システムソフトウェアと
オペレーティング・システム 77−37
マルチメディア通信と分散処理 87−37
（1998. 2. 27）

ATM結合PCクラスタのTCP再送機構の解析と並列データマイニングの性能向上

小口正人、新谷隆彦、田村孝之、喜連川優

東京大学 生産技術研究所

〒106 東京都港区六本木7-22-1

{oguchi,shintani,tamura,kitsure}@tkl.iis.u-tokyo.ac.jp

近年大規模並列計算機はその構成要素であるCPUやディスク，メモリ等に汎用の部品を用いることが多くなってきた．プロセッサ間結合網だけは従来独自に開発されていたが，通信分野に於けるATMネットワーク技術の発展ならびに標準化の進行に伴い，プロセッサ間結合網にATMネットワークを用いることにより，低コストで性能の良いATM結合PCクラスタを構築することが可能になってきた．

本論文においては，100台のパーソナルコンピュータからなるATM結合PCクラスタを構築し，並列データマイニングのアプリケーションを実装して評価を行った．ATMスイッチのセル廃棄により引き起こされるTCP再送の解析を行い，大規模クラスタ上の並列処理に適した再送機構のパラメータ設定を明らかにした．この方式を用いることにより，100ノードのPCクラスタ上で並列データマイニングを実行した場合の性能向上が達成された．

Characteristics of TCP Retransmission Mechanism on ATM Connected PC Cluster and Performance Improvement of Parallel Data Mining

Masato OGUCHI, Takahiko SHINTANI, Takayuki TAMURA,
and Masaru KITSUREGAWA
Institute of Industrial Science, University of Tokyo

7-22-1 Roppongi Minato-ku, Tokyo 106, Japan

{oguchi,shintani,tamura,kitsure}@tkl.iis.u-tokyo.ac.jp

Recently, massively parallel computers employ commodity parts rather than proprietary components. Although the interconnection network has not yet been commoditized, ATM technology has come to be one of de facto standards for high speed networks. Therefore ATM connected PC cluster is promising platform from the cost/performance point of view.

In this paper, ATM connected PC cluster consists of 100 PCs is constructed, and performance of parallel data mining application implemented on the cluster is evaluated. TCP retransmission caused by cell loss at the ATM switch is analyzed, and parameters of retransmission mechanism suitable for parallel processing on the cluster are clarified. Using this proposed method, performance improvement of parallel data mining application is achieved in the case of execution on the 100 nodes PC cluster.

1 はじめに

PC/WS(パーソナルコンピュータ／ワークステーション)クラスタは，並列処理の研究分野において近年注目を集めている．スケーラビリティ，コストパフォーマンス(単位価格当たりの性能)などの面から，次世代の大規模並列計算機システムには，PC/WSクラスタやこれに類するプラットフォームが中心的な役割を果たす可能性が高いと考えられる．それは以下のような理由による．

市販の並列計算機の最近の傾向として，システムの構成要素であるCPUやディスク，メモリなどに，従来は専用のものが用いられていたのに対し，汎用の部品が用いられるようになってきたことが挙げられる．これは，汎用の製品でも十分な性能が得られる程にこれらに関する技術が成熟したためであり，開発コストを削減しコストパフォーマンスを向上させる事を目的とし，出来るだけ多く汎用の部品を取り入れようとする傾向が見られる．

また近年の傾向として，パーソナルコンピュータが性能の面でワークステーションと互格になってきたということがあげられる．今日のワークステーションに用いられているRISCプロセッサは，浮動小数点演算に関してはパーソナルコンピュータのプロセッサに比べてかなり高い能力を持っている．しかしある種のアプリケーション，例えばデータベース処理などにおいては，浮動少数点演算よりもむしろ整数演算処理に関して高いパフォーマンスが要求される場合が多い．パーソナルコンピュータは整数演算においてはワークステーションに近い処理能力を持っており，従ってデータベース処理に関してはパーソナルコンピュータの方がワークステーションよりも優れたコストパフォーマンスを実現しているといえる．

