

پیشنهاد شیوه‌ای مبتنی بر الگوریتم PSO چند هدفه جهت استخراج قوانین انجمنی در داده کاوی

مهدی نصیری*، احمد اسمعیلی، بهروز مینایی، ناصر مزینی
دانشکده مهندسی کامپیوتر، دانشگاه علم و صنعت ایران، تهران، ایران

رسید مقاله: سیزدهم تیرماه ۱۳۹۰
پذیرش مقاله: بیست و پنجم آبان ماه ۱۳۹۰

چکیده

استخراج قوانین انجمنی یا همان استخراج روابط ممکن مابین داده های یک پایگاه داده بزرگ، یکی از وظایف مهم و بسیار زمان بری در داده کاوی به شمار می آید. بدین منظور، الگوریتم های متنوعی پیشنهاد شده اند که از نظر سرعت اجرا و کیفیت جواب ها با هم متفاوت می باشند. در این مقاله سعی شده است با استفاده از الگوریتم تکاملی **PSO** چند هدفه یک روش سریع و با کیفیتی مناسب جهت استخراج این قوانین ارائه شود. این الگوریتم پیشنهادی روی یک مجموعه داده از نوع تراکشی و با اندازه چهار هزار رکورد آزمایش شده است که نتایج حاصل حاکی از کارآمدی این روش پیشنهادی است.

کلمات کلیدی: قوانین انجمنی، داده کاوی، الگوریتم PSO. چند هدفه.

۱ مقدمه

با رشد سریع در اندازه و تعداد پایگاه داده ها، کاوش دانش، قواعد یا اطلاعات سطح بالا از داده ها به منظور پشتیبانی از تصمیم گیری ها و پیش بینی رفتارهای آتی، ضروری به نظر می رسد. روش های داده کاوی در جهت دستیابی به اهداف بالا به چندین گروه دسته بندی می شوند که عبارتند از: رده بندی، خوشه بندی، کاوش قواعد انجمنی، تحلیل الگوهای ترتیبی، پیش بینی، به تصویر کشیدن داده ها و غیره [۱-۲].

کاوش قوانین انجمنی یکی از وظایف مهم در داده کاوی، روند یافتن روابطی ما بین خصیصه ها یا ما بین مقادیر آن ها در یک پایگاه داده بزرگ است که در جهت امر تصمیم گیری کمک ساز باشند. یافتن چنین روابطی داخل یک مجموعه وسیعی از داده ها به علت ماهیت نمایی آن کار ساده ای نیست.

*عهده دار مکاتبات

آدرس الکترونیکی: Nasiri_m@iust.ac.ir

این روابط را می‌توان به صورت عبارات *IF-THEN* نمایش داد. به شرط بررسی شده در قسمت IF، مقدم و به آنچه که در قسمت THEN می‌آید، نتیجه گفته می‌شود که در این مقاله آن‌ها را به ترتیب با A و C نشان می‌دهیم.

بنابراین می‌توان یک رابطه را به صورت $A \rightarrow C$ نشان داد و یک چنین رابطه‌ای که ما بین خصیصه‌های داخل رکوردهای یک پایگاه داده با پیروی از ملاک‌های خاصی برقرار باشد قانون انجمنی گوئیم.

نوع دیگری از قوانین، که قوانین رده بندی نامیده می‌شوند، را نیز می‌توان با ساختار مشابهی نمایش داد ولی کاملاً با قوانین انجمنی متفاوت هستند. در چنین قوانینی، قسمت نتیجه تنها شامل مقادیری از یک خصیصه از پیش تعیین شده به نام رده است، حال آنکه چنین محدودیتی در مورد قوانین انجمنی برقرار نیست. یعنی هر خصیصه و هر تعداد خصیصه‌ای می‌تواند در هر دو قسمت قانون داشت. تنها محدودیتی که در مورد قوانین انجمنی باید برقرار باشد این است که دو طرف رابطه نباید دارای خصیصه مشترکی باشد یعنی $A \cap C = \phi$.

