب می‌شود به گونه‌ای که در آستانه ۱٫۵٪ الگوریتم استار بدتر عمل می‌کنند. الگوریتم IFP به علت حذف ۵۰٪ درخت‌های شرطی در همه آستانه‌ها بهتر از الگوریتم استار و الگوی متناوب عمل می‌کند. الگوریتم پیشنهادی در آستانه‌های کمتر که درخت عریض‌تر و عمیق‌تر است کارایی خود را نشان می‌دهد؛ چرا که با هربار پیمایش ماتریس، اکثر خانه‌ها مقداری بیشتر از حد آستانه دارند و در نتیجه زمانی که صرف پیمایش ماتریس می‌شود با استخراج الگو همراه است. بهبود روش کاوش نیز باعث می‌شود تا در هر بار پیمایش ماتریس، الگوهای بیشتری استخراج شوند. با افزایش حد آستانه و کوچک‌تر شدن درخت، تاثیر سربار تشکیل ماتریس دوبعدی همانند الگوریتم استار مشخص‌تر می‌شود. در حد آستانه ۱٪ هر دو روش پیشنهادی و IFP بسیار نزدیک به هم می‌شوند اما با گذشت از این حد آستانه، الگوریتم IFP عملکرد بهتری از خود نشان می‌دهد.



شکل (۹) نتایج اجرا بر روی دیتاست T۱۰I۴D۱۰۰K

۲,۳,۳ نتایج اجرا بر روی دیتاست T۴۰I۴D۱۰۰K

نتایج اجرای الگوریتم‌های FP-Growth، FP-Growth-Star، IFP و الگوریتم پیشنهادی بر روی این دیتاست در شکل (۱۰) آورده شده است. در این دیتاست به علت افزایش متوسط طول تراکنش‌ها عمق درخت بیشتر می‌شود. از طرف دیگر افزایش طول تراکنش‌ها باعث شده تا قلم‌داده‌های بیشتری در یک تراکنش قرارگیرند و لذا تعداد تکرار هر قلم‌داده بیشتر شده است. با در نظر گرفتن این نکته کاملا مشهود است که با کمتر کردن حد آستانه پشتیبان کمینه تعداد قلم‌داده‌های متناوب زیاد شده و در نتیجه تعداد الگوهای متناوب نیز زیاد می‌شوند. این امر تاثیرش را بر روی هر الگوریتم با افزایش زمان اجرا نشان می‌دهد. به همین خاطر است که حد آستانه پشتیبان کمینه در محدوده ۱٪ تا ۶٪ انتخاب شده است. با افزایش قلم‌داده‌های متناوب تعداد شاخه‌های درخت نیز افزایش می‌یابد و درخت عریض‌تر می‌شود. در این شرایط الگوریتم‌هایی که درخت را بیشتر پیمایش می‌کنند کندتر می‌شوند. برای آستانه‌های بزرگتر درخت کوچک‌تر می‌شود و در نتیجه روش پیشنهادی به خاطر پیمایش ماتریس دچار سربار می‌شود؛ چرا که اکثر خانه‌های ماتریس از حدآستانه کوچکترند. برای آستانه‌های کوچکتر درخت انبوه‌تر شده و لذا روش پیشنهادی با هر بار پیمایش ماتریس، با توجه به تغییر روش کاوش، می‌تواند تعداد بیشتری الگوی متناوب تولید کند و به همین خاطر بهتر از سایر الگوریتم‌ها عمل می‌کند.



