社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－EVA－16（8）
2006／3／20

DIMMnet-2 NICのTCP/IPプロトコルスタック用ネットワークドライバの実装と評価

森 拓郎† 荒木 健志† 金井 遵† 中條 拓伯† 並木 美太郎†

本論文ではクラスタ向けに開発されたNICであるDIMMnet-2を通信路として用いるネットワークドライバについて研究する。DIMMnet-2はPCクラスタを構築する際のボトルネックを解消し広帯域で低遅延な通信を実現するNICである。しかし、ユーザはプログラム中にハードウェア操作のコードを書かねばならず、作業が煩雑となるため、ソケットを用いたDIMMnet-2利用という要求が発生する。そこで、プロトコルスタックからの呼び出しでDIMMnet-2 NICを用いた通信を行うIPドライバを作成した。また、メッセージ受信処理の効率化を行うためにタイマ割り込みをフックしたポーリング機構を作成した。本論文では、このIPドライバの検証と今後の有効性について述べる。

Implementation and Evaluation of a Network Driver for the TCP/IP protocol stack using "DIMMnet-2" NIC

Takuro Mori† , Takeshi Araki† , Jun Kanai†
Hironori Nakajo† and Mitaro Namiki†

This paper discribes the network driver using "DIMMnet-2" NIC for cluster's channel . "DIMMnet-2" is designed as wideband and low latency NIC resolving bottleneck of communication for PC cluster. However, the work becomes complex because the user must write the code of the hardware operation in user program. Therefore, demand for the socket-base communication using "DIMMnet-2" comes up. So the IP driver was developed as a communication that uses "DIMMnet-2" NIC and protocol stack's. In addition, the hook of timer interrupt feature at polling was developed for making the efficient message processing. In this reseach, the evaluation of the IP driver and the effectiveness in the future are described.

1 はじめに

PCクラスタシステムは安価に構築可能なハイパフォーマンスコンピューティング環境として用いられている。近年のPCにおけるCPUの性能の向上はめざましく、また光通信や高速シリアル通信の登場により相互結合網で使用できる帯域幅は拡大してきている。これらを反映させることによってより演算性能の高いクラスタシステムが構築可能であると考えられる。

しかし、PCIバスを用いたネットワークインタフェースがボトルネックとなりこれらの帯域幅を十分に活用できないのが現状である。

DIMMnet-2[1]はPCIバスを用いたネットワークインタフェースの問題を解消するために開発されたネットワークインタフェースである。DIMMnet-2はDIMMスロットに搭載され、通信にメモリバスを用いることによってPCIバスで発生していた問題を解決している。

しかしこのDIMMnet-2を用いる際には、ユーザがプログラム中にパケットの作成やハードウェアレジスタ制御、待ち合わせの処理などを記述する手法が用いられてきた。これは性能を発揮させるため行われているが、作業が煩雑となるため、このような作業を行わずにソケットベースでDIMMnet-2を利用したいという要求が発生する。

そこで本研究では、DIMMnet-2を用いたネットワークドライバを開発する。作成されたネットワークドライバにおいて、DIMMnet-2の利点を用いた処理方式やボトルネックの改善方法について述べる。

2 DIMMnet-2の概要

DIMMnet-2はメモリスロットであるDIMMスロットに搭載されるNICである。通信におけるコントローラチップには、RHiNETにも搭載されて

† 東京農工大学工学教育部
Graduate School of Technology, Tokyo University of Agriculture and Technology

いる Martini[2] が用いられている。

また、DIMMnet-2 用パケットはハードウェアによって Infiniband パケットにカプセル化されて送出される。これにより、低遅延な Infiniband 環境および Infiniband スイッチを用いることができる。