الگوریتم‌های مختلفی به منظور جستجوی این قوانین پیشنهاد شده‌اند [۳-۵]. در بخش بعدی، مقدمه مختصری در رابطه با این الگوریتم‌ها ارائه می‌کنیم. همانطور که خواهیم دید، این الگوریتم‌ها محدودیت‌های مربوط به خود را دارند. در این مقاله سعی گردیده است با ارائه روشی بر مبنای یک الگوریتم تکاملی سریع و کارآمدی به نام PSO برخی از این محدودیت‌ها مرتفع شوند.

در ادامه این مقاله ابتدا در بخش ۲ کلیتی از روش‌های موجود کاوش قوانین انجمنی ارائه می‌کنیم. در بخش ۳ توضیحی در مورد الگوریتم تکاملی PSO می‌دهیم. در بخش ۴ به جزئیات روش پیشنهادی خواهیم پرداخت. سپس تحلیل نتایج بدست آمده از اجرای این روش را در بخش ۵ ارائه کرده و در نهایت در بخش ۶ نتیجه‌گیری این مقاله ارائه خواهد شد.

۲ الگوریتم‌های کاوش قوانین انجمنی

الگوریتم‌های موجود برای جستجوی قوانین انجمنی به طور کلی بر مبنای رویکرد پیشنهادی Agrawal می‌باشند [۳,۶]. به عنوان نمونه، Apriori، SETM، AIS، جستجوی Pincer، و DIC تعدادی از الگوریتم‌های متداول بر مبنای این رویکرد می‌باشند [۶-۹]. این الگوریتم‌ها روی یک پایگاه داده باینری که پایگاه داده سبد خرید (market basket) نامیده می‌شود بکار می‌روند. به منظور آماده سازی پایگاه داده سبد خرید، هر رکورد از پایگاه داده اصلی به صورت یک رکورد باینری نشان داده می‌شود که در آن فیلدها، که آیتم نامیده می‌شوند، با استفاده از مقداری یکتا از هر پایگاه داده اصلی تعریف می‌شوند. برای یک پایگاه داده با تعداد بسیار زیادی از خصیصه‌ها که هر کدام دارای مقادیر متمایزی می‌باشند، تعداد کل این آیتم‌ها بسیار بزرگ می‌شود. در نتیجه ذخیره این پایگاه داده باینری مورد استفاده در الگوریتم‌های قانون کاوی، یکی از محدودیت‌های الگوریتم‌های حاضر است.

این الگوریتم‌ها در دو مرحله کار می‌کنند [۳]. مرحله اول برای تولید مجموعه آیتم‌های مکرر است که از تمامی مجموعه آیتم‌های ممکن، با استفاده از سنجشی با نام تعداد پشتیبان (support count) و پارامتری به نام پشتیبان

کمینه (minimum support) که توسط کاربر تعیین می‌شود، به دست می‌آید. تعداد پشتیبان (SUP) یک مجموعه آیتم عبارت است از تعداد رکوردهایی از پایگاه داده که تمامی عناصر آن مجموعه باشد. اگر ارزش پشتیبان کمینه خیلی بالا باشد، تعداد مجموعه‌های آیتم مکرر تولید شده کمتر خواهد بود و در نتیجه منجر به تولید قوانین کمتری می‌شود. به طریق مشابه اگر این مقدار خیلی پایین باشد، تقریباً تمامی مجموعه آیتم‌های ممکن، مکرر شده و در نتیجه تعداد قوانین نهایی تولید شده زیاد خواهد گردید. حال انتخاب قوانین بهتر از میان آن‌ها مشکل دیگری خواهد بود.

