分類階層を考慮した並列データマイニング処理方式

3R－3

新谷 隆彦，喜連川 優
東京大学生産技術研究所

1 はじめに

相関関係の抽出を目的としたデータマイニングが近年注目されている。我々は、その処理性能の向上を目的とした並列処理方式について提案してきた [1, 2]。本稿では、更にデータの分類階層を導入した相関関係抽出の並列処理方式について報告する。分類階層を考慮することにより、より一般化されたルールの抽出が可能となる [3]。一方、その処理負荷は更に増大することとなる。HPA なるハッシュを用いた並列処理方式を提案すると共に、分散メモリ型並列計算機上に実装し、その有効性を明らかにする。

2 分類階層を考慮した相関関係

データの分類階層を $\mathcal{T}$ とし、図1に示す木構造とする。$\mathcal{T}$ の要素をアイテムと呼び、集合 $\mathcal{I}$ とする。$\mathcal{T}$ の辺はアイテム間の階層を示し、アイテム A から C への辺がある場合、A を C の上位アイテム($ancestor(C)$) と呼ぶ。

図1: データの階層構造

トランザクションデータベース $\mathcal{D}$ はトランザクション T の集合とし、$T \subseteq \mathcal{I}$ である。また、アイテムの組合せ (アイテム集合) X がトランザクション T 内のアイテムまたは T 内のアイテムの上位アイテムで構成される場合、T は X を含むと表現する。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、Y は $ancestor(X)$ を含まない。相関関係は、サポート値、コンフィデンス値の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 を含むトランザクションの割合を表す。相関関係 $X \Rightarrow Y$ のサポート値 $sup(X \Rightarrow Y)$ は X と Y を共に含むトランザクションの割合 ($sup(X \cup Y)$) となる。また、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。ここで、Y が X の上位アイテムである相関関係 $X \Rightarrow ancestor(X)$ はコンフィデンス値が常に 100% となり、冗長なルールである。したがって、Y に X の上位アイテムを含む相関関係は考慮する必要がない。

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る。その処理は、まず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) を全て取り出し、それ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。ここで、頻出アイテム集合を求める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、効率の良い処理方式の研究が進められている。

Parallel Database Mining Algorithm with Classification Hierarchy
T.Shintani and M.Kitsuregawa
Institute of Industrial Science, University of Tokyo
Roppongi 7-22-1, Minato-ku, Tokyo, 106 Japan

3 並列処理方式

本節では分類階層を考慮したで提案した相関関係抽出の並列処理方式 (NPA,HPA) について述べる。

単一ノードの主記憶上に全ての候補アイテム集合を保持できる場合、並列化は容易であるが、データの分類階層を考慮した場合には更に多くの候補アイテム集合について調べなければならないため、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ノードの主記憶上に入り切らない場合を想定し、並列処理方式を提案する。また簡単のため、候補アイテム集合は全てのノードの主記憶の総量より小さいとする。主記憶の総量よりも大きい場合の拡張は容易である。

3.1 Non Partitioned Apriori : NPA

NPAでは候補アイテム集合を全ノードに複製してサポート値を調べる。以下にアイテム数 k の頻出アイテム集合を求める処理 (長さ k のパス) を示す。

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。ここで、分類階層の上下関係となるアイテムの組合せを含むものを削除し、残った候補アイテム集合を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に上位アイテムを付加する。ここで、どの候補アイテム集合にも含まれていないアイテムを除去する。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。ここで、アイテムとその上位アイテムを共に含む組合せについては調べない。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。
3. 全てのトランザクション対する処理が終了した時点で、それぞれの候補アイテム集合の全ノードでの生起回数の総和を求め、頻出アイテム集合を決定する。

全ての候補アイテム集合を単一ノードの主記憶に保持できない場合、候補アイテム集合を分割して主記憶内のハッシュ表に挿入し、繰り返しデータベースを検索してサポート値を求める。NPA は単純であるが、候補アイテム集合の数が多い場合、データベースを検索する回数が多くなる。

3.2 Hash Partitioned Apriori : HPA

HPAでは候補アイテム集合をハッシュ関数を用いてノード間に分割して割り当てる。以下に長さ k のパスを示す。

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。ここで、分類階層の上下関係となるアイテムの組合せを含むものを削除し、残った候補アイテム集合にハッシュ関数を適用し、対応するノードの識別子を求め、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に上位アイテムを付加する。どの候補アイテム集合にも含まれていないアイテムを削除したものから 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。ここで、アイテムとその上位アイテムを共に含む組合せについては調べない。ハッシュ値に対応するノードの識別子を求め、そのノードにアイテムの組合せを送信する。他のノード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. 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、ノード毎に頻出アイテム集合を決定し、他のノードに放送する。これにより、全てのノードが長さ 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。