さらに DIMMnet-2 は、PCI バスを用いないことにより既存の NIC ではボトルネックとなっていた DMA による衝突等の問題を解決している。

DIMMnet-2 において、ホストからネットワークインタフェース上の SO-DIMM を直接アクセスを行うと、メモリバスへの負荷が大きくなってしまい、メモリバスを高周波で駆動させることが困難であると予想される。そこで、負荷の増加を防ぎ、高周波化を実現しするためにネットワークインタフェース上の SO-DIMM へのアクセスはコントローラ内部のバッファを経由して行う間接アクセスとなっている。この際書込みに用いられるバッファを WriteWidndow(1[KB])、読込みに用いられるバッファを PrefetchWindow(2[KB]) と呼ぶ。

また、リモートのネットワークインタフェース上の SO-DIMM への読み書きといった通信処理もコントローラにコマンドを書込むことによって発行されるため、ローカルとリモートの双方のネットワークインタフェース上の SO-DIMM へのアクセスに統一された手法を提供することが可能となっている。

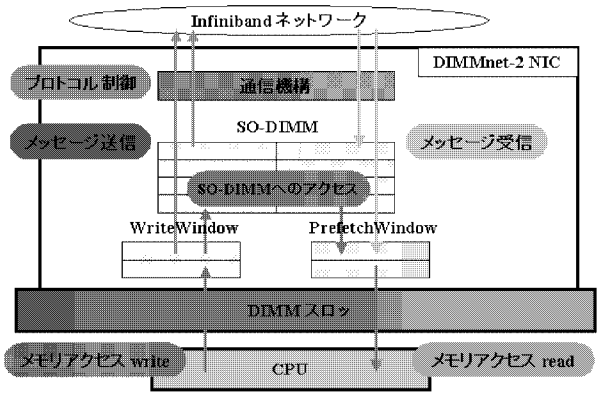

図 1: DIMMnet-2 概要

しかし、DIMMnet-2 を利用するためにはプログラム中にコマンドを書き込まなければならないため、ユーザはアプリケーションを用いる際にそれを DIMMnet-2 専用に書き換える必要があり、作業が煩雑となってしまう問題点がある。

また、DIMMnet-2 は DIMM スロットに搭載されている。このため、メッセージの受信などをハードウェアが検知しても OS に対して割込みを発生させることができない。

3 目標と設計方針

3.1 目標

本研究の目標は既存アプリケーションを変更することなく DIMMnet-2 環境で利用することである。そのために、DIMMnet-2 向けネットワークドライバを作成する。これにより、DIMMnet-2 環境で既存のソケットを用いたアプリケーションを変更することなく利用可能となり、TCP/IP や UDP/IP といったプロトコルによる通信を行うことができる。

本研究におけるネットワークドライバのインタフェースは DIMMnet-2 利用環境である Linux のもの [3] に基づき、一般的な NIC と同様の方法で利用可能とする。DIMMnet-2NIC は一般的な NIC と比較すると、接続方式や通信方式など相違点は数多く存在するが、これをドライバで吸収する。Eicken ら [4] の研究では、不要なカーネルを媒介しないことによりオーバヘッドの低減に成功したが、ユーザプログラム側に手を加える必要があった。本研究では応用プログラムに手を加えずすべてドライバに吸収させることを目的としているため、このような方式は用いない。

また、DIMMnet-2NIC はクラスタ向け NIC として開発され、その帯域幅は実測値で約 700[MB/s] と高速なものを実現してる。本研究で作成するネットワークドライバでは、この帯域幅を可能な限り利用することに重点を置く。

3.2 設計方針

作成する DIMMnet-2 向けネットワークドライバの設計方針について述べる。まず、前述のようにメモリスロットに搭載する DIMMnet-2 では割込みを発生してメッセージの到着を検知することができないためポーリングを用いる。その際の応答時間の悪化や帯域幅の減少を防ぐよう設計を行う。

また、DIMMnet-2 は 512[MB] 用意されている SO-DIMM 内の任意の領域を送受信バッファとして用いることができる。そのため、バッファを活用することによって連続した送受信においても性能を低下させることなく通信が行える可能性がある。

