時系列トランザクションに対するリアルタイム相関ルール抽出方式の提案

1T－5

西山 智　小野 智弘　小花 貞夫
株式会社 KDD 研究所

1. はじめに

相関ルール抽出は，トランザクションの集合からなるデータベースから新たな知識を獲得するデータ発掘の一分野であり，Apriori[1], DHP[2], FUP[3]等多くのアルゴリズムが提案されている．しかし，これらのアルゴリズムは予め与えられたトランザクションデータベースに対して相関ルールを抽出するものであり，時々刻々と発生するトランザクション (以下時系列トランザクションと呼ぶ) の傾向 (特定の期間に成立する相関ルール) をリアルタイムで計算するものではない．一方，実世界においては，時系列トランザクションの傾向をリアルタイムで計算できれば，網管理において障害発生や外部要因によるサービス需要の急増等を把握できる等，その有用性は高い．これらのアルゴリズムのうち FUP はデータベースの差分がある場合，それを用いて相関ルールの変化差分のみを計算するため，相関ルール抽出を最初から計算しなおす Apriori, DHP よりも計算量が少ない．しかしながら，FUP を時系列トランザクションの相関ルールに対するリアルタイム抽出に適用するには，新たにルール候補が発生する場合に全データベースのスキャンが必要になる等の問題がある．そこで本稿では，時系列トランザクションに対するリアルタイム相関ルール抽出における FUP の問題点を解析し，それらを解決する抽出方式について提案する．

2. 問題領域の定義

ここでは議論を簡単にするため，離散値を対象とした相関ルール抽出に限定して述べる．アイテムの集合を $I = \{i_1, i_2, ..., i_m\}$ とする．トランザクション t は時々刻々発生するアイテム集合であり，$t = (ts, TID, t_{data})(t_{data} \subseteq I, ts$ は発生時刻，TIDはトランザクションの識別子) の型をもつ．またトランザクション全体の集合を $D = \{t_1, t_2, ..., t_n\}$ とする．ここで，相関ルール抽出の対象期間を Tとすると，トランザクションの集合 Dの直近の期間 Tにおける部分集合 D_{now}^{now-T}において，アイテム集合 Xの支持度 $support(X)$ は D_{now}^{now-T} に対して Xを含むトランザクションの割合を示す．相関ルール $X \Rightarrow Y : (X, Y \subset I, X \cap Y = \phi)$ は，アイテム集合 X が成立するならば，確信度 $confidence(X \Rightarrow Y) = support(X \cup Y)/support(X)$ でアイテム集合 Y が成立することを示し，このルールは D_{now}^{now-T} において支持度 $support(X \Rightarrow Y) = support(X \cup Y)$ の確率で支持される．この時，D_{now}^{now-T}において，要求する最小支持度 $minS$, 最小確信度 $minC$ を満たす相関ルールをリアルタイムに抽出する事を問題領域とする．

3. FUP の概略とリアルタイム処理における問題点

3.1 FUP の概略

FUP は，Apriori, DHP と同様に，長さ 1 のラージアイテム集合から計算を開始し，より長いラージアイテム集合を順次求める．長さ i のラージアイテム集合を L_i, 元および更新差分のデータベースをそれぞれ DB, db とすると，例えば長さ 1 の場合以下の処理を行う．

(1) 全アイテム集合 $X \in L_1$について，出現頻度 $Support(X)_{DB+db} < minS \times (size(DB) + size(db))$ を満たさないものを取り除き，L_1'とする．

(2) 同時に，新たに発生したトランザクションに含まれるアイテムについて，L_1に含まれないアイテム集合を候補集合 C_1として作成し，そこから $Support(X)_{db} < minS \times size(db)$ であるアイテム集合を除き，残りを C_1'とする．

(3) DB をスキャンし，全アイテム集合 $Y \in C_1'$について $Support(Y)_{DB+db} < minS \times (size(DB) + size(db))$ を満たす Yを L_1'に加え，新しい L_1を作る．

3.2 リアルタイム処理の問題点

FUP を本稿で述べるような問題領域に適用する場合以下の問題点がある．

(1) 差分により新たに候補集合 C_i'が発生した場合，対象データベース D_{now}^{now-T}のスキャンが必要である．

(2) 差分のトランザクション数がある程度大きい (例えば数百) ことを利用して，C_iから C_i'を絞り込むが，リアルタイムの抽出では差分の大きさは常に 1 であり絞り込みが機能しない．

(3) 対象期間の概念がなく，全データベース通算での相関ルールのみしか計算できない．

(4) L_iから L_i'を求めるために L_iを全てアクセスする．

4. リアルタイムアルゴリズムの提案

4.1 概略

提案方式は，相関ルールの差分抽出を行う点で FUP と類似であるが，FUP の問題点を以下の通り解決する．

(1) アイテム集合候補として可能性の高い長さ 1, 2 のアイテム集合については，全ての組合せを数えあげ，維持する．さらに例えば長さ 2, かつ $Support(X) < n \times minS + \alpha$以下のアイテム集合について，アイテム集合毎に全ての該当トランザクションの TID

"On a real-time mining algorithm for association rules from online transactions" by Satoshi NISHIYAMA, Chihiro ONO and Sadao OBANA, KDD R&D Labs., Inc.