一方結合ネットワークに関しても，最近では高速ネットワーク技術が急激に進歩しており，近未来の大規模並列計算機には汎用ネットワークが用いられる可能性が高い．次世代高速ネットワークに関しては，現在一つに絞られているとは言い難くいくつかの有力候補が混沌としている状況であるが，品質保証の概念やスケーラビリティなどこれまでの方式には見られない優れた特徴を持つATMは，有力な候補の一つであると考えられる．既にATMスイッチやNIC(Network Interface Card)の低価格化が進んでおり，今後更なるコストパフォーマンスの向上も期待できる．

このような近年の技術動向を鑑みると，ATM結合PCクラスタは将来の大規模並列処理システムの有望なプラットフォームであると見做すことができる．これまでFast EthernetやMyrinet等を用いたPC/WSクラスタの研究がいくつか報告されており，その中では科学技術計算のベンチマークプログラムの実行結果などが示されている[1][2][3]．本論文では，100台のパーソナルコンピュータをATMネットワークで接続した大規模PCクラスタを構築し，並列データマイニングのアプリケーションを実装し評価，検討を行った．データマイニングのようなデータベース処理を中心とするアプリケーションは，科学技術計算と並び並列処理の重要なアプリケーションと考えられ，特に大量のデータを処理する場合には，大規模な並列計算機環境が必要となる．しかしながら，ATMやパーソナルコンピュータはもともとクラスタを構築することを念頭に置いて設計されたものではないため，物理的に接続しただけでは効率良く並列処理を行うことが難しい可能性がある．特に本研究に於いては100台の計算機が汎用の通信ネットワークで結合された環境で並列処理を行うため，100台の負荷が同時にかかるネットワークが期待通りの性能を発揮できるかどうかが，システム全体の成功の鍵となる．本論文では，通信プロトコルに標準的なTCP/IPを用いて効率良くシステムを動作させることを検討する．

表 1: Each node of PC cluster

CPU	Intel Pentium Pro 200MHz
Chipset	Intel 440FX
Main memory	64Mbytes
Disk drive	2.5Gbytes IDE hard disk
OS	Solaris2.5.1 for x86
ATM NIC	Interphase 5515 PCI ATM Adapter

2 PCクラスタの構成

クラスタの各ノードには，200MHzのPentium Pro PCを用いた．各PCの構成要素を表1に示す．

全ノードが155Mbps ATM LANとイーサネットにより結合されている．ATM LANのドライバには，IP over ATM用のLLC/SNAP encapsulationをサポートするRFC-1483 PVC(Permanent Virtual Channel)ドライバを用いた[4][5]．このドライバはQoSクラスとしてUBRのみをサポートしている．ATMスイッチには128ポートの155Mbps UTP-5インタフェースを持っている日立製のAN1000-20を用いた．各ポートあたりのバッファサイズは2kセルである．PCクラスタの全体構成を図1に示す．

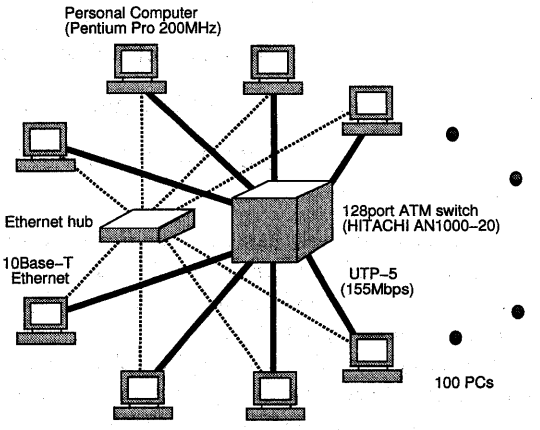

図 1: An overview of the PC cluster

3 データマイニングアプリケーションとその並列化

3.1 相関関係抽出の概要

データマイニングは近年データベースの分野において注目されているテーマの一つである．データマイニングの代表的な例として，トランザクションデータのアイテム間の相関関係 (association rule) 抽出が挙げられる．トランザクションデータは一般に，トランザクションIDとトランザクション毎のアイテム集合から構成される．例えば「商品Aおよび商品Bを買った顧客の90%は商品Cも買っている」に示される法則や傾向をトランザクションデータから探り出すことが試みられている [6].