بعد از یافتن مجموعه آیتم‌های مکرر در مرحله اول، شروع به تولید قوانین با استفاده از یک پارامتر دیگری به نام اطمینان کمینه (minimum confidence) که توسط کاربر تعریف می‌شود می‌کنیم. ضریب اطمینان یا دقت پیش بینی یک قانون $A \Rightarrow C$ به صورت رابطه (۱) تعریف می‌شود.

$$Confidence = \frac{SUP(A \cup C)}{SUP(A)} \quad (1)$$

محدودیت دیگر این الگوریتم‌ها در نحوه‌ی کد کردن است که در آن‌ها نمادهای مجزایی برای هر مقدار ممکن خصیصه‌ها استفاده می‌شود. این شمای کد کردن ممکن است برای خصیصه‌های با مقادیر اسمی مفید باشد، ولی برای خصیصه‌های عددی مناسب نیست. زیرا می‌تواند مقادیر مختلفی داخل هر رکورد بگیرند. غیر از این محدودیت‌ها، مشکل دیگری که این الگوریتم‌ها دارند این است که در تولید قوانین، ترتیب آیتم‌ها نقش مهمی در آن‌ها بازی می‌کند [۳]. حال آنکه روش پیشنهادی در این مقاله دارای این محدودیت نیست.

در روشی که در این مقاله معرفی می‌شود، سعی می‌کنیم تا قوانین انجمنی را بر اساس درجه اطمینان، پشتیبان آن‌ها با استفاده از الگوریتم بهینه‌سازی جمعی ذرات (PSO) و با یک کدینگ بسیار ساده و سریع استخراج کنیم به گونه‌ای که قوانین تولید شده دارای طول مناسبی نیز باشند. به همین منظور در بخش بعدی، مقدمه‌ای کوتاه از الگوریتم PSO ارائه می‌کنیم.

۳ الگوریتم بهینه‌سازی جمعی ذرات (PSO)

الگوریتم PSO که توسط Kennedy و Eberhart پیشنهاد شده است [۱۰]، یک تکنیک بهینه‌سازی تصادفی بر مبنای جمعیت می‌باشد که از رفتارهای اجتماعی دسته پرندگان و ماهی‌ها الهام می‌گیرد. در الگوریتم PSO ابتدا سیستم با یک جمعیتی از جواب‌های تصادفی مقدار دهی اولیه می‌شود و سپس با به روز رسانی نسل‌ها جواب بهینه جستجو می‌شود. برخلاف روش مشابه، الگوریتم ژنتیک، این الگوریتم هیچ عملگر تکاملی مانند ادغام و جهش ندارد و از سرعت همگرایی بالاتری نسبت به آن برخوردار است. در PSO جواب‌های بالقوه که ذره نامیده می‌شوند، در کل فضای مسئله با دنبال کردن بهینه‌ترین ذره کنونی به حرکت در می‌آیند. اگر یکی از ذرات مسیر خوبی را بیابد، سایر ذرات به دنبال آن ذره حرکت می‌کنند، هرچند که از آن خیلی دور باشند. رفتار جمعی با استفاده از ذرات داخل فضای چندبعدی که دارای دو مشخصه‌ی مکان و سرعت هستند مدل می‌شود.

