社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

IP-SAN 統合型 PC クラスタにおける 並列データ処理アプリケーション実行時の特性解析

原　明日香†　　神坂紀久子††　　山口　実靖†††　　小口　正人†

† お茶の水女子大学　〒 112–8610 東京都文京区大塚 2–1–1
†† 情報通信研究機構　〒 184-8795 東京都小金井市貫井 4–2–1
††† 工学院大学　〒 163–8677 東京都新宿区西新宿 1–24–2
E-mail: †asuka@ogl.is.ocha.ac.jp, oguchi@computer.org, ††kikuko.kamisaka@nict.go.jp,
†††sane@cc.kogakuin.ac.jp

あらまし　近年、情報システムにおいて処理される情報量が爆発的に増大しており、その中からユーザが必要とする情報を高速に取り出すことが求められている。そこで膨大なデータを処理するために、本研究ではバックエンドとフロントエンドのネットワークを統合した IP-SAN 統合型 PC クラスタを構築して利用した。ただしアプリケーション実行時にクラスタのノード間通信や I/O の実行がどのように振舞いシステム性能に影響を与えているのかなど詳しい解析は行われていない。そこで本研究では、さまざまなデータ処理アプリケーションを実行し、システムのモニタを行い解析することによって、IP-SAN 統合クラスタの詳しい振舞を明らかにする。

キーワード　Storage Area Network，iSCSI, PC クラスタ, 並列分散処理

Analyzing characteristics of PC cluster consolidated with IP-SAN in the execution of data-intensive applications

Asuka HARA†, Kikuko KAMISAKA††, Saneyasu YAMAGUCHI†††, and Masato OGUCHI†

† Ochanomizu University　2–1–1 Otsuka, Bunkyo-Ku, Tokyo 112–8610 Japan
†† National Institute of Information and Communications Technology　4–2–1 Nukui, Koganei-city, Tokyo, 163–8677 Japan
††† Kogakuin University　1–24–2 Nishi-shinjuku, Shinjuku-ku, Tokyo, 163–8677 Japan
E-mail: †asuka@ogl.is.ocha.ac.jp, oguchi@computer.org, ††kikuko.kamisaka@nict.go.jp,
†††sane@cc.kogakuin.ac.jp

Abstract　In the information society that has been developing in recent years, the volume of processed data increases explosively. Information required from users is requested to be taken out immediately from them. In order to process huge data, we have constructed PC cluster consolidated with IP-SAN that integrated the front-end and the back-end network into the same IP network. However, the analysis how the communication between nodes of the cluster and the execution of I/O influences the behavior of system performance, when the application is executed, is not performed in detail. Thus, in this research, various data processing applications are executed, while the system is monitored and analized, so that detailed behavior of PC cluster consolidated with IP-SAN is clarified.

Key words　Storage Area Network，iSCSI, PC cluster, Parallel Distributed Processing

1. はじめに

近年、パーソナルコンピュータにおけるコモディティなハードウェアの価格低下と性能改善が進み、大規模な科学技術計算、データベース処理やデータマイニングのようなハイパフォーマンスなデータ処理アプリケーションが PC クラスタ上で実行されるようになっている。大規模な PC クラスタにおいては、Fibre Channel や Infini Band のような高速な専用回線がノードとストレージの間のネットワークとしてよく使用使されている。しかしながら、IP ネットワークをベースとした Storage

Area Network (IP-SAN) の出現により、コモディティな技術をベースとしたネットワークだけで PC クラスタの構築が可能となってきている。

そこで本研究では、IP-SAN の代表的なプロトコルである iSCSI を用いることでフロントエンド LAN とバックエンド SAN のネットワークを統合した IP-SAN 統合型 PC クラスタの提案を行う。本稿においては、Target に接続する Initiator の数を変化させることにより、ネットワークと Target に高負荷をかけながら、ストレージのベンチマークと並列データマイニングを用いて IP-SAN 統合型 PC クラスタの評価を行う。

2. IP-SAN 統合型 PC クラスタ

2.1 PC クラスタにおける SAN の利用

近年、情報システムにおいて処理されるデータの量が膨大になってきたことから、ストレージ分野においてネットワークストレージ技術が発展し、サーバとストレージを結ぶネットワークである SAN が登場して、普及するようになった。HPC 分野では、PC クラスタの記憶装置において、計算ノードーストレージ間のバックエンドのネットワークに SAN を用いることが多くなっている。SAN は、分散したストレージをネットワークで統合し、集中管理とディスク資源の効率的な活用を可能にしている。

