データマイニングにおける相関関係抽出の並列処理方式

2P－5

新谷 隆彦, 喜連川 優
東京大学生産技術研究所

1 はじめに

近年のプロセッサの高性能化、二次記憶装置の大容量化等の計算機技術の進展により、莫大な履歴データが蓄積され、それらのデータを解析することが可能となった。このような大量に蓄積された履歴データを解析することにより、その中に埋もれた法則や関係など有用な情報を抽出し、データの効果的な活用を図る「データマイニング (Database Mining)」の研究が着目されている。

データマイニングで得られる情報の代表的なものに、データ間の相関関係があり、これを抽出する効率の良い手法の研究が進められている [1]。大量のデータを操作対象とする場合には、より根本的な性能向上が不可欠であり、我々は並列処理方式について研究を進めている [2]。我々の知る限り並列化に関する報告は、ほぼ同時期に発表された [1] しかなく、3章で述べる様に [1] は候補アイテム集合を全プロセッサに複製しており、記憶効率が著しく低い。本報告では、データ複製を行わず記憶効率を大幅に向上させ、より大規模なマイニングを可能とすると同時に、性能向上を実現する方式を提案するとともに、並列計算機 AP1000 上に実装し、その有効性を明らかにする。

2 相関関係の定義と抽出方法

アイテムと呼ばれる属性の集合を $\mathcal{I}$、トランザクションデータベースを $\mathcal{D}$ とし、その各要素はトランザクションの識別子とアイテムの集合である。相関関係は $X \Rightarrow Y$ で表現され、$X, Y \subset \mathcal{I}$, $X \cap Y = \emptyset$ であり、サポート値 (support)、コンフィデンス値 (confidence) の2つの値を伴う。アイテム集合 X のサポート値 $sup(X)$ は $\mathcal{D}$ のうち X を含む割合を表し、相関関係 $X \Rightarrow Y$ のコンフィデンス値は $\mathcal{D}$ の中で X を含むトランザクションのうち、X と Y を共に含むトランザクションの割合を表し、$sup(X \cup Y)/sup(X)$ で定義される。

相関関係の抽出はユーザが定義した最小サポート値と最小コンフィデンス値を満足する全ての相関関係を見つけ出すことになり、次の2段階の処理で求める。

1. 最小サポート値を満足するアイテム集合 (頻出アイテム集合) を全て取り出す。
2. "1" で求めた頻出アイテム集合を用いて最小コンフィデンス値を満足するルールを作成する。

"1" の処理はアイテムの数、トランザクション量が多い場合に負荷の高い処理となるため、効率の良い処理方法に関する研究がいくつか行われている。以下に [3] で提案された Apriori アルゴリズムを示す。3節で提案する並列アルゴリズムは Apriori アルゴリズムを基にしている。

はじめに、データベースをスキャンし、各アイテムの生起回数を数え上げ、サポート値を求め、最小サポート値を満足するものを取り出して長さ1の頻出アイテム集合とする。次に長さ1の頻出アイテム集合から2つのアイテムの組合せを作成する。これらを長さ2の候補アイテムの集合と呼ぶ。データベースをスキャンしてサポート値を求め、長さ2の頻出アイテム集合を取り出す。以降、長さ (k-1) の頻出アイテム集合から、長さ k の候補アイテム集合を作成し、長さ k の頻出アイテム集合を求める処理 (長さ k のパス) を繰り返す。データベースをスキャンしてサポート値を求め、最小サポート値を満足するものを取り出し、長さ k の頻出アイテム集合とする。

3 並列処理方式

論文 [1] の方式では、各プロセッサが全ての候補アイテム集合をコピーして保持できることを仮定しており、並列化は容易であるが、大規模なトランザクションデータの処理を考慮すると、多くの場合にこの仮定は成立しない。本稿では、全ての候補アイテム集合が単一プロセッサの主記憶上に入り切らない場合の考察を進める。候補アイテム集合はプロセッサ間に分割して割り当てる。簡単のため、候補アイテム集合は全てのプロセッサの主記憶の総量より小さいとする。