質の高い相関関係を得るためには大量のトランザクションデータを処理する必要があり，長い処理時間を要する．そのため単一プロセッサのシステムでは，十分に速いレスポンスタイムは得られない．そこで筆者らは相関関係抽出の並列化の研究に取り組んできた．相関関係抽出のアルゴリズムとして知られているものに，IBM アルマデン研究所の R. Agrawal による Apriori がある [7][8]．筆者らはこれまで Apriori をベースとした相関関係抽出の並列アルゴリズムをいくつか考案してきた [9]．本論文ではそのうちの一つの HPA(Hash Partitioned Apriori) を取り上げる．Apriori は，まず始めに長さが 1 の候補アイテム集合 (candidate itemset) を作り，トランザクションデータベースを読み出して，候補アイテム集合がユーザの指定した最小サポート値を満たすかどうか判断する．条件を満たすものを頻出アイテム集合 (large itemset) と呼ぶ．この頻出アイテム集合を用い，次に長さ 2 の候補アイテムを作り同じ処理を行う．これを最小サポート値を満たすアイテムが無くなるまで繰り返す．頻出アイテム集合より最大コンフィデンス値を満たすルールを導出することが出来る．

3.2 抽出アルゴリズムの並列化

Apriori の最も単純な並列化は，候補アイテム集合を全てのプロセッサにコピーし，各プロセッサにおいてトランザクションデータベースの読み出しなどを並列に処理する方法である．候補アイテム集合の数が少なくこれが一つのプロセッサのローカルメモリに全て乗り切る小規模なマイニングではこの方法でも問題は生じないが，大規模なデータマイニングを行う場合には，候補アイテム集合に必要なメモリ量が単一プロセッサにおいて使用できるメモリ空間より大きくなり，2 次記憶へのアクセスに伴い大幅な性能低下が起こる．

HPA は候補アイテムを，ハッシュ関数を用いて各プロセッサに分配する．従って全ノードのメモリ空間を有効に利用することができ，大規模なデータマイニングには上述の単純な並列化に比べ高い性能が達成できる．HPA アルゴリズムの各ステップは以下の通りである．

1. 長さ k の候補アイテム集合の作成：

   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することにより，対応するプロセッサ ID を決定し，当該プロセッサの主記憶上のハッシュ表に挿入する．

2. 生起回数の数えあげ：

   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長さ k のアイテムの組合せを作成し，前項と同一のハッシュ関数を適用する．これにより得られたハッシュ値に対応するプロセッサ ID を求め，そのプロセッサにアイテムの組合せを送信する．受信したプロセッサは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. 長さ k の頻出アイテム集合の決定：

   すべ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，これを他のすべてのプロセッサにブロードキャストする．全てのプロセッサに於いて頻出アイテム集合が得られなければ，処理を完了する．

表 2: The number of candidate and large itemsets

C	候補アイテム数
L	頻出アイテム数
T	各パスの実行時間 [sec]

pass	C	L	T
pass 1		1023	11.2
pass 2	522753	32	69.8
pass 3	19	19	3.2
pass 4	7	7	6.2
pass 5	1	0	12.1

4 データマイニングアプリケーションの PC クラスタへの実装

4.1 HPA プログラムの実装

PC クラスタ上に前章で述べた HPA プログラムを実装して実験を行った．各ノードは自身のハードディスクを持ち，分割されたトランザクションデータファイルがこれに格納されている．各ノードにおいては二つのプロセスが生成され，一つのプロセスが頻出アイテム集合を元に候補アイテム集合を作り，他のプロセスがハッシュ表のチェックを行い，互いに協調して処理が行われる．