図 1 は、SAN を用いて構築した PC クラスタの例である。現在、SAN としては、高速な専用回線である Fibre Channel を用いる FC-SAN が普及している。一般に、ディスクへの I/O 処理を行うストレージアクセスはノード間通信と比べてバースト性が高く、転送データ量が多いため、計算ノード (サーバ) ーストレージ間のバックエンドには高速な FC-SAN を用いることが多い。しかし FC-SAN には、FC 用のスイッチが高価であることなど、PC クラスタに導入して管理するにはコスト面で障害がある。

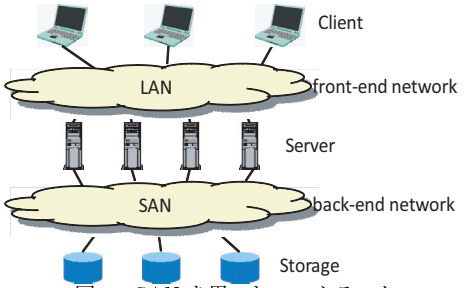

図 1　SAN を用いた PC クラスタ

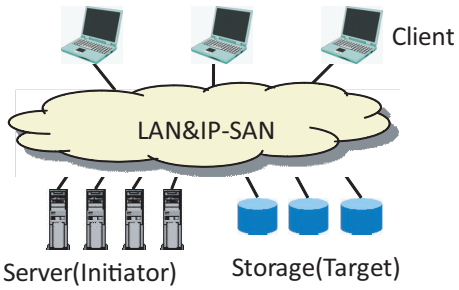

図 2　IP-SAN 統合型 PC クラスタ

IP-SAN は、TCP/IP ネットワークで SAN を構築する次世代の SAN である。図 1 に示す PC クラスタにおいて、FC を用いて構築する従来の SAN に代わり、バックエンドのネットワークを IP-SAN で構築することにより、安価なコストで PC クラスタのストレージを導入、運用が出来る。今後、Gigabit Ethernet/10Gigabit Ethernet が広く普及していくであろうことを考慮すると、IP-SAN をバックエンドに持つ PC クラスタが使用されるようになると考えられる。

2.2 IP-SAN 統合型 PC クラスタと性能への懸念

我々は、図 2 に示すように、計算ノード (サーバ) ーストレージ間のバックエンドネットワークを、ノード間を接続するフロントエンドに統合した iSCSI(Internet SCSI) 接続の IP-SAN 統合型 PC クラスタを提案し、評価を行っている。iSCSI は、2003 年 2 月に IETF により正式認証された IP-SAN のプロトコルであり、SCSI コマンドを TCP/IP パケットの中にカプセル化することでブロックレベルのデータ転送を行う [1]。iSCSI の性能についての議論が数多くされている [2][3][4]。iSCSI の階層構造は、図 3 のようになっている。

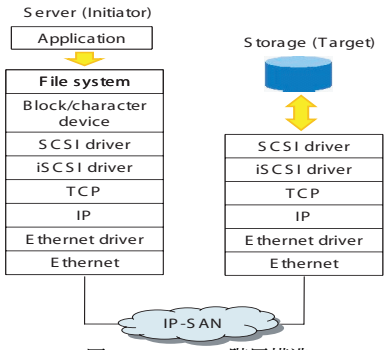

図 3　iSCSI の階層構造

SAN を使用した PC クラスタでは、一般にフロントエンド LAN とバックエンド (IP-)SAN が別々になっているため、2 つの異なるネットワークの構築が必要になる。これに対し、IP-SAN 統合型 PC クラスタでは、iSCSI を使用することで、双方のネットワークを TCP/IP と Ethernet を用いたコモディティなネットワークに統一することができる。それにより、ネットワーク構築コストの削減と運用管理の効率化が可能となる。