4 全体構成

ネットワークドライバは処理を以下の 4 つに分類し、それぞれ処理を行わせる。

(1) プロトコル処理部
(2) ハードウェア処理部

(3) 外部入力処理部
(4) ポーリング処理部

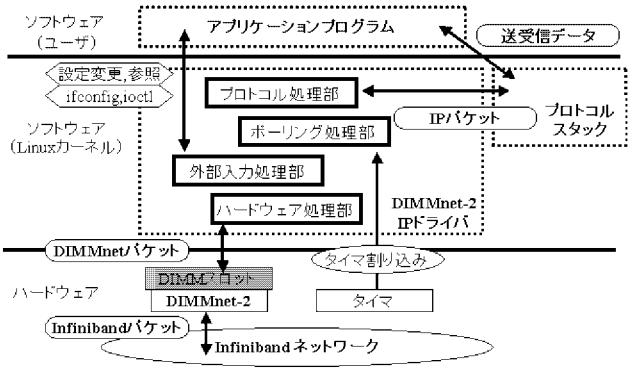

図 2: ドライバ全体構成

(1) のプロトコル処理部はプロトコルスタックとのやり取りを行う部分である。パケット送信においては、プロトコルスタックから与えられる IP パケットを DIMMnet-2 で用いるパケットへとカプセル化し、ハードウェア処理部へと渡す。また受信の際にはハードウェア処理部から DIMMnet-2 用パケットを受け取り、それから IP パケットを取り出してプロトコルスタックへと渡す。

(2) のハードウェア処理部は DIMMnet-2 NIC を用いる部分である。プロトコル処理部から渡された IP パケットを DIMMnet-2 パケットへとカプセル化し、DIMMnet-2 に送信を行わせる。また、受信時はハードウェアから受信パケットを取り出す作業を行う。

(3) の外部入力処理部はアプリケーションからの呼び出しを処理する部分である。ioctl や ifconfig によって設定の変更や参照が要求された際にそれを実行する。

(4) のポーリング処理部はタイマ割り込みによるポーリングの処理を行う。パケットの受信を検知した場合は、そのデータの取り出しをハードウェア処理部に要求する。

5 設計

5.1 ポーリング方式

一般のポーリング方式では、ハードウェアタイマからの割り込みでカーネル関数であるタイマ割込み処理関数が実行され、そこからドライバのポーリング用関数が呼び出される。この割込み間隔は一般には 10[ms] で、カーネルの改変により短縮が可能である。ここで問題となるのは、タイマ割り込み処理関数である。この関数はオーバヘッドが大きく、ハードウェアタイマの呼び出し間隔を短

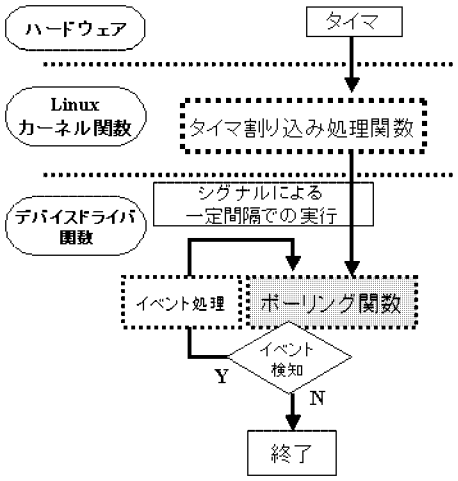

図 3: 一般のポーリング方式

くしてポーリング間隔を短くしようとした場合、さらにオーバヘッドが増大してしまう。

そこでタイマ割込みをフックする方法が考えられる。割込みで呼び出される関数をポーリングを行う関数に変更し、その関数内で時間を測定し 10[ms] ごとに元のタイマ割込み処理関数を呼び出す。これにより、タイマ割込みの間隔を短くしてもオーバヘッドの増加を抑えつつポーリングの処理が行える。