3.1 Naive Partitioned Apriori : NPA

NPA では候補アイテム集合をプロセッサ間に均等になるように、無作為に分割する。長さ k のパスを以下に示す。

1. 長さ (k-1) の頻出アイテム集合を用いて、長さ k の候補アイテム集合を作成し、プロセッサ間に分割して割り当て、プロセッサ毎に割り当てられた候補アイテム集合のみ主記憶上のハッシュ表に挿入する。
2. ローカルディスクからトランザクションデータベースを読み出し、他の全てのプロセッサに放送する。同時に、トランザクションから k 個のアイテムの組合せを作成し、ハッシュ表を検索する。対応する候補アイテム集合が存在する場合、その生起回数を1増加する。他のプロセッサから送信されたトランザクションに対しても同様の処理を行う。
3. 全てのトランザクションに対する処理が終了した時点で、プロセッサ毎に頻出アイテム集合を決定し、他の全てのプロセッサに放送する。これにより、全てのプロセッサが長さ k の頻出アイテム集合を持つことになる。ここで、頻出アイテム集合の数は、候補アイテム集合と比較して非常に少ないため、この様な処理が可能となる。

このアルゴリズムは単純であり、実装が容易であるが、全てのトランザクションデータを他の全てのプロセッサに放送するため、通信量が莫大になる。

3.2 Hash Partitioned Apriori : HPA

HPA は候補アイテム集合をハッシュ関数を用いてプロセッサ間に分割する。これにより、全てのトランザクションを放送する必要がなくなる。長さ k のパスを以下に示す。

1. 長さ (k-1) の頻出アイテム集合を用いて、長さ k の候補アイテム集合を作成する。これにハッシュ関数を適用し、対応するプロセッサの識別子を求め、自分の識別子と等しい場合に、主記憶上のハッシュ表に挿入する。
2. ローカルディスクからトランザクションデータベースを読み出す。各トランザクションから k 個のアイテムの組合せを作成し、"1" と同一のハッシュ関数を適用する。ハッシュ値に対応するプロセッサの識別子を求め、その

Parallel Algorithms for Mining Association Rules
T.Shintani and M.Kitsuregawa
Institute of Industrial Science, University of Tokyo
Roppongi 7-22-1, Minato-ku, Tokyo, 106 Japan

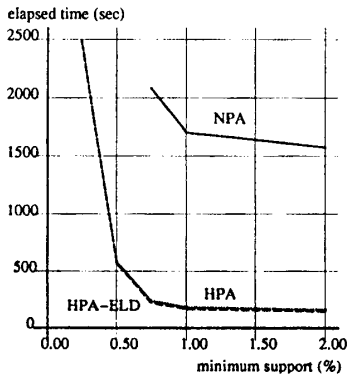

図 1: 各アルゴリズムの処理時間

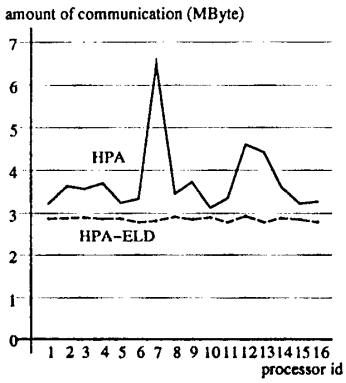

図 2: 各プロセッサの受信量

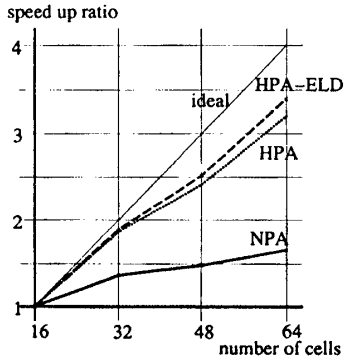

図 3: 台数効果

プロセッサにアイテムの組合せを送信する。他のプロセッサから送信されたメッセージに対して、ハッシュ表を検索し、対応する候補アイテム集合の生起回数を 1 増加させる。

3. NPA と同様

3.3 HPA with Extremely Large Itemset Decomposition : HPA-ELD

トランザクションデータに偏りがある場合、つまり、極端に多くのトランザクションに含まれるアイテム集合が存在する場合、そのアイテム集合を割り当てられたプロセッサに他のプロセッサからのメッセージが集中し、処理の偏りが生じる。HPA-ELD では、この影響を減少する方法をとる。