しかし、IP-SAN 統合型 PC クラスタは、フロントエンドにおけるノード間通信とバックエンドのストレージアクセスにおけるバルクデータが、TCP/IP over Ethernet である同一のネットワーク経由で混在して転送される。そのため、フロントエンドとバックエンドのネットワークを個々に構築する非統合型と比較して、並列分散処理実行時のネットワークへの負荷が懸念される。例えば、ノード間通信とストレージアクセスで同じネットワークリソースを使用するため、互いに衝突する可能性がある。その結果、ストレージアクセスのバルクデータにより並列計算のためのノード間通信が多大な影響を受け、全体の性能が劣化する可能性が推測される。従って、バックエンドネットワークをフロントエンドネットに統合した IP-SAN 統合型 PC クラスタは、非統合型 PC クラスタと比較して、統合がどの程度性能に影響を及ぼすかを評価する必要がある。

3. 相関関係抽出とその並列化

相関関係抽出では、巨大なデータから有益な規則性や関係を

抽出するために、あるパターンが現れる頻度 (サポート値) を調べる。その頻度が多ければ、そのパターンから得られる関係は有意義な情報となり、販売戦略などに活用出来る。

相関関係抽出で扱うデータはしばしば巨大であるため、データベースを分散し計算処理を並列化して、多数台のコンピュータをネットワークで接続した PC クラスタなどの環境でマイニング処理を実行する並列相関関係抽出の研究が行われている [7]。以下に相関関係抽出の代表的な 2 つのアルゴリズムの概要を説明し、本研究で用いる並列化アルゴリズムを紹介する。

3.1 Apriori アルゴリズム

1994 年に Agrawal らによって提案されたもので、発見された頻出アイテムセットから候補アイテムセットを生成し、繰り返し数え上げを行っていくアルゴリズムである [5]。

Apriori アルゴリズムには、候補アイテムセットを格納するために大容量のメモリが必要となる、何度も繰り返しデータベースをスキャンする可能性があるといった問題点がある。

Apriori をベースにした並列相関関係抽出のアルゴリズムはいくつか提案されているが、本研究ではハッシュ関数を使用して Apriori を並列化する HPA (Hash Partitioned Apriori) を用いる。

3.2 FP-growth アルゴリズム

2000 年に Han らによって提案されたもので、巨大なトランザクションデータベースから相関関係抽出に必要な情報をコンパクトに圧縮したデータ構造である FP-tree を利用している [6]。候補パターンを生成せずに頻出パターンを抽出することで、Apriori アルゴリズムの問題点を改善したアルゴリズムである。

FP-growth は構築された FP-tree の性質を利用することにより、頻出パターンを発見していくアルゴリズムである。FP-growth の並列相関関係抽出のアルゴリズムは、本研究では HPA を元に行われた既存研究 [7] で提案された PFP(Parallelized FP-growth) を用いる。

FP-growth アルゴリズムは Apriori アルゴリズムと比較して極めて高速であると言われている。ただしデータの性質によっては、FP-tree が巨大になってしまう可能性のある点が問題である。

4. 実験概要

本実験では、複数 Initiator と単数 Target を Gigabit Ethernet で接続し、ネットワークと Target に高付加がかかる環境にした IP-SAN 統合型 PC クラスタの性能評価を行う。Target ストレージとして、近年におけるハイパフォーマンスなストレージである SAS ディスクを RAID0 に構成したものを用いる。表 1 にクラスタノードのスペックを示す。

クラスタの構成と管理を行うために Rocks[8] を導入し、クラスタのモニタリンスツールとして Ganglia[9] をインストールした。また、iSCSI の Initiator として open-iscsi-2.0-865[10]、Target として iSCSI Enterprise Target(IET)-0.4.15[11] を使用する。

表 1　Experimental setup：PCs

OS	Initiator : Linux 2.6.18 Target : Linux 2.6.18
CPU	Initiator : Intel Xeon 3.6GHz Target : quad-core Intel Xeon 1.6GHz
Main Memory	Initiator : 4GB Target : 2GB
HDD	Initiator : 250GB SATA Target : 73GB SAS × 2 (RAID 0)

5. 実験結果と考察

5.1 Bonnie++

まず、複数 Initiator から単数 Target にアクセスすることで、ネットワークと Target に高負荷をかけた実験システムの評価をハードディスクベンチマークツールである Bonnie++[12] を用いて行った。RAID0 で構成された SAS ディスクを用いたローカルデバイスと SAS ディスクを iSCSI の Target ストレージとした iSCSI アクセスにおけるの Sequential read アクセスと write アクセスを測定した。