4 性能評価

前節で述べた並列列処理方式を IBM 社製分散メモリ型並列計算機 SP-2 上に実装し、性能測定を行った。本性能測定では 16 台のノード (RS/6000) が HPS(ハイパフォーマンス・スイッチ) と呼ばれる高速ネットワークを介して接続された構成を使用した。各ノードには 2GB のローカルディスクが接続されている。また、3 つの並列処理方式の性能を評価には、[3] で述べられた手順を基に小売業における購買トランザクションを模倣して作成したデータセットを用いた。各パラメータを表 1 に示す。

Parameter	Value
Number of transactions	1600000
Average size of the transactions	12
Average size of the maximal potemtially large itemsets	4
Number of maximal potemtially large itemsets	10000
Number if items	100000
Number of roots	250
Number of levels	4-5
Fanout	5

表 1: データセットのパラメータ

図 2にそれぞれの並列処理方式の最小サポート値を変化させた場合の長さ 2 のパスの処理時間の変化を示す。以下の実験では、最も候補アイテム集合数の多い長さ 2 のパスについてのみ測定した。また、トランザクションデータファイルはノード間でほぼ均等になるように分割して、各ノードのローカルディスクに割り当てた。

NPA では最小サポート値が小さくなるに従い、処理時間が急増する。最小サポート値が小さくなると候補アイテム集合数が増大し、全候補アイテム集合を単一ノードの主記憶上に保持できなくなる。この場合、NPA は候補アイテム集合を分割し、トランザクションデータベースを繰り返し検索して処理するため、多くのディスク入出力処理が必要となり、処理時間が非常に長くなる。また、HPA では候補アイテム集合を分割し、システム全体の主記憶を効果的に活用できるため、NPA と比較してディスク入出力コストが少ない。しかし、サポート値を求める処理でトランザクションデータの通信を必要とするため、候補アイテム集数が少ない場合、NPA よりも処理性能が悪くなるが、最小サポート値が小さい場合に効果的であることがわかる。実際の利用環境では興味深いルールを抽出するにはサポート値を低くする必要があり、HPA は有効であると言える。

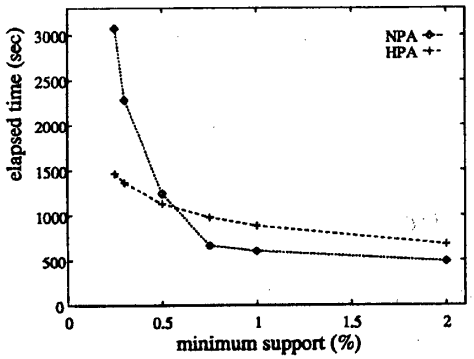

図 2: 並列処理方式の処理時間

図 3にノード数を変化させた場合 (4,8,16 台) の結果を示す。ここで、最小サポート値 0.5% とし、全ノードのトランザクション量を一定とした。また、グラフは 4 台のノードでの処理時間で正規化してある。

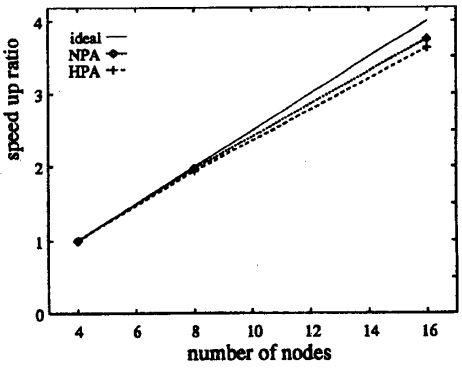

図 3: 台数効果

NPA、HPA 共に比較的良い台数効果が得られている。NPA ではサポート値を調べる処理で通信を必要としない。一方、HPA ではトランザクションデータの通信を必要とするが、その性能低下はわずかに押えられることが分かる。

5 まとめ

本稿では分散メモリ型並列計算機環境におけるデータの分類階層を考慮した相関関係抽出の並列処理方式について述べ、実際に並列計算機上に実装を行った。提案する HPA は、候補アイテム集合を分割する方法により主記憶を有効利用することで、サポート値の低い大規模なデータマイニングを効率良く実行できることを明らかにした。

参考文献

[1] 新谷, 喜連川: "データマイニングの並列化に関する一考察", 電子情報通信学会コンピュータシステム研究会 (CPSY95-88), 信学技報 Vol.95 No.47, pp.57-62, Dec 1995.

[2] 新谷, 喜連川: "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の実装とその評価", *JSPP'96*, 並列処理シンポジウム論文集 Vol.96No.3, pp.97-104, June 1996.

[3] R.Srikant, R.Agrawal: "Mining Generalized Association Rules", *VLDB'95*, pp.407-419, Sept 1995.