HPA-ELD では、生起回数が極端に大きいと予想される候補アイテム集合を全てのプロセッサにコピーし、プロセッサ毎に生起回数を調べる。全てのトランザクションデータに対する処理が終了した時点で、全プロセッサでの生起回数の総和を求め、サポート値を求める。

長さ (k-1) の頻出アイテム集合から長さ k の候補アイテム集合を作成するとき、元にした頻出アイテム集合のサポート値の和が閾値以上の場合、全てのプロセッサにコピーして割り当てる。他の候補アイテム集合は HPA と同様に、ハッシュ関数を適用して分割する。

4 性能評価

前節で述べた並列アルゴリズムを富士通社製分散メモリ型並列計算機 AP1000DDV[4] 上に実装し、性能測定を行った。今回の性能測定では 64 台のプロセッサ (SPARC(25MHz),DRAM(16MB)) の構成のシステムを利用し、各プロセッサには DDV(分散ディスクビデオ) ボードで 1GB のローカルディスクが 1 台接続されている。3 つの並列アルゴリズムの性能を評価には、[3] で述べられた手順を基に、トランザクション量を 2048K、1 つのトランザクションの平均アイテム数を 15、頻出アイテム集合の平均アイテム数を 4 として作成したトランザクションデータセット (ファイルサイズ 145MB) を用いた。

4.1 各アルゴリズムの処理時間

図 1 に各アルゴリズムの最小サポート値を変化させた場合の処理時間の変化を示す。ここでは、プロセッサ数 16 台 (4×4) の構成を用い、トランザクションデータファイルはプロセッサ間でほぼ均等になるように分割して、各プロセッサのローカルディスクに割り当てた。

結果から、HPA、HPA-ELD と比較して NPA の処理性能が著しく劣っていることが分かる。NPA では他の全てのプロセッサにトランザクションデータを放送するため、対応するプロセッサにのみ送信する HPA、HPA-ELD と比較して総通信量が非常に多くなる。また、HPA、HPA-ELD ではトランザクションから作成したアイテムの組合せを対応するプロセッサに送信するため、NPA と比較して総送信量が少なくなる。このアイテムの組合せの総数は処理対象とするデータに依存している。ここで、HPA-ELD で一部の候補アイテム集合を全プロセッサにコピーしてサポートを調べる処理は処理負荷が最大となる長さ 2 のパスでのみ行った。

図 2 に HPA、HPA-ELD の長さ 2 のパスにおける各プロセッサの受信量を示す。HPA-ELD により、プロセッサ間の負荷の偏りを取り除くことが出来ている。

4.2 台数効果

図 3 にプロセッサ数を変化させた場合 (16,32,48,64 台) の結果を示す。ここで、最小サポート値 0.75% とし、全プロセッサのトランザクション量の総量を一定とした。また、グラフは 16 台のプロセッサによる処理時間で正規化してある。

HPA は NPA と比較して良い台数効果が得られている。特に、HPA-ELD は負荷の偏りを取り除くことによってより良い台数効果が達成されていることがわかる。

5 まとめ

本論文では分散メモリ型並列計算機環境における相関関係抽出の並列処理方式を提案し、実際に並列計算機上に実装し、性能評価を行い、提案した並列処理方式により比較的良い台数効果が得られることを示した。

謝辞 AP1000 を利用する場を提供頂いた富士通並列処理研究センターの方々に深く感謝致します。

参考文献

[1] J.S.Park, M.-S. Chen, and P.S.Yu: "Efficient Parallel Data Mining for Association Rules", *CIKM'95*, pp.31-36, Nov 1995.

[2] 新谷, 喜連川: "Consideration on Parallelization of Database Mining", 電子情報通信学会データ工学研究会 (CPSY 95-88), 信学技法 Vol.95, No.47, pp.57-62, Dec 1995.

[3] R.Agrawal, and R.Srikant: "Fast Algorithms for Mining Association Rules", *Proc. of the 20th VLDB*, Sep 1994.

[4] 清水, 堀江, 石畑:"AP1000 の性能評価 - メッセージハンドリング、放送、バリア同期の効果 -", SWoPP'92(92-ARC-95), 情報処理学会研究報告 Vol.92, No.64, 1992.