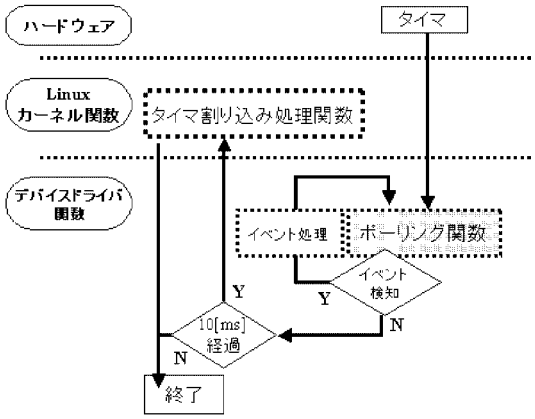

図 4: タイマのフックを用いる方式

5.2 ポーリング間隔

ポーリング間隔には、TCP/IP により送信されるメッセージが最大の場合においてそれを処理できる間隔が必要である。そのため、以下のデータを参考とする。DIMMnet-2 の通信性能は最大 1[GB/s] であり、TCP のスライディングウィンドウのサイズは最大で 64[KB] である。つまり、TCP のスライディングウィンドウのメッセージが全て到着す

るまでに費やされる時間は 64[μs] となる。本研究では、TCP/IP を最大限用いた通信に対するポーリング間隔として、この半分の 32[μs] 程度の間隔があれば十分と考える。

5.3 IP パケットのカプセル化

本ネットワークドライバはプロトコルスタックから IP パケットを受け取る。このパケットを DIMMnet-2 を用いて Infiniband ネットワークへと送出するが、DIMMnet-2 ハードウェアの MTU は本研究で用いたものは 512[B] と IP パケットの最大サイズである 64[KB] と比べて小さい。そのため、IP パケットはドライバ中で分割し、ドライバ用のパケットとする。ドライバはこのドライバ用パケットを DIMMnet-2 用パケットにカプセル化し、DIMMnet-2 へ送信要求を行う。DIMMnet-2 NIC は受け取ったパケットを Infiniband パケットにカプセル化して Infiniband ネットワークへと送出する。

5.4 受信バッファとしての SO-DIMM

DIMMnet-2 はパケットの送信を行う際、その受信するメモリ領域をオフセットを用いることで指定することができる。これを用い、パケットは SO-DIMM メモリ上に並ぶように送信する。この領域のサイズは、SO-DIMM のサイズである 512[MB] を DIMMnet-2 の最大ノード数である 256 で割った値である 2[MB] が最小でも与えられる。これは前述のポーリング間隔と照らし合わせても十分な値である。

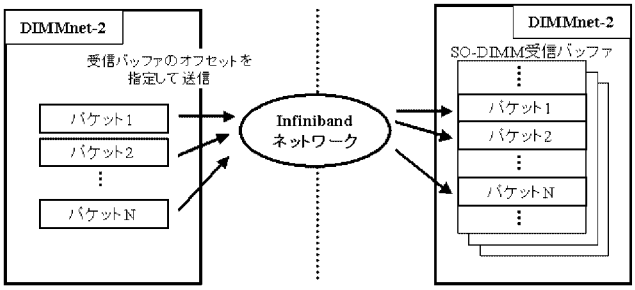

図 5: SO-DIMM 領域の受信方式

6 評価

作成したネットワークドライバを用いて評価を行った。評価には 2 台の PC を用い、対向通信によって性能を測定した。CPU に Pentium4 2.6[GHz]、OS に RedHatLinux9 を搭載している。また、現在の DIMMnet-2 搭載メモリへのアクセスは、書き込みで 69.3[MB/s]、読み込みで 17.0[MB/s] である。これはハードウェアのバグによるものであり、本来は読み込み、書き込み共に、DDR-RAM と同程度の 3.6[GB/s] でのアクセスを想定している。

6.1 TCP/IP による評価

プロトコルに TCP/IP を選択し、その通信性能を測定して評価を行う。割り込み間隔は 20[μs] とし、256[MB] のデータを分割して送信した。一度に送信するメッセージのサイズは 1[KB] から増加させて測定し、MTU は TCP スライディングウィンドウのサイズと同様の 64[KB] とし、これを変化させて性能を測定した。また、一台のマシンから一度に作られるコネクション数を 1〜16 と変化させ、コネクション増加に伴う性能の変化を測定した。