図 4 に Target における Initiator からのアクセスの合計スループット、図 5 に Target のネットワークトラフィック、図 6 に Target の CPU 使用率、図 7 に Initiator と Target を 8 対 1 に接続したときの Target のストレージ I/O をそれぞれ示す。

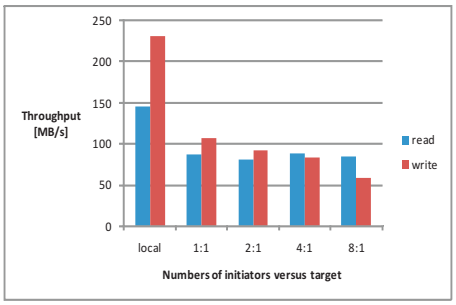

図 4　Sequential read アクセスと Sequential write アクセス

図 4 の結果から、まずローカルアクセスと iSCSI アクセスを比較した結果、ローカル write が極めて速いことから、iSCIS write はその半分以下に留まっているが、ローカル read と比較して iSCSI read はその 2/3 近い性能が出ている。また、Sequential read アクセスの場合においては、1 対 1、2 対 1、4 対 1、8 対 1 のどの場合においても合計スループットは殆んど変わらず、Sequential write アクセスの場合においては、1 対 1、2 対 1、4 対 1、8 対 1 と負荷を高くしていくごとにスループットがやや低下するということがわかる。

図 5 の結果から、どの場合においても最大 60Mbytes/sec(約 500Mbps) のトラフィックしか流れておらず、本実験では Gigabit Ethernet を使用しているため、ネットワーク帯域にはまだ余裕があることがわかる。また、Target のネットワークトラフィックの最大値はどの場合もほぼ一定であり、ネットワークにはそれ以上の負荷はかかっていないことがわかる。

図 6 と Initiator の CPU 使用率のモニタリングの結果から、Initiator と Target のどちらとも、負荷が高くなるにつれて実行時間が長くなり、Target にアクセスが集中した場合におい