を記憶し, 新たに長さ3以上で候補対象となったアイテム集合を再計算する際に, 該当トランザクションを直接参照可能とする. 図1にアイテム種類数4, $minS = 0.3$ の場合の内部状態の例を示す. ここで, 図中の (X,Y) で示す長さ2のアイテム集合の左下半分のハッチは冗長なので不要であることを示している. また, アイテム集合 (i_1, i_4) は $TID = t_7$ を記憶している. これにより, データベーススキャンが必要となるのは, 長さ3以上の候補の計算で $Support(X)$ が一度 $n \times minS + \alpha$ となり, その後 $n \times minS$ を下回り, 再度これを上回った場合のみとなる. なお, α は $n \times minS$ 近辺でのスキャンの繰り返しを防止するための定数である.

(2) 上記 (1) の全数数え上げにより差分を用いた絞り込みは殆ど効果がなくなるため, 行わない.

(3) 対象期間の概念を入れるために, トランザクション t 発生時に, 対象期間 T 後に t を削除する仮想的なトランザクション t^- を追加する. T 後にこの t^- を処理する事により対象期間から外れたトランザクションは自動的に計算から除外され, 常に D_{now}^{now-T} に対するラージアイテム集合を維持する.

(4) トランザクション t, t^- の発生により対象期間中のトランザクション件数 $n = size(D_{now}^{now-T})$ は変化する. L を全て参照することなしに L' を求めるために, 維持すべき各アイテム集合を $Support(X)$ の値の順に常にソートしておく. (図1左側) さらに $Support(X)$ が $n \times minS$ の値の境界にあるアイテム集合を保持しておき, トランザクション発生時には新たな n' による $n' \times minS$ へ境界を移動することで L' を求める. 図1では, $n = size(D_{now}^{now-T})$ が1増加することで境界が変動し, L からハッチ部分のアイテム集合を除いた集合が L' となる.

4.2 アルゴリズムの複雑さ

多数のトランザクションから相関ルールを抽出するため, リアルタイム処理ではトランザクション件数に対する計算量を減らす事が必要である.

4.2.1 記憶領域について

長さ1, 2のアイテム集合を全て数え上げ, 維持するための領域は $O(m^2)$: (m はアイテムの種類) である. 長さ3以上についてはアイテム集合候補は急激に減少することが知られている事[4]と, 最悪でも候補アイテム集合数 $\leq$ ($c \times$ 最大のアイテム集合の長さ)$/minS$: (c はトランザクション中の平均アイテム数) が成立するので, 全体として $O(m^2)$ が成立し, 期間中のトランザクション件数 n には影響されない. 解決法の (1) で示した, 一定出現度以下のノードの TID の記憶に必要な領域は, 最悪 $O(n \times m^2)$: (n は対象期間 D_{now}^{now-T} におけるトランザクション件数) となり n が影響するが, 実際には出現頻度が少ない ($n \times minS + \alpha$ 以下) のアイテム集合に限っているため実現可能な範囲に収まると考える.

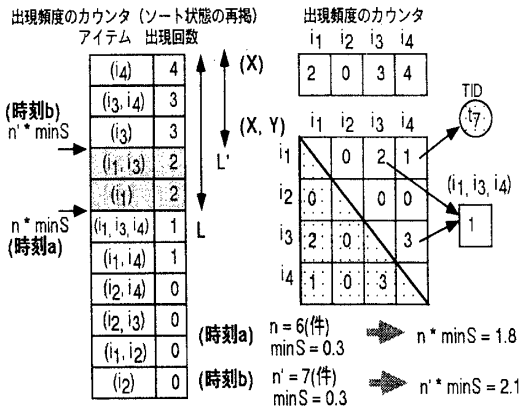

図 1: アイテム種類数 4 の場合の内部状態の例

4.2.2 処理時間について

トランザクション毎に長さ1, 2のアイテム集合の計数を常に行うための時間は $O(c^2)$ である. 長さ3以上の候補を計算するコストは, 領域の場合と同様に候補が急減するので全体として $O(c^2)$ が支配的であると考えられる. また, アイテム集合を $Support(X)$ の値でソート状態を維持する時間は $O(\log m)$ 以下である. 提案方式では, 長さ3以上のアイテム集合の候補計算の際, 4.1節 (1) で述べた場合にデータベーススキャンが必要となり, その時間は $O(n)$ である. しかし, 例えば $n \times minS + \alpha$ を下回った時点で予めバックグラウンドでデータベーススキャンを行い TID を再記憶する等の手法により, 実質的にトランザクション発生時のデータベーススキャンを回避できる.

5. おわりに

本稿では, 時系列トランザクションに対して直近の一定期間に成立する相関ルールをリアルタイムで求めるアルゴリズムについて提案した. 長さ2までのアイテム集合については全数数え上げ, また $Support(X)$ の少ないアイテム集合は TID を記憶する事, 等により, ほぼトランザクションデータベースをスキャンせずに相関ルールを求めることができる. 今後, 提案方式の評価を行っていく予定である. 最後に日頃御指導頂く (株)KDD 研究所 村谷拓郎所長, 鈴木健二副所長に感謝します.

参考文献

[1] R, Agrawal, et. al.: "Fast Algorithms for Mining Association Rules," Proc. of the 1994 VLDB, (1994).
[2] J. S. Park, et. al.: "An Effective Hash Based Algorithm for Mining Association Rules," Proc. of the 1995 ACM SIGMOD, (1995).
[3] D. W. Cheung, et. al.: "Maintenance of Discovered Association Rules in Large Databases: An Incremental Updating Technique, " Proc. of the 1996 ICDE, (1996).
[4] 喜連川: "データマイニングにおける相関ルール抽出技法", 人工知能学会誌, Vol12, No.4, (1997).