また、その通信性能について予測を行う。DIMMnet-2 の通信性能は実測値で約 700[MB/s] となることが分かっているが、現在はハードウェアのバグで到着したパケットを読出す部分のメモリアクセス速度が最大で 17[MB/s] と低速で、受信のオーバヘッドが大きいと考えられる。これにより、通信速度よりも遅いこの読出しがボトルネックになる可能性が高く、その場合、メモリ速度や通信速度などから計算すると本来 270[MB/s] 程度と予測される帯域幅は最大でも 10[MB/s] 程度まで減少すると予測される。

6.1.1 1 コネクションにおける通信性能

前述の環境で 1 コネクションでメッセージサイズを変化させてその通信性能を測定した。

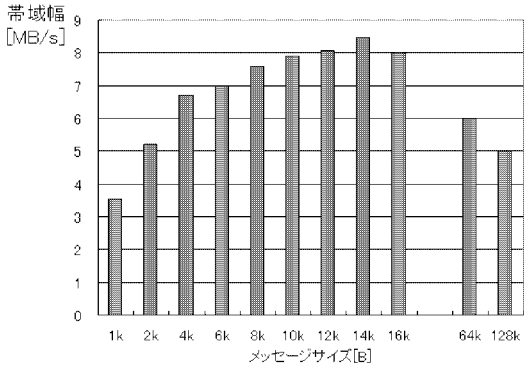

図 6: 1 コネクションにおける通信性能

帯域幅はメッセージサイズ 10[KB] 付近から 8[MB] 程度となり、メッセージサイズの増加においてもほとんど上昇せず、1[GB/s] という DIMMnet-2 の通信性能と比べると非常に低速な結果となった。ドライバ内部で消費した時間を調べると、受信データ取り出しに大部分の時間を消費していることが分かった。予測した通り受信バッファのバグによる性能低下がドライバの通信性能に大きく影響している。このバグが改善された場合、アプリケーション

から見た帯域は、通信時間やメッセージのコピーおよびユーザプロセスへの切替えなどのオーバヘッドより、メッセージサイズが 64[KB] で 270[MB/s] の性能が、16[KB] で 141[MB/s]、8[KB] で 84[MB/s] 程度となると予測される。一度に大量のパケットを送付しても受信側 SO-DIMM に十分容量があるため、メッセージサイズが大きいほうがより DIMMnet-2 の帯域を利用できると考えられる。

6.1.2 MTU の変化における通信性能

1 コネクションでメッセージサイズ 8[KB] の通信において、その MTU を 1[KB] まで減少させて性能の変化を測定した。MTU が減少することによって送信するデータは細かいパケットへと分割される。そのため通信回数が増加してオーバヘッドは増加するが、前述のメモリアクセスのバグによるオーバヘッドにより目立たず、帯域幅があまり減少していない。メモリアクセスのバグが改善された際には、前述の帯域幅 84[MB/s] と比べ、MTU が 4[KB] で 69[MB/s]、1[KB] で 33[MB/s] 程度まで、通信回数の増加により帯域幅が減少すると予測される。