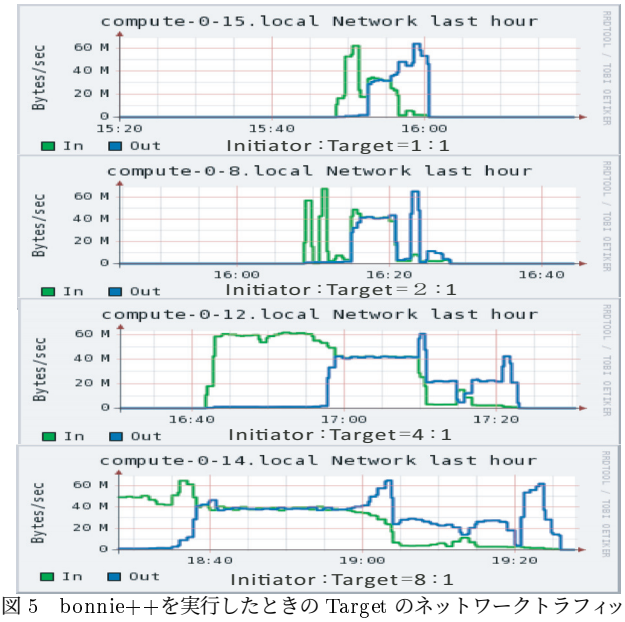

図 5　bonnie++を実行したときの Target のネットワークトラフィック

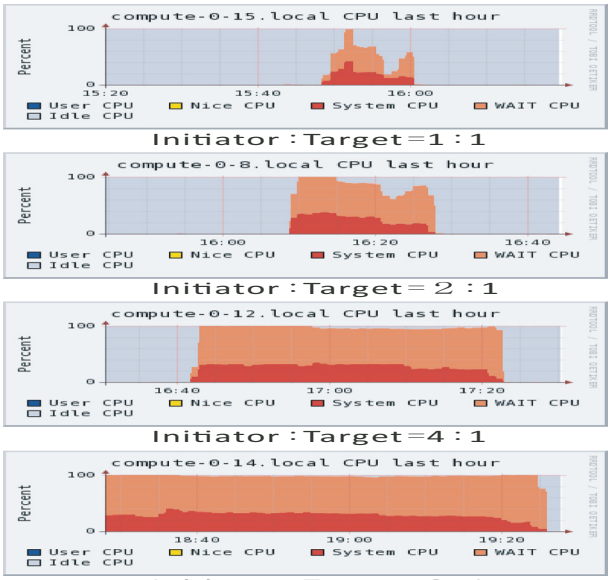

図 6　bonnie++を実行したときの Target の CPU 使用率

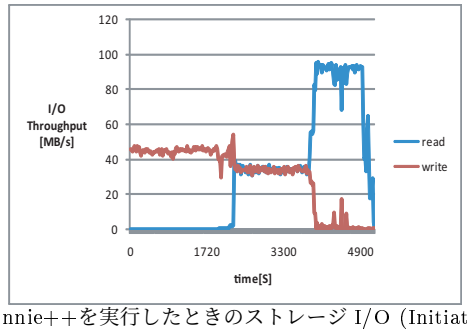

図 7　bonnie++を実行したときのストレージ I/O (Initiator:Target = 8 : 1)

ても CPU は実行時間に比べて待ち時間が長く、ストレージアクセスのための CPU 処理にはまだ余裕があると言える。これに対し、Target の CPU の待ち時間もそれに伴い長くなっている。一方、図 7 の結果から、パケット I/O は最大限に行われており、これがボトルネックとなっていると考えられる。

以上の結果から、Sequential read アクセスの場合においては、1 対 1、2 対 1、4 対 1、8 対 1 のどの場合においてもスループットは変わらず、Sequential write アクセスの場合においては、1 対 1、2 対 1、4 対 1、8 対 1 と負荷を高くしていくごとにスループットが低下するということがわかった。その時のネットワーク帯域にはまだ余裕があり、CPU も待ち時間が長い状態であるが、パケット I/O がボトルネックとなり、Target にアクセスを集中させた場合の Sequential write アクセスの性能が低下していると考えられる。

5.2　並列データマイニング

次に、実アプリケーションを用いて、複数 Initiator から単数 Target にアクセスしたときの IP-SAN 統合型 PC クラスタの評価を行った。HPA アルゴリズムと PFP アルゴリズムの並列データマイニングプログラムについて、アイテム数を 1000 とし、トランザクション数が 1M、2M、4M、8M のトランザクションデータを用い、最小サポート値を 0.7 %として実行した。プラットフォームとしては、先の Bonnie++実験時と同じ環境を用いた。

図 8 に HPA アルゴリズム、図 9 に PFP アルゴリズム実行時の各クラスタにおける実行時間を示す。

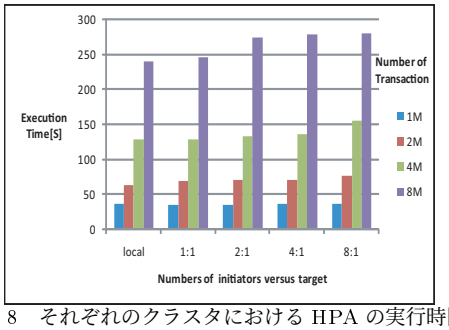

図 8　それぞれのクラスタにおける HPA の実行時間

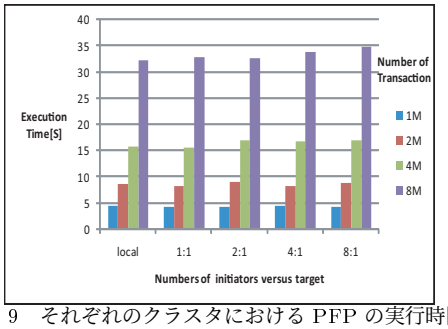

図 9　それぞれのクラスタにおける PFP の実行時間

この結果から、HPA、PFP どちらのアルゴリズムにおいても、どの接続方式においても実行時間はほとんど変わらないことがわかる。

ネットワークトラフィックのモニタリングの結果から、どちらのアルゴリズムにおいてもネットワークの帯域にはまだ余裕があることがわかる。iSCSI を用いた場合も、iSCSI のトラフィックはネットワークにあまり大きな影響を与えておらず、バックエンドネットワークをフロントエンドネットに統合してもネットワークの帯域を完全に使い切ることはない。