این ذرات در کل این فضا حرکت می‌کنند و بهترین مکانی را که تاکنون ملاقات کرده‌اند را به خاطر می‌سپارند. آن‌ها این موقعیت‌های خوب را به اطلاع یکدیگر رسانده و موقعیت و سرعت حرکت خود را براساس این موقعیت‌های خوب تنظیم می‌کنند. اگر بخواهیم این روند را به صورت دقیق‌تر بیان کنیم، مراحل زیر را خواهیم داشت: جمعیت ذرات در ابتدا با جمعیتی تصادفی از جواب‌ها مقاداردهی اولیه می‌شوند. این جمعیت اولیه به صورت تکراری در کل فضای جستجو d -بعدی حرکت می‌کنند و به دنبال جواب‌های جدید می‌گردند. برای هر ذره تابع برازندگی f به منظور اندازه‌گیری کیفیت جواب محاسبه می‌شود تا بهترین ذره مشخص گردد. هر ذره دارای یک مکان و یک سرعت است که به ترتیب توسط بردارهای مکان X_i (که i اندیس ذره می‌باشد) و سرعت V_i نشان داده می‌شوند. هر ذره بهترین مکان خود تا لحظه کنونی را در بردار $P_i = (p_{i1}, p_{i2}, \dots, p_{id})$ و مقدار j -امین بعد خود را در p_{ij} نگهداری می‌کند. بهترین بردار در میان همه‌ی ذرات در بردار P_g ذخیره می‌شود که مقدار j -امین بعد آن در p_{gj} قرار دارد. در طول زمان تکرار t ، بروزرسانی سرعت از سرعت قبلی به مقدار جدید توسط رابطه‌ی (۲) و مقدار جدید برای مکان از رابطه‌ی (۳) محاسبه می‌شوند.

$$v_{ij}(t+1) = wv_{ij}(t) + c_1r_1(p_{ij}(t) - x_{ij}(t)) + c_2r_2(p_{gj}(t) - x_{ij}(t)) \quad (2)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \quad (3)$$

که در آن w ضریب اینرسی نامیده می‌شود. r_1 و r_2 اعداد تصادفی هستند که برای حفظ چگالی جمعیت بوده و به صورت یکنواخت در بازه‌ی [۰-۱] توزیع شده‌اند. c_1 و c_2 ثابت‌هایی مثبت می‌باشند که به ترتیب ضرایب اجزای خود تشخیصی و اجتماعی نامیده می‌شوند. شبه کد زیر مراحل فوق را به صورت خلاصه نشان می‌دهد:

```

Initialize();
for t=1 to the limit of iterations
  for i=1:N
    Fitnessi(t) = EvaluationFitness (Xi(t));
    UpdateVelocity (Vi(t+1)) according to formula (2);
    LimitVelocity (Vi(t+1));
    UpdatePosition (Xi(t+1)) according to formula(3);
    if needed, update Pi and Pg;
  End
  Terminate if Pg meets problem requirements;
End

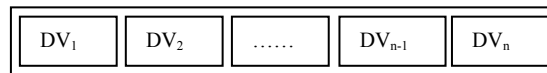
```

۴ روش پیشنهادی

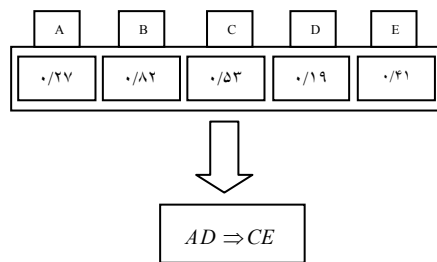
۴-۱ پیکربندی یک ذره

در این مقاله ما سعی می‌کنیم قوانین انجمنی را با استفاده از الگوریتم PSO بیابیم. بدین منظور اولین کاری که باید انجام گیرد نمایش دادن یا به عبارت بهتر کد کردن قوانین انجمنی داخل یک ذره است. در کدینگ به کار رفته در این مقاله، هر ذره شامل متغیرهای تصمیمی (DV) است که نماینده آیت‌ها می‌باشند. به این ترتیب