プロセス間通信には Solaris の socket ライブラリを用いた．全てのプロセス間は互いに socket コネクションにより接続され，メッシュ状のトポロジーとなっている．socket コネクションのタイプには，両方向のバイトストリームである SOCK_STREAM を用いた．ATM レベルのコネクションとしては，データが全ノード間で絶えず送受信されるため，PVC を用いた．QoS クラスは UBR であり，TCP/IP over ATM により通信が行われる．

トランザクションデータは R. Agrawal によって開発されたデータ生成プログラムを使って作成した．これはトランザ

クション数，アイテム数など幾つかのパラメータを指定すると人工的なトランザクションデータを作り出すものである．

4.2 実行結果

本実験に使われたパラメータは以下の通りである．トランザクション数を10,000,000，アイテムの種類を5000，最小サポート値を0.7%とした．トランザクションデータの総量は約800Mbytesである．メッセージのブロックサイズは8Kbytes，ディスクI/Oサイズは64Kbytesとした．

上記のパラメータ及びトランザクションデータを用いて100ノードPCクラスタ上で実験を行った結果，各パスにおける候補アイテム数と頻出アイテム数，そして各パスの実行時間は表2のようになった．

表からパス2における候補アイテム数が他に比べ多く，実行時間も長くなっていることがわかる．これは相関関係抽出における典型的な傾向である．

一方パス3，4，5に関しては，処理するデータ量が少ない割には実行時間が長い．トレースを取って詳細に調べた結果，TCPの再送がこの時点で起こっていることがわかった．TCPの再送は，ネットワークが輻輳した場合にATMスイッチでセルが廃棄されることにより生じる．興味深いのは，TCPの再送がパス3–5においてのみ起きており，データ転送量が非常に多いパス2では起きていないという点である．

各パスの終わりではバリア同期が実行され，全ノード間でデータの交換が必要となる．即ちこれらの時点に於いて全対全のブロードキャストが実行される．たとえブロードキャストされるデータ量が多くなくとも，ブロードキャストの実行タイミングが全ノードでほとんど同じである場合には，ATMスイッチに於いてセル廃棄が起こる可能性が高い．パス3–5は処理するデータの量が少ないため各パスの実行時間は短く，従ってブロードキャストが全ノードでほぼ同時に行われると考えられる．その結果ネットワークに輻輳が起こり，ATMスイッチでセルが廃棄され，TCPの再送が起こる．

このようなバリア同期やデータのブロードキャストは，並列・分散アプリケーションにおいて不可欠な処理であり，この問題を次章で検討する．

5 ATM結合PCクラスタにおけるTCP再送機構の影響及びHPAプログラムの性能向上

前章で述べた再送処理は，ネットワークの輻輳と密接な関連がある．本研究で構築したPCクラスタのATM層はUBRクラスで動作しているため，ネットワーク上でセルの衝突が起こることは必然的であり，輻輳によりATMスイッチのバッファが溢れるとセル廃棄が起こる．この際上位層であるTCPの再送機構に於いて再送間隔などのパラメータが適当な値に設定されていないと，ネットワークの混雑をさらに助長する可能性がある．

そこで本論文では「TCP再送間隔の最大値(maximum interval of TCP retransmission)」及び「TCP再送間隔の最小値(minimum interval of TCP retransmission)」という

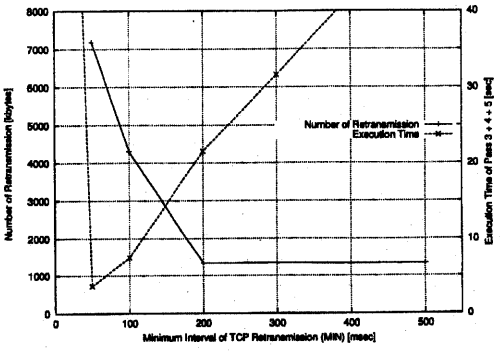

図2: Changing minimum interval of TCP retransmission (MAX = Default(60000[ms]))