CPU 使用率のモニタリングの結果から、HPA においては何回もデータベーススキャンを繰り返し行うため、Initiator の

CPU 使用率が高くなっていることがわかる。PFP においては、計算量が少ないため、あまり Initiator の CPU が使われていない。

以上の結果から、HPA アルゴリズムと PFP アルゴリズムのどちらの場合においても、IP-SAN 統合型 PC クラスタにおいて Target にアクセスを集中させた場合の性能はほとんど変わらないということがわかった。また、ネットワークトラフィックのモニタリングにより、今回の実験では IP-SAN のトラフィックを統合してもネットワークの帯域にまだ十分余裕があることがわかった。これらは、今回使用した HPA のプログラム、PFP プログラムどちらの場合においても、ノード間通信やストレージアクセスといった処理だけでなく、これは計算処理が主に行われており、ノード間通信とストレージアクセスを行うパケットの衝突がネットワーク上であまり起こることがないためであると考えられる。

5.3 複数プロセス

次に、複数プロセスを用いて、複数 Initiator から単数 Target にアクセスしたときの IP-SAN 統合型 PC クラスタの評価を行った。bonie++を実行させながら HPA アルゴリズムと PFP アルゴリズムの並列データマイニングプログラムについて、アイテム数を 1000 とし、トランザクション数が 1M、2M、4M、8M のトランザクションデータを用い、最小サポート値を 0.7 %として実行した。プラットフォームとしては、先の実験時と同じ環境を用いた。

図 10++を実行させたながら HPA プログラムを実行させたときの実行時間を、図 11 に bonnie++を実行させたながら PFP プログラムを実行させたときの実行時間を示す。

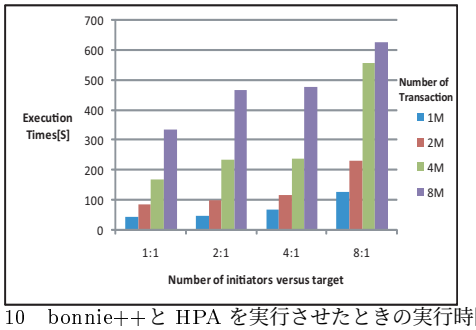

図 10　bonnie++と HPA を実行させたときの実行時間

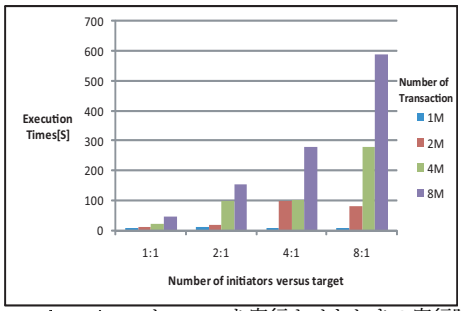

図 11　bonnie++と PFP を実行させたときの実行時間

先ほどの実験と異なり、HPA と PFP どちらの場合においても、高負荷をかけたときに実行時間が長くなっている。

図 12 に bonnie++を実行させながら HPA を実行させたときの Target のネットワークトラフィックを示す。

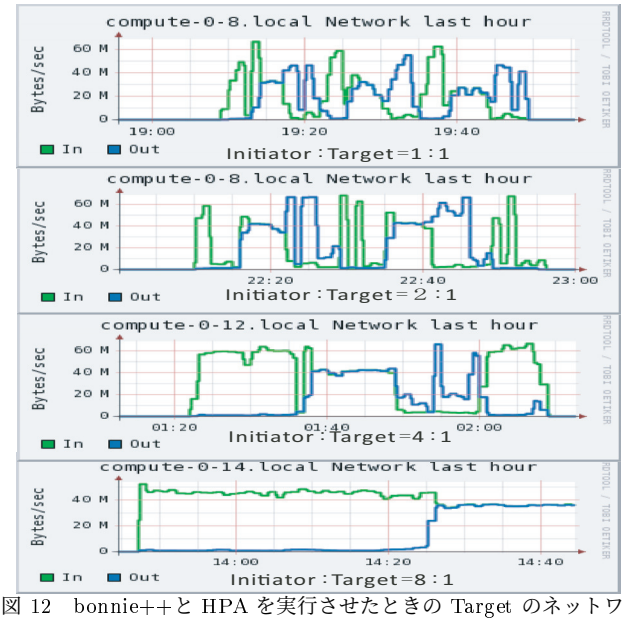

図 12　bonnie++と HPA を実行させたときの Target のネットワークトラフィック

この結果から、先の実験と同様に、Target のネットワークスループットは最大 60Mbytes/sec (約 500Mbps) であり、ネットワーク帯域にはまだ余裕があることがわかる。したがって、ストレージアクセスとデータマイニングを同時に実行することで高負荷をかけても、ネットワークの性能には影響を与えることはない。