که آیت i -ام در مکان i -ام ذره قرار دارد. حال متغیر تصمیم i -ام در این ذره مشخص خواهد کرد که آیا خصیصه i -ام جزء خصیصه های قانون نمایش داده شده توسط این ذره است یا خیر و اینکه این خصیصه جزء بخش مقدم قانون است یا جزء نتیجه قانون. به این صورت که، اگر این متغیر مقداری در بازه $0.00 \leq DV_i \leq 0.33$ بگیرد به این معنی است که متغیر i -ام در بخش مقدم قانون خواهد آمد. و اگر مقداری در بازه $0.33 < DV_i \leq 0.66$ بگیرد، یعنی متغیر i -ام جزء قسمت نتیجه است و در نهایت اگر $0.66 < DV_i \leq 1.00$ باشد، آن متغیر در آن قانون ظاهر نخواهد شد. در نتیجه تمامی متغیرهای تصمیمی که در بازه مابین $0/۳۳$ و $0/۱۰۰$ مقدار بگیرند قسمت مقدم قانون را خواهند ساخت در حالی که متغیرهای دارای مقداری بین $0/۳۳$ و $0/۶۶$ ، قسمت نتیجه را می سازند. در شکل (۱) پیکربندی یک ذره جهت نمایش یک قانون را می بینیم و شکل (۲) مثالی از نحوه تبدیل کدینگ یک ذره به یک قانون را نشان می دهد.



شکل ۱. پیکربندی یک ذره جهت نمایش یک قانون



شکل ۲. مثالی از نحوه تبدیل یک ذره به یک قانون

همان طور که مشاهده می شود، در این روش نمایش طول ذره برابر خواهد بود با تعداد خصیصه های موجود در پایگاه داده.

۴-۲ طراحی تابع برازندگی

هدایت گر ذرات داخل جمعیت به سمت یک جواب بهینه با استفاده از تابع برازندگی می باشد. در اینجا به طور معمول فرض می شود که ذره با بزرگترین مقدار برازندگی، بهترین ذره می باشد. در روش پیشنهادی اگر فرض کنیم A و C به ترتیب مجموعه خصیصه های شرکت کننده در قسمت مقدم و نتیجه قانون باشند که با استفاده از دیکد کردن ذره متناظر، طبق روش بخش قبل به دست آمده اند تابع برازندگی را به صورت زیر تعریف می کنیم:

$$Fitness = \alpha_1 \times \frac{Num(AUC)}{DS} + \alpha_2 \times \frac{Num(AUC)}{Num(A)} - \alpha_3 \times NA \quad (۴)$$

که در این رابطه $\text{Num}(X)$ تابعی است که تعداد رکوردهای شامل تمامی خصیصه‌های مجموعه X را بر می‌گرداند، DS اندازه مجموعه داده است، NA تعداد خصیصه‌های شرکت کننده در قانون تولید شده بوده و ضرایب α به منظور کنترل تاثیر هر یک از پارامترهای داخل تابع برازندگی می‌باشند و بنابر نیاز کاربر می‌توانند به دلخواه تنظیم شوند.

همانگونه که دیده می‌شود، قسمت‌های اول و دوم این تابع به ترتیب مربوط است به محاسبه‌ی مقدار پشتیبان و درجه اطمینان قانون تولید شده. در نظر گیری این دو مولفه با هم در محاسبه برازندگی قانون تولید شده ضروری به نظر می‌رسد زیرا درجه اطمینان و یا پشتیبان به تنهایی نمی‌توانند معیاری برای قضاوت روی کیفیت قانون تولید شده باشند. بدیهی است قانونی از درجه کیفیت بالایی برخوردار است که این دو فاکتور با هم در آن از مقدار بالایی برخوردار باشند. از سویی دیگر می‌دانیم که در قوانین با طول زیاد احتمال وجود خصیصه‌های زائد که باعث کاهش کیفیت جواب تولید شده می‌شوند زیاد است. در نتیجه در قسمت سوم سعی می‌کنیم قوانین با طول نسبتاً کوتاه و در نتیجه با قابلیت خوانایی، درک و کیفیت بالاتر که در داده کاوی از اهمیت ویژه‌ای برخوردار است، تولید نماییم.

۵ نتایج شبیه سازی

ما روش پیشنهادی خود را روی چند مجموعه داده پیاده سازی و آزمایش کردیم که نتایج حاصل از آن‌ها بسیار راضی کننده بود. در این مقاله برای نمونه نتیجه کار روی یک مجموعه داده با ۴۰۰۰۰ رکورد از نوع تراکشنی که دارای ۲۹ خصیصه می‌باشد، نشان می‌دهیم. در پیاده سازی روش پیشنهادی، پارامترهای الگوریتم PSO را به صورت جدول (۱) می‌گیریم.