شکل (۱۰) نتایج اجرا بر روی دیتاست T۴۰I۱۰D۱۰۰K

۴. نتیجه گیری و کارهای آینده

مهم‌ترین بخش در کشف قوانین وابستگی یافتن الگوهای متناوب است. از سال ۱۹۹۴ که آگراوال خاصیت Apriori را در مجموعه قلم‌داده‌های متناوب ارائه‌کرد تاکنون روش‌های متعددی برای یافتن الگوهای متناوب ارائه شده‌اند. تمامی این روش‌ها در دو دسته کلی قرار می‌گیرند. دسته اول آنهایی هستند که بر اساس تولید کاندیدها و شمارش آن‌ها عمل‌می‌کنند و دسته دوم قلم‌داده‌های متناوب را رشد می‌دهند تا الگوهای جدید تولید شوند. یکی از معروف‌ترین روش‌های مبتنی بر رشد الگو، الگوریتم رشدالگوی‌متناوب است که پایگاه‌تراکنش‌ها را در یک ساختار درختی به نام درخت‌الگوی‌متناوب به صورت فشرده نگهداری می‌کند و از این ساختار برای کاوش الگوهای متناوب بهره می‌برد. مهم‌ترین مشکل این روش هزینه زیاد ساختن درخت‌های شرطی است که به طور بازگشتی ساخته و پیمایش می‌شوند. الگوریتم‌های استار و IFP با ارائه راه‌کارهایی سعی در بهبود زمان اجرای الگوریتم رشدالگوی‌متناوب داشته‌اند. الگوریتم استار با استفاده از یک ساختار ماتریسی تعداد دفعات پیمایش درخت را کاهش داده و باعث سرعت بخشیدن به الگوریتم می‌شود. در الگوریتم IFP با تغییر ساختار گره و تغییر روش کاوش و حذف حداکثر نیمی از درختان شرطی باعث بهبود الگوریتم می‌شود. بر اساس مطالعات انجام شده در این تحقیق، یکی از روش‌های ممکن برای بهبود بخشیدن به الگوریتم رشدالگوی‌متناوب تغییر ساختارداده مورد استفاده در پیاده‌سازی درخت است. نگهداری درخت در لیست‌های پیوندی اتلاف زمان زیادی را در هنگام پیمایش به الگوریتم تحمیل می‌کند، در صورتیکه با تغییر ساختار و نگهداری درخت در ساختارداده آرایه این اتلاف زمان به حداقل می‌رسد. از طرف دیگر بجای ساختن متوالی و بازگشتی درختان شرطی می‌توان از ساختار ماتریس برای کاوش و رشد الگوهای جدید استفاده کرد. این کار نه تنها از سربار ساختن درختان شرطی به شکل محسوسی می‌کاهد، بلکه این امکان را می‌دهد تا در هر بار اجرای عملیات کاوش، الگوی جاری تا حداکثر دو مرحله رشد داده شود. همانطور که نتایج تجربی نشان می‌دهد الگوریتم پیشنهادی بر روی دیتاست T۱۰I۴D۱۰۰K در اکثر آستانه‌ها و بر روی دیتاست T۴۰I۱۰D۱۰۰K در تمامی آستانه‌ها بهتر عمل می‌کند. اگرچه با استفاده از ساختار ماتریس دوبعدی در الگوریتم پیشنهادی عملیات کاوش سریعتر شده است اما حالت‌هایی وجود دارد که ماتریس به صورت ماتریس خلوت در می‌آید. در این حالت‌ها بیشتر خانه‌های ماتریس دارای مقدار عددی ۰ هستند و لذا پیمایش آنها نه تنها باعث تولید الگوی متناوب جدید نمی‌شود بلکه سرباری را در زمان ایجاد و زمان پیمایش بر الگوریتم تحمیل می‌نماید. طراحی ساختارهای داده‌ای که از پیمایش خانه‌های دارای مقدار ۰ در ماتریس دو بعدی جلوگیری میکند می‌تواند به عنوان یکی از زمینه‌های کاری آینده مورد بررسی قرار بگیرد.

۵. منابع

[۱] R. Agrawal, T. Imielinski, and A. Swami, *"Mining Association rules Between Sets of Items in Large Databases,"* Proceedings of ACM SIGMOD, May, ۱۹۹۳.

[۲] K. Cios, W. Pedrycz, R. Swiniarski, and LA. Kurgan, *Data Mining: A Knowledge Discovery Approach*, Springer, ۲۰۰۷.

[۳] R. Agrawal, T. Imielinski, and A. Swami, *"Mining association rules between sets of terms in large databases,"* In Proc. of the ACM- SIGMOD Int. Conf. on management of data (SIGMOD'۹۳), Washington, DC, pp. ۲۰۷–۲۱۶, ۱۹۹۳.

[۴] R. Agrawal, and R. Srikant, *"Fast algorithms for mining association rules,"* In Proc. Int. Conf. Very Large Data Bases (VLDB'۹۴), pages ۴۸۷–۴۹۹, Santiago, Chile ,Sept. ۱۹۹۴.

[۵] J. Han, J. Pei, and Y. Yin, *"Mining frequent patterns without candidate generation,"* In Proc. of the ACM-SIGMOD Int. Conf. on management of data (SIGMOD'۰۰), Dallas, TX, pp. ۱–۱۲, ۲۰۰۰.

[۶] G. Grahne, and J. Zhu, "*Fast Algorithms for Frequent Itemset Mining Using FP-Trees,"* IEEE Trans. On Knowledge and Data Engineering, vol. ۱۷, no. ۱۰, October ۲۰۰۵.

[۷] K. C. Lin, I. E. Liao, and Z. S. Chen, *"An Improved Frequent Pattern Growth Method for Mining Association Rules,"* Expert Systems with Applications, vol. ۳۸, no. ۵, pp. ۵۱۵۴-۵۱۶۱, ۲۰۱۱

زیرنویس‌ها

[۱] Frequent Patterns
[۲] Itemsets
[۳] Sequences
[۴] Structures
[۵] Frequent Itemset
[۶] Frequent Structural Pattern
[۷] Minimum Support
[۸] Minimum Confidence
[۹] Association Rule Mining
[۱۰] Agrawal
[۱۱] Candid Generation
[۱۲] Pattern Growth
[۱۳] Scan
[۱۴] Han
[۱۵] Frequent Pattern Growth
[۱۶] FP-Tree
[۱۷] FP-Growth*
[۱۸] Grahne & Zhu
[۱۹] Header Table
[۲۰] Ke-Chung
[۲۱] Address Table
[۲۲] Tree Level
[۲۳] Minimum Support Threshold
[۲۴] Frequent Items
[۲۵] Safe Mode
[۲۶] Mining Procedure