図 13 に Initiator の CPU 使用率、図 14 に Target の CPU 使用率、図 15 に Iniatiator と Target を 8 対 1 に接続させたときのストレージ I/O をそれぞれ示す。

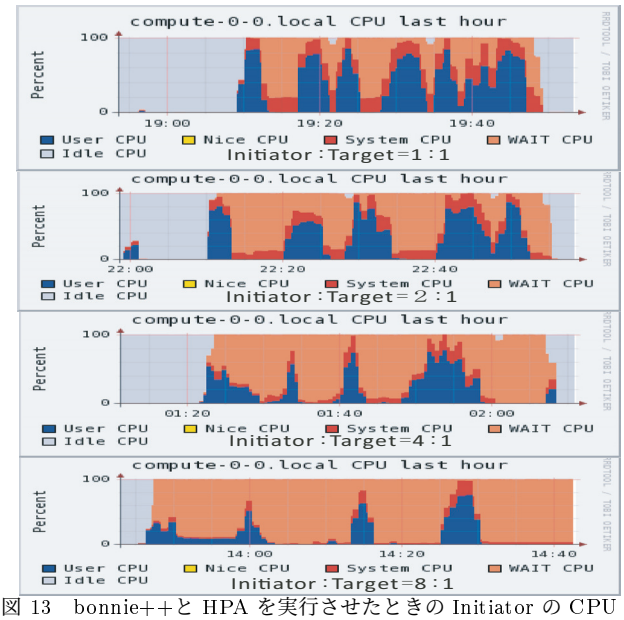

図 13　bonnie++と HPA を実行させたときの Initiator の CPU 使用率

図 13 に示すように、単数 Initiator から単数 Target にアクセスしたときは、Initiator の”USER CPU”が高くなり、ストレージアクセスとデータマイニングが忙しく実行されていることがわかる。そのときの Target の CPU は図 14 から I/O レス

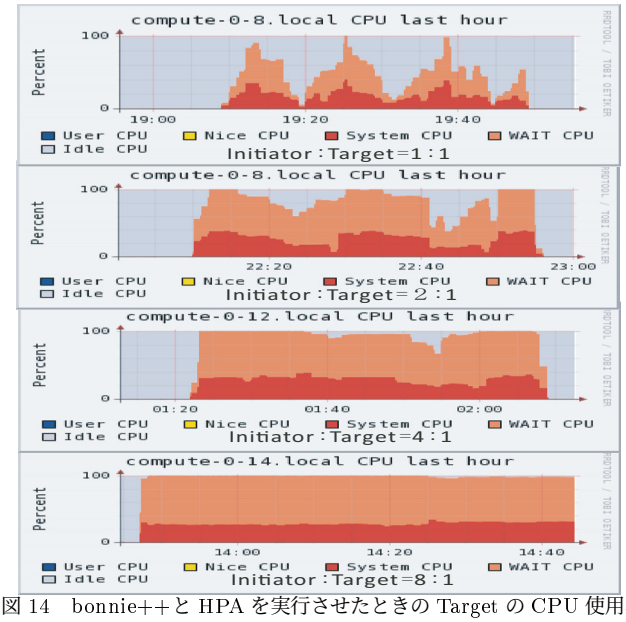

図 14　bonnie++と HPA を実行させたときの Target の CPU 使用率

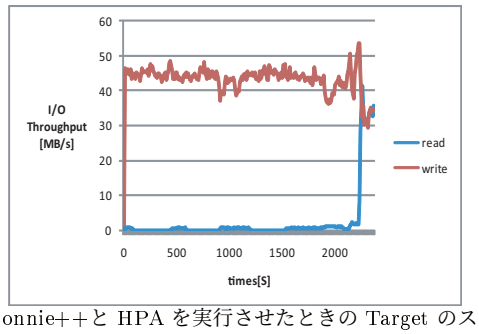

図 15　bonnie++と HPA を実行させたときの Target のストレージ I/O (Initiator:Target = 8 : 1)