2つのパラメータを調節することにより，TCP再送機構の動作を変え，システム性能に対する影響を評価した．これらの2つのパラメータを以後MAX及びMINと呼ぶ．デフォルトの設定は，現在用いているOS(Solaris2.5.1)においてはMAX = 60000[msec]，MIN = 200[msec]である．TCPの再送間隔は，ネットワークの混雑等に対応してレスポンスタイムに基づきMAXとMINの間で動的に変更される．

本章では，HPAプログラムのパス3–5に焦点を当てて解析を行い，TCP再送機構のパラメータを変更した時にこれがHPAプログラムの実行にどのような影響を及ぼすかという点を検討する．

5.1 TCP再送間隔の変更

はじめに，TCP再送間隔の最大値(MAX)をデフォルト(60000[msec])のままとし，TCP再送間隔の最小値(MIN)を変更する実験を行った．HPAプログラムの実行中のTCPの再送バイト数とパス3+4+5の実行時間(パス3からパス5までの実行時間の合計)を図2に示す．以下本論文中において，TCPの再送バイト数は100台のPCにおけるTCPの再送バイト数の平均値を指す．この値には，パス3–5の間を含めプログラム実行中に起きたすべてのTCP再送が含まれる．

図2の横軸はMINである．MINが長くなるとパス3+4+5の実行時間は長くなっている．アプリケーションそのものは全く同じ条件で実行しているため，この違いはTCPの再送間隔にのみ依存している筈である．ネットワーク上で衝突が起こり再送が繰り返される度に再送間隔は長くなってゆくが，この場合MAXの値が大きいため，再送の繰り返し回数が多くなると再送間隔は長いものになる．一方再送バイト数の方は，MINが200[msec]よりも短くなると急激に増加している．

次にMAXの値もデフォルトから変更して実験を行った．デフォルトのMAXは，このようなクラスタ上の通信には長すぎると考えられるので，MAX = MIN + 100[msec]と短い値を用いた．図3にTCPの再送バイト数とパス3+4+5

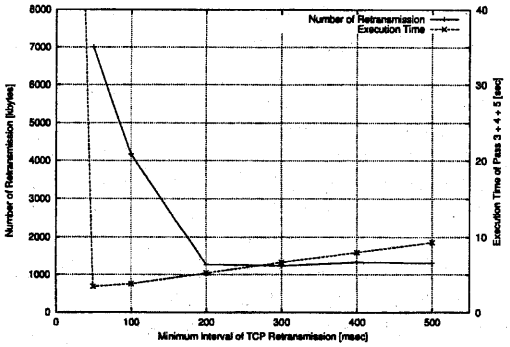

図 3: Changing minimum interval of TCP retransmission (MAX = MIN + 100[ms])

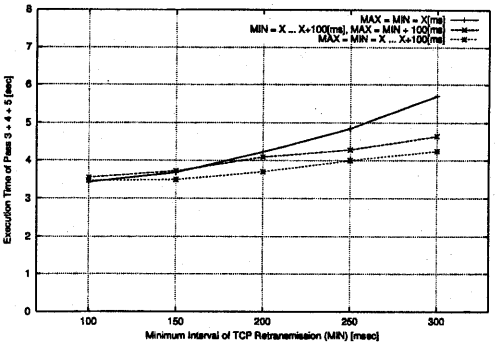

図 4: Execution time of pass 3 + 4 + 5

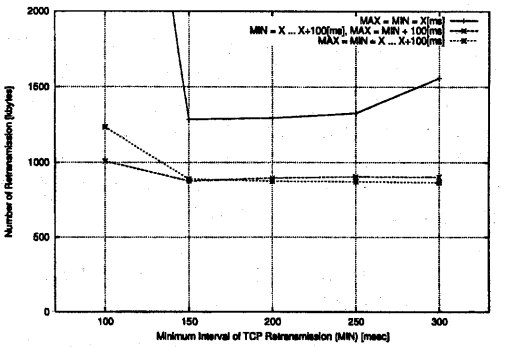

図 5: Number of retransmission

の実行時間を示す．横軸は MIN である．この場合は図 2に比べて，パス 3+4+5 の実行時間は全体としてかなり短くなっている．しかしながら MIN が長くなるとパス 3+4+5 の実行時間が長くなるという傾向は変わっていない．また MIN が 200[msec] よりも短くなると TCP の再送バイト数が急激に増加するという傾向も図 2と同じである．MIN を短くすると再送間隔が短くなる可能性が高くなるが，再送は途中で送信データが紛失したことを想定しての投機的な処理であるため，再送間隔が短すぎると実際には確認応答の到達が遅れているだけであるにも拘らず再送を実行する可能性があり，その場合再送パケットが原因でネットワークやシステムにさらに負荷がかかる．