جدول ۱. پارامترهای الگوریتم PSO

پارامترها	اندازه جمعیت	تعداد تکرارها	نرخ‌های یادگیری $c2, c1$	تابع اینرسی	V_{max}
مقادیر	۲۵	۵۰۰	۲	۰/۹-۰/۴	۰/۵

ضرایب α_1, α_2 و α_3 مربوط به تابع برازندگی در این آزمایش نیز به ترتیب مقادیر ۰/۸، ۰/۸ و ۰/۰۵ گرفته‌اند. به علت بزرگی مجموعه داده برای بارگذاری در حافظه، ما در هر مرحله نمونه‌ای با اندازه مشخص را به صورت تصادفی انتخاب کرده و الگوریتم پیشنهاد شده در این مقاله را روی آن اجرا می‌کنیم. در هر اجرا ما شرط خاتمه را بر اساس بیشینه مقدار تعداد قوانین انجمنی تولید شده کلی در حین اجرا می‌گیریم. از میان قوانین تولید شده، قوانین تکراری و قوانین با درجه اطمینان زیر ۰/۸۵ و پشتیبان زیر ۰/۲۰ را حذف کرده و باقی مانده‌ها را به عنوان قوانین نهایی معرفی می‌کنیم. نتایج حاصل در جدول (۲) آمده است. در دو ستون آخر این جدول علاوه بر پارامترها و نتایج گفته شده، میانگین درجه اطمینان و مقدار پشتیبان قوانین نهایی داخل هر زیر گروه را نیز آورده‌ایم.

از روی مجموعه قوانین تولید شده برای نمونه‌های مختلف و با بیشینه قوانین مختلف، می‌توان مشاهده کرد که بعد

از حداکثر ۲۰۰ قانون بدون محدودیت تولید شده، الگوریتم قوانین چندان زیادی تولید نمی‌کند. به عبارت دیگر بعد از این تعداد الگوریتم تقریباً همگرا می‌شود. به عنوان مثال در مورد نمونه سوم تنها ۱ قانون بیشتر با قیمت سنگین ۱۰۰ بار اجرای اضافی تولید می‌شود. قابل ذکر است که به علت نرمال شدن داده‌های این مجموعه داده و وجود مقادیر زیاد صفر در آن، مقدار پشتیبان نسبتاً پایینی بدست آمده است زیرا همانگونه که می‌دانیم این مولفه شدیداً وابسته به نوع مجموعه داده‌ها می‌باشد. در حالی که روی یکسری داده‌های دست ساز این الگوریتم قوانینی با تقریباً حداکثر مقدار پشتیبان ممکن تولید کرده است که نشانگر کارایی بسیار بالای آن در استخراج این نوع قوانین می‌باشد.

جدول ۲. نتایج حاصل از آزمایش

اندازه نمونه	حداکثر تعداد قوانین	تعداد قوانین تولید شده	میانگین درجه اطمینان	میانگین مقدار پشتیبان	میانگین طول قوانین
۱۰۰۰	۱۰۰	۳۹	%۹۱	%۲۳	۱۴/۲
	۲۰۰	۴۸	%۹۳	%۲۳	۱۳/۶
	۳۰۰	۵۱	%۹۷	%۲۶	۱۱/۷
۱۰۰۰	۱۰۰	۴۱	%۹۴	%۲۲	۱۵/۱
	۲۰۰	۵۲	%۹۴	%۲۴	۱۵/۵
	۳۰۰	۵۴	%۹۶	%۲۵	۱۴/۶
۱۰۰۰	۱۰۰	۴۶	%۹۴	%۲۲	۱۲/۹
	۲۰۰	۶۷	%۹۵	%۲۵	۱۳/۳
	۳۰۰	۶۸	%۹۸	%۲۷	۱۱/۸
۲۰۰۰	۱۰۰	۶۱	%۹۶	%۳۰	۱۲/۳
	۲۰۰	۷۲	%۹۷	%۳۳	۱۱/۸
	۳۰۰	۷۴	%۹۹	%۳۵	۱۰/۱
۲۰۰۰	۱۰۰	۶۵	%۹۶	%۲۹	۱۲/۸
	۲۰۰	۷۱	%۹۹	%۳۴	۱۱/۹
	۳۰۰	۷۱	%۹۹	%۳۶	۱۱