ポンス待ち状態であると考えられる。

それに対して、単数 Target に接続する Initiator の数が増えていくと、Initiator も Target もほどんとが”WAIT CPU”となっており、図 15 の結果からこの場合はストレージ負荷が高くなっていることがわかる。

これらの結果から、Target におけるストレージアクセスが軽い場合は、Initiator の CPU がボトルネックとなり、Target にストレージが集中した場合は、Target のストレージ I/O がボトルネックとなる。結果として、iSCSI のトラフィックを統合してもネットワークには影響を与えないということが分かった。。

6. まとめと今後の課題

本稿では、複数 Initiator から単数 Target にアクセスすることでネットワークと Target に高負荷をかけ、ハードディスクベンチマーク、データマイニングアプリケーションおよびそれらの両方を実行することで、IP-SAN 統合型 PC クラスタの評価を行った。実行時間に加えて、アプリケーション実行時のネットワークトラフィック、CPU 使用率、ストレージ I/O をモニタリングを行った。

ハードディスクベンチマークの結果から、sequential read アクセスにおいては、ネットワークと Target に高負荷をかけたときもスループットは低下せず、sequential write アクセスにおいては、複数 Initiator から単数 Target にアクセスをしたときに少しスループットが低下した。ネットワークと CPU がボトルネックになっていないことから、ストレージ I/O が性能低下の原因であると考えられる。しかしながら、IP-SAN 統合型 PC クラスタの I/O の性能はかなり高いと言える。

並列データマイニングを実行した場合、iSCSI ストレージアクセスのトラフィックとノード間通信のパケットとの間のネットワークにおける衝突が起こると考えられたが、複数 Initiator から単数 Target にアクセスを行った場合でも性能は同じであった。データ処理アプリケーションでは、ストレージアクセスだけでなく、計算処理も行われるため、ネットワーク衝突があまり起こらないためであると考えられる。したがって、IP-SAN 統合型 PC クラスタはデータ処理アプリケーションを実行するときに性能が低下しない。

ハードディスクベンチマークを実行しながら並列データマイニングを実行した場合、特にストレージに高負荷をかけた場合において、並列データマイニングのみを実行したときよりも実行時間が長くなった。この場合の IP-SAN 統合型 PC クラスタは、ネットワークバウンドではなく、CPU バウンドもしくは I/O バウンドであった。。

以上から、今回の実験ではどの場合においても、当初懸念されたネットワークバウンドにはならず、IP-SAN 統合型 PC クラスタは有効であり、その柔軟なシステム構成を実現することが可能であると言える。今後の課題としては、OS に対してカーネルモニタなどの導入を行うことで、IP-SAN 統合型 PC クラスタのシステム内部の振舞を詳細に解析し、最適化を行っていきいと考えている。

謝　辞

本研究は一部，文部科学省科学研究費特定領域研究課題番号 18049013 によるものである．

文　献

[1] iSCSI RFC: http://www.ietf.org/rfc/rfc3722.txt
[2] D. Xinidis, A, Bilas, and M. D. Flouris, “Performance Evaluation of Commodity iSCSI-based Storage Systems,” MSST2005, April 2005.
[3] A. Joglekar, M. E. Kounavis, and F. L. Berry, “A Scalable and High Performance Software iSCSI Implementation,” USENIX FAST 2005, pp.267-280, December 2005.
[4] B. K. Kancherla, G. M. Narayan, and K. Gopinath, “Performance Evaluation of Multiple TCP connections in iSCSI,” MSST2007, September 2007.
[5] R. Agrawal, T. Imielinski, A. Swami: “Mining Association Rules Mining between Sets of Items in Large Databades,” VLDB1994, pp.487-499, September 1994.
[6] J. Han, J. Pei, and Y. Yin: “Mining Frequent Patterns without Candidate Generation,” ACM SIGMOD2000, pp.1-12, May 2000.
[7] Iko Pramudiono and Masaru Kitsuregawa: “Tree structure based Parallel Frequent Pattern Mining on PC cluster,” DEXA2003, pp.537-539, September 2003.
[8] Rocks Cluster: http://www.rocksclusters.org/
[9] Ganglia Monitoring System: http://www.ganglia.info/
[10] Open-iSCSI: http://www.open-iscsi.org/
[11] iSCSI-Enterprise Target: http://sourceforge.net/projects/iscsitarget/
[12] Bonnie++: http://www.coker.com.au/bonnie++/