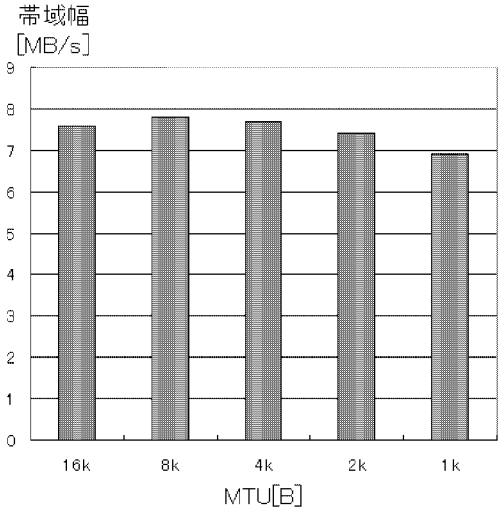

図 7: MTU の変化における通信性能

6.1.3 ギガビット Ethernet との比較

コネクション増加時の性能を評価するため、ギガビット Ethernet で同様の通信を行いその性能を測定した。Ethernet NIC は PLANEX GN-1200TW を用いた。DIMMnet-2 はハードウェアが 8[Gbps] で動作しているため、帯域幅で 8 倍の性能を発揮できれば本方式の有効性が確認されると考えられる。結果においては、ソケットプログラムが一度に送信を行うサイズをメッセージサイズとし、顕著に差異が見られたメッセージサイズが 1[KB] の場合と 64[KB] の場合について示す。

表 1: コネクションごとの帯域幅 [MB/s]

コネクション数	DIMMnet(1k)	Ethernet(1k)
1	3.58	7.16
2	3.03	6.91
4	2.60	5.90
8	2.46	4.15
16	2.36	2.57
コネクション数	DIMMnet(64k)	Ethernet(64k)
1	8.01	43.74
2	4.53	27.67
4	3.60	21.18
8	3.49	19.42
16	3.37	10.53

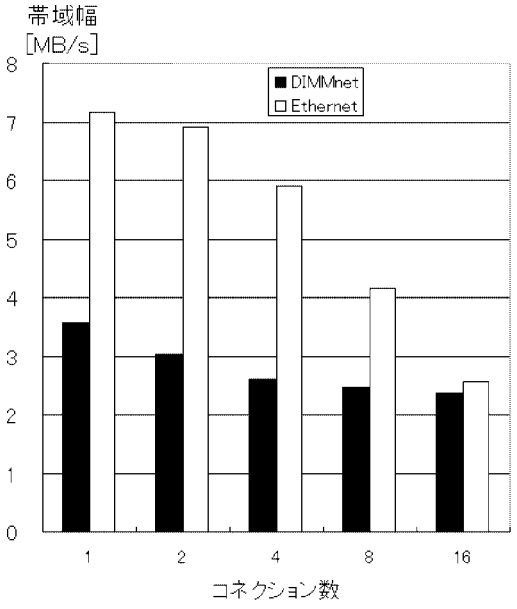

図 8: メッセージサイズ 1[KB] での性能

メッセージサイズ 1[KB] において、Ethernet はコネクションの増加に伴って 35[%] 程度まで大きく帯域幅が減少しているのに対して、DIMMnet-2 は 65[%] の減少に留まっている。これは、DIMMnet-2 がパケットを受信する際に SO-DIMM に受信した後にポーリングで一括して処理を行うため、メッセージサイズの増加による影響が小さかったと考えられる。結果として、ハードウェアがバグを抱えている現状においてもコネクション数 16 においては Ethernet に匹敵する性能を発揮した。

メッセージサイズ 64[KB] においては、1[KB] の際の結果とは違い、大きく性能の低下が見られた。これは受信バッファが大きくなったことにより、メッセージ読み出しのオーバヘッドが増大したためである。現在はハードウェアのバグでこのオーバヘッドが大きいが、改善された際には SO-DIMM から取り出された受信パケットに対して RAM と