۶ نتیجه‌گیری

کشف قوانین انجمنی در یک مجموعه داده بزرگ یک عمل زمان بر و در عین حال بسیار مهم با کاربردهای متنوع و وسیع است. ما در این مقاله سعی کردیم از ویژگی‌های الگوریتم‌های تکاملی به خصوص سرعت همگرایی بالای الگوریتم PSO برای کاوش قوانین انجمنی استفاده کنیم و راهکاری را بر این اساس پیشنهاد دادیم. الگوریتم پیشنهادی را روی چند مجموعه داده آزمایش کردیم که نتایج حاصل حاکی از موفقیت خوب روش پیشنهادی در عمل بود. ما در این مقاله از درجه اطمینان، مقدار پشتیبان و طول قانون تولید شده به عنوان برازندگی الگوریتم PSO استفاده کردیم، حال آنکه استفاده از پارامترهای دیگر مانند درجه جالبی و استفاده از

الگوریتم PSO از نوع پارتو می‌تواند بهتر عمل کرده و قوانین قوی‌تری تولید نماید. از سوی دیگر ما در قسمت آزمایش از نمونه‌های به تصادف انتخاب شده از مجموعه داده اصلی استفاده کردیم ولی از آنجایی که استفاده از نمونه خوب صحت قوانین تولید شده را افزایش می‌دهد، می‌توان از روش‌هایی مانند خوشه‌بندی جهت انتخاب نمونه استفاده کرد. ما این ایده‌ها را به عنوان موضوعی برای کارهای آتی در این زمینه پیشنهاد می‌کنیم.

منابع

- [1] Tan, P. N., Steinbach, M., Kumar, V., (2005). Introduction to Data Mining. Addison Wesley.
- [2] Adamo, J. M., (2001). Data Mining for Association Rules and Sequential Patterns, Springer-Verlag, New York.
- [3] Agrawal, R., Srikant, R., (1994). Fast algorithms for mining association rules, in: Proceedings of the 20th International Conference on Very Large Databases, Santiago, Chile.
- [4] Ayad, A. M., (2000). A new algorithm for incremental mining of constrained association rules. Master Thesis, Department of Computer Sciences and Automatic Control, Alexandria University.
- [5] Hipp, J., et al., (2000). Algorithms for association rule mining-a general survey and comparison, SIGKDD Explorations 2.
- [6] Agrawal, R. T., Imielinski, Swami, A., (1993). Mining association rules between sets of items in large databases,” Proc. of ACM SIGMOD Conf. on Management of Data, pp.207-216.
- [7] Freitas, A. A., (2001). Understanding the crucial role of attribute interaction in data mining, Artificial Intelligence Review 16 pp.177-199.
- [8] Lin D. I., Kedem, Z. M., (1998). Pincer-search: an efficient algorithm for discovering the maximal frequent set, Proc. of 6th European Conf. on Extending Database Technology.
- [9] Brin, S., et al., (1997). Dynamic itemset counting and implication rules for market basket data, Proc. of ACM SIGMOD International Conf. on Management of Data. Tucson, AZ.
- [10] Kennedy, J., Eberhart, R. C., (1995). Particle Swarm Optimization, Proc. of IEEE In Int. Conf. on Neural Networks, pp. 1942-1948, Piscataway, NJ.