時系列トランザクションに対する実時間相関ルール抽出方式の設計と評価

3U－9　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手法である．筆者らはこれまでに，網管理における障害発生や外部要因によるサービス需要の急増等の把握を目的として，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) を実時間で抽出することを提案した[1]．本稿では，この提案に基づく相関ルール抽出方式の設計と評価について報告する．

2. 実時間相関ルール抽出の定義

ここでは離散値を対象とした場合を示す．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，$t = (ts, t_{data})(t_{data} \subseteq I, ts$ は発生時刻) の型をもつ．またトランザクション全体の集合を $D = \{t_1, t_2, ..., t_z\}$ とする．相関ルール抽出の対象期間を Tとすると，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 $sup(X)$ 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，X が成立するならば，確信度 $conf(X \Rightarrow Y) = sup(X \cup Y)/sup(X)$ で 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 $sup(X \Rightarrow Y) = sup(X \cup Y)$ 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実時間で抽出する事を目的とする．

3. 設計

アイテム集合 Xについて $sup(X) \geq minS$を満たす Xの集合をラージアイテム集合と呼ぶ．ラージアイテム集合において長さ 3 以上のアイテム集合の数は急速に収束するとされている[2] 事から，ラージアイテム集合に含まれるアイテム集合およびラージアイテム集合になる可能性の高いアイテム集合の全てについてトランザクション毎に支持度を数え上げ維持する．

3.1 データ構造

以下の 2 種類のノードを設け，各ノードは対応するアイテム集合 X ($X \subseteq I$)，支持数 $Sup(X)$，最終更新時刻を管理する．$Sup(X)$ は D_{now}^{now-T}での X の出現回数であり，期間 Tに含まれるトランザクション数を n とすると $sup(X) = Sup(X)/n$ の関係にある．

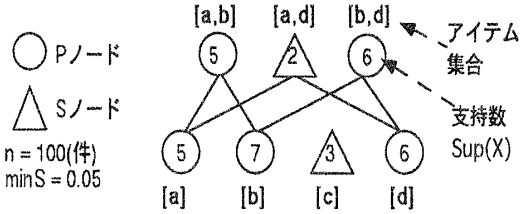

図 1: アイテム種類数 4 の場合のノードの例

- P(rimary) ノード:
  ラージアイテム集合に含まれるアイテム集合を示すノード．長さ 0 のアイテム集合は実際には存在しないが仮想的に P ノードであるとみなす．
- S(upplement) ノード:
  ラージアイテム集合には含まれないが，そのアイテム集合から任意の 1 アイテムを取り除いたアイテム集合が全てラージアイテム集合に含まれる (P ノードである) アイテム集合を示すノード

図 1にこれらのノードの例を示す．

トランザクションが発生する毎に，或は時間の経過によりトランザクションが対象期間から外れる毎にこれらのノードの支持数が変化するため，それに応じて動的にノードの作成/削除および種別の変更を行う．各ノードへのアクセスを高速化するため，アイテム集合をキーとする索引 (本設計ではハッシュ表) をノードに対して設ける．またノードを支持数の順に常にソートし，支持度が $minS$を新たに上回った，或は下回ったノードを容易に検出できるようにした．

3.2 データ構造への操作

3.2.1 トランザクション発生時の処理

トランザクションが発生した場合以下の処理を行う．ここでは，あるアイテム集合のノード (低次ノード P_L) とそれに任意の 1 アイテムを追加したアイテム集合のノード (高次ノード P_H) の関係で示す．以下の各項番は表 1での状態に対応している．

(1) 発生前に $conf(P_L \Rightarrow P_H) < minC$で発生後に $conf(P_L \Rightarrow P_H) \geq minC$であれば $P_L \Rightarrow P_H$ を新たな相関ルールとして追加する．

(2) (2a) 支持度の低下により P_HがS ノードになった場合，相関ルール $P_L \Rightarrow P_H$を削除し，P_Hの高次ノードを全て削除する．(2b) また $conf(P_L \Rightarrow P_H) < minC$であるならば $P_L \Rightarrow P_H$を削除する．

(3) (2a) の場合と同様に P_HがS ノードとなった場合，相関ルール $P_L \Rightarrow P_H$を削除し，P_Hの高次ノード

"Design and evaluation of a real-time mining algorithm for association rules" by Satoshi NISHIYAMA, Chihiro ONO and Sadao OBANA, KDD R&D Labs., Inc.