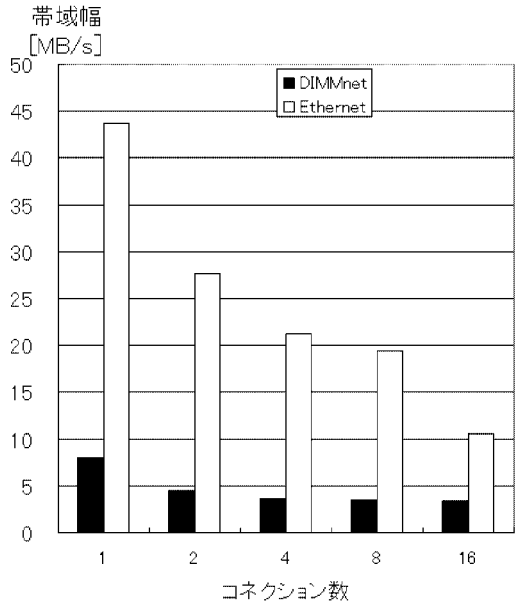

図 9: メッセージサイズ 64[KB] での性能

同様にアクセスが行えるため、コネクション数の増加に対して帯域幅は 270[MB/s] からほとんど低下しないと予測される。そのためコネクション数 2 以上では 1[Gbps] の Ethernet に対して 8[Gbps] の DIMMnet-2 が 8 倍以上の性能を発揮でき、有効性が確認できると考えられる。

6.1.4 バグ改善時の性能予測

得られたデータより、現在のバグが改善された際の帯域幅は、大量のメッセージ送信や複数コネクションによって TCP のスライディングウィンドウ 64[KB] を使い切る状態で最大となり、270[MB/s] 程度であると予測される。この際のネットワークドライバの MTU は 64[KB] で動作し、コネクションの数に対して帯域幅の低下はほとんど起きないと考えられる。

7 考察

バグが改善された際の性能の予測をギガビット Ethernet の結果と比較すると、1 コネクションでの性能はメッセージサイズ 64[KB] と、TCP のスライディングウィンドウを使い切るような通信の頻発する状態で 6.2 倍と、DIMMnet-2 とギガビット Ethernet のハードウェア的な帯域の比である 8 倍には及ばなかった。しかし、コネクション数が複数となった際の性能は 2 コネクションで 9.8 倍、16 コネクションで 10.5 倍と、SO-DIMM の大容量なバッファを生かした処理が行えると考えられる。

これらのことから、DIMMnet-2 を既存のアプリケーションで用いる場合、複数のコネクションを用いたり、大量のデータを通信する環境において、広帯域の通信性能と大容量の受信バッファが効果的と考えられる。例として以下のものが有効と考えられる。

(1) OpenMOSIX による分散並列実行環境
(2) Apache などのサーバプログラム
(3) MPI による分散並列処理

(1) の OpenMOSIX はカーネルに修正を加えて複数マシンでの分散並列処理の実行環境を提するものである。マシン数が増大した場合や大規模な演算を行う環境で DIMMnet-2 の通信性能が生かされると考えられる。

(2) のサーバプログラムにおいては、DIMMnet-2 の複数コネクションからのアクセスに対して強い特性から、クライアントの増加に対しても安定した帯域幅での通信が期待できる。

(3) の MPI は、並列分散環境でよく用いられる通信手法である。大規模な並列計算において大量のデータをやり取りする際は、大容量な SO-DIMM を用いて受信を行う DIMMnet-2 の性能が発揮できると考えられる。

8 おわりに

本研究では DIMMnet-2 用ネットワークドライバを作成し、TCP/IP で用いることによってその有効性を発見した。現在ハードウェアが持っているバグを改善することによって既存のアプリケーションにおいて有効な性能を発揮することができると考えられる。

参考文献

[1] 北村聡 伊豆直之 田邊昇 濱田芳博 中條拓伯 渡邊幸之介 大塚智宏 天野英晴, DIMMnet-2 ネットワークインタフェースボードの試作 情報処理学会研究報告 ARC-159(SWoPP'04) pp.151-156, 2004.

[2] 山本 淳二 田邊 昇ほか. 高速性と柔軟性を併せ持つネットワークインタフェース用チップ : Martini 情報処理学会研究報告 ARC-140 pp.19-24, 2000.

[3] Alessandoro Rubini Jonathan Corbet 著 山崎康宏 山崎邦子 長原宏治 長原陽子 訳 Linux デバイスドライバ オーム社, 2002.

[4] T. von Eicken, A. Basu, V. Buch, and W. Vogels U-Net: A User-Level Network Interface for Parallel and Distributed Computing. Proc. of the 15th ACM Symposium on Operating Systems Principles, 1995.