以上の結果から本実験環境では，MIN の値はデフォルト (200[msec]) あたりがほぼ適当なものであり，短い値の方が良いが短すぎると不必要な再送を引き起こす可能性があることがわかった．一方デフォルトの MAX の方は，PC クラスタにおける並列処理には長すぎる値なので短く設定すべきである．

5.2 TCP 再送間隔のランダム化

TCP は再送間隔を，各時点において最適と考えられる値に動的に調整する機能を備えているが，これだけでは大規模な PC クラスタ上での並列処理においては十分ではない．接続されているノードが小数のローカルネットワークの場合と異なり，大規模 PC クラスタの場合には多くのノードがほぼ同じタイミングで同じ処理を行っている．従って多くのノードの再送間隔が同じ値に揃ってしまい，その結果パケットの衝突，更なる再送が引き起こされる可能性がある．

そこで，TCP の再送間隔としてノード毎に異なる値を用いる方法を試みた．実験結果を図 4及び図 5に示す．

これらの実験では，TCP の再送間隔を決定するのに以下の 3 種類の方式を用いた．まず一番目は，全てのノードが固定された同じ値を用いた場合，つまり MAX = MIN = X(横軸 X の値) としたケースである．二番目は，MIN として X ... X + 100[msec] の間のランダムな値を取り，MAX = MIN + 100[msec] とした場合である．そして三番目は MAX，MIN 共に X ... X + 100[msec] の間のランダムな値を用い，MAX = MIN とした場合である．ランダムな値を用いる方式では，MAX や MIN はノード毎に異なる値を取るが，プログラムの実行中各ノードにおいてそれらの値は変化しない．また三番目の方式では，MAX と MIN はノード毎に異なる値であるが，各ノードにおける TCP の再送間隔はプログラムの実行中動的に変化せず一定値となる．

図 4，図 5を見ると，この実験環境において，TCP 再送間隔にノード毎にランダムな値を用いる方法が有効であることがわかる．とりわけ再送バイト数の低減に対する効果が大きい．例えば全てのノードで MAX = MIN = 200[msec] と同じ値を用いた場合よりも，MAX = MIN = 200 ... 300[msec] とより長いがランダムな値を用いた場合の方が良い結果となっている．

MIN の値が短くなると，この場合もパス 2 における再送の問題が起こる．これは再送間隔の絶対的な長さの問題であるので，MIN にランダムな値を用いてもパス 2 での再送を防ぐことは出来ない．

以上の結果より大規模 PC クラスタにおいては，同じタ

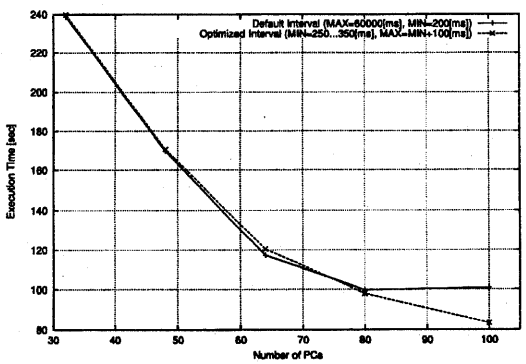

図 6: Execution time of HPA program on PC cluster

イミングでの全対全ブロードキャストを含むようなアプリケーションの場合，TCP 再送間隔としてノード毎に異なるランダムな値を用いる方法が効果的であることが分かった．ただし TCP 再送間隔の最小値を短くしすぎると，データ送信量が多い時に受信確認の到達が遅れて不必要な再送が起こってしまうため，例えばこの実験環境に於いては最小値を 150−200[msec] 程度以上にする必要がある．