表 1: トランザクション発生時の状態

状態	低次ノード P_L種別	高次ノード P_H種別	$Sup(P_L)$	$Sup(P_H)$	$sup(P_L \Rightarrow P_H)$	$conf(P_L \Rightarrow P_H)$
(1)	P	P	+1	+1	↑	↑
(2)	P	P	+1	0	↓	↓
(3)	P	P	0	0	↓	→
(4)	P	S	+1	+1	↑	↑
(5)	P	S	+1	0	↓	↓
(6)	P	S	0	0	↓	→

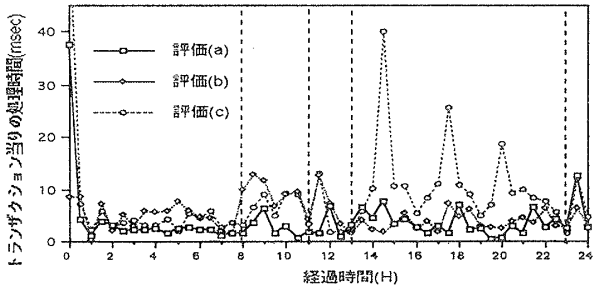

図 2: トランザクション当りの平均処理時間

を全て削除する.

(4) P_Hが支持度の上昇によりPノードとなった場合, (1) と同じ処理を行う.

(5) 特に処理無し

(6) P_Lが支持度の低下によりSノードになった場合, P_Lの高次のノードを削除する.

3.2.2 トランザクションの打ち消し処理

発生から対象期間Tが経過すると, トランザクションは計算対象外となる. この時トランザクションの影響を打ち消す処理を行う. この処理はトランザクションの発生とほぼ逆の処理であるためここでは省略する.

3.2.3 トレーニング処理

提案方式は, 起動直後のトランザクション数が少ない状態で一時的に冗長なノードを作成し, その後削除されて安定状態となる. これを回避する手法として, ここでは最初の一定数 ($1/minS$以上) のトランザクションを受け付けるまでノードの維持を行わないようにする.

4. 評価

提案方式を実装し, Sun Server1000 (CPU:40MHz, MM:1GB) を用いて評価を行った. ランダムに選択した相関関係を予め設定した試験データを入力として, 相関ルールの抽出を確認した. 評価に用いたパラメータを表2に示す. 評価の結果, 作成した相関ルールは100%検出できたが概ね30分程度 ($T/2$) 検出が遅れた.

図2および図3にトランザクション当りの処理に必要な時間 (測定点付近の10トランザクションの平均) と内部に維持されているノード数を示す. 図で縦線は負荷 (および相関ルールの半数) を変更した時点を示す. 処理時間については, トレーニング終了直後 (0Hの時点) はノード数の変動が激しく評価 (c) では平均64.5msec

表 2: 評価パラメータ

a) 入力データのパラメータ

パラメータ	評価 (a)	評価 (b)	評価 (c)
アイテムの種類 m	512	1024	1024
平均アイテム数 i	6	6	8
相関ルール最高次数	3		
相関ルール数	最高次のノード5種類分, 負荷変動毎に半数を入れ替え		
相関ルール最高次での支持度	0.04		
相関ルールでの確信度	0.6～1.0		
データ期間	1日		
トランザクション発生量	時間変動 (8,11,13,23時に変更. 毎秒0.2件から1件)		

b) 抽出時のパラメータ (全ての例で固定)

最小支持度 $minS$	0.02
最小確信度 $minC$	0.6
抽出対象期間 T	60分

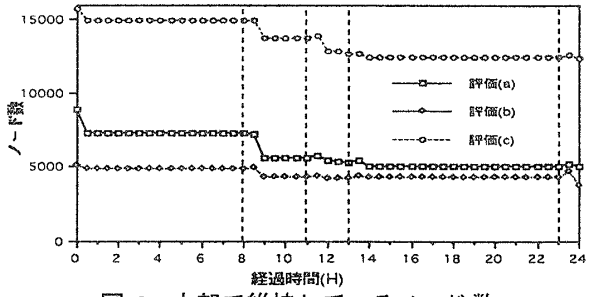

図 3: 内部で維持しているノード数

を要したが, その後は10msec以下程度に収まった. 相関ルールの変更時以降に処理時間の増加が見られるが, ノードの追加削除の発生によるものと考えられる. また, 評価 (c) で14時間経過以降3回にわたり平均処理時間が長い場合があるが, この時は測定点にアイテム数17-18程度のトランザクションが含まれていた.

ノード数については, 相関ルール変更時に若干の増加が見られたものの, 徐々に減少してほぼ一定値となり, 安定している事が確認できた. 評価 (a) が (b) よりノード数が多い原因は (a) の入力データでの各アイテムの出現比率が (b) よりも高いためと考えられる. 評価 (c) は評価 (b) の概ね3倍程度で, これは入力データが3次の相関ルールを含むことでSノードまで考えると $O(i^4)$ 程度の記憶領域を必要とするためと考えられる.

5. おわりに

本稿では, 時系列トランザクションに対する実時間相関ルール抽出方式の設計と評価について報告した. 評価の結果, 実用的な処理時間で正しく相関ルールが抽出できることを確認した. 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] 西山 他: "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", 第58回情処全大 1T-05,(1999).

[2] 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).