5.3 TCP 再送間隔に提案手法を用いた場合の HPA プログラムの性能向上

本節では，PC クラスタのノード数を変えた時の HPA プログラムの実行時間について考察する．図 6の実線は TCP 再送パラメータをデフォルト，つまり MAX = 60000[msec], MIN = 200[msec] とした場合，破線は前節で提案したノード毎に異なるランダムな値をパラメータに用いた方式の場合である (この場合 MIN = 250 ... 350[msec], MAX = MIN + 100[msec] とした)．

再送間隔の最大値を短くしてランダムなパラメータを用いた場合には，デフォルトのパラメータを用いた場合に比べて，ノード数が 100 に近付いた所で実行時間が短縮されている．パス 1 および 2 の実行時間は二つの方式でほとんど違わないため，提案したパラメータの設定方法がパス 3–5 で有効に働き，その結果全体として良好な性能を達成していることがわかる．

6 むすび

本論文では，100 台の 200MHz Pentium Pro パーソナルコンピュータを ATM スイッチで接続した大規模 PC クラスタを構築した．通信プロトコルには TCP/IP over ATM を用いた．TCP は再送間隔を，各時点において最適と考えられる値に動的に調整する機能を持っているが，それだけでは十分でない場合がある．大規模 PC クラスタの場合，多くのノードが同じタイミングでブロードキャストを行う可能性があり，再送処理も同じ手続きなため多くのノードの動的に決定される再送間隔が同じ値に揃ってしまい，その結果パケットの衝突，更なる再送が引き起こされる可能性があるからである．

再送処理のパラメータをどのように設定するかは，ネットワークを広域通信に用いる場合と大規模クラスタに用いる場合とでは自ずと異なってくる．本論文では汎用通信プロトコルである TCP を大規模 PC クラスタで用いる場合の問題点を明らかにし，解決策を提案して，データマイニングアプリケーションを用いてその有効性を実証した．再送処理は通信プロトコルに必須の機能であることから，本論文で得られた結論は，TCP に限らず一般に汎用通信プロトコルを大規模クラスタで利用する場合に有効であると考えられる．

参考文献

[1] R. Carter and J. Laroco, "Commodity Clusters: Performance Comparison Between PC's and Workstations", *Fifth IEEE International Symposium on High Performance Distributed Computing*, pp.292-304, August 1996.

[2] D. E. Culler, A. A. Dusseau, R. A. Dusseau, B. Chun, S. Lumetta, A. Mainwaring, R. Martin, C. Yoshikawa, and F. Wong, "Parallel Computing on the Berkeley NOW", 並列処理シンポジウム *JSPP '97*, pp.237-247, May 1997.

[3] H. Tezuka, A. Hori, Y. Ishikawa, and M. Sato, "PM: An Operating System Coordinated High Performance Communication Library", *HPCN Europe 1997*, pp.708-717, April 1997.

[4] J. Heinanen, "Multiprotocol Encapsulation over ATM Adaptation Layer 5", *RFC1483*, July 1993.

[5] M. Laubach, "Classical IP and ARP over ATM", *RFC1577*, January 1994.

[6] U. M. Fayyad, G. P. Shapiro, P. Smyth, and R. Uthurusamy, "Advances in Knowledge Discovery and Data Mining", *The MIT Press*, 1996.

[7] R. Agrawal, T. Imielinski, and A. Swami, "Mining Association Rules between Sets of Items in Large Databases", *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, pp207-216, May 1993.

[8] R. Agrawal and R. Srikant, "Fast Algorithms for Mining Association Rules", *Proceedings of the 20th International Conference on Very Large Data Bases*, September 1994.

[9] T. Shintani and M. Kitsuregawa, "Hash Based Parallel Algorithms for Mining Association Rules", *Proceedings of the Fourth IEEE International Conference on Parallel and Distributed Information Systems*, pp.19-30, December 1996.