過去のNIC負荷とプロセスのデータ送信量を考慮した複数NIC間での負荷分散法

谷口 秀夫[1,a)]　吉田 泰三[1]　山内 利宏[1,b)]　佐藤 将也[1,c)]

概要：通信路の通信速度の向上に伴い、10Gbps 以上の高速な通信機器が開発されている。しかし、高速な通信機器は、高価である。そこで、廉価な複数の Network Interface Card（以降、NIC と略す）を連携制御し、1 台の計算機からデータを送信することにより、高速な通信が可能である。FreeBSD や Linux は、複数の NIC を仮想的に 1 つに見せて、NIC 間の負荷分散を行いデータを送信できる。しかし、これらの方法は、CPU 使用量が多い、あるいは複数のデータ送信プロセスのデータ送信処理を複数の NIC へうまく負荷分散できない問題がある。そこで、本稿では、過去の NIC の負荷状態とデータ送信プロセスのデータ送信量を考慮した複数 NIC 間での負荷分散法を述べる。提案法は、NIC の I/O 性能を要因としてデータ送信プロセスのデータ送信量が抑制されているか否かを確認し、この結果に応じてデータ送信プロセスと NIC の対応付けを変更する。

1. はじめに

通信路の通信速度は高速化し、10Gbps 以上の高速な通信機器が開発されている。しかし、高速な通信機器は、非常に高価であるため、これらを用いた通信環境の構築は容易でない。

そこで、1 台の計算機において複数の廉価な NIC を連携制御し、データを送信する手法 [1][2] が提案されている。これらの手法は、複数の NIC を仮想的に 1 つに見せて、NIC 間の負荷分散を行うことにより、データを送信している。複数の NIC を仮想的に 1 つに見せる手法は、大きくラウンドロビン分散方式と負荷分散方式に分類できる。ラウンドロビン分散方式は、送信パケットをラウンドロビンで NIC に振り分ける方式である。この方式は、ソフト割り込みに要する処理時間が長くなるため、CPU 使用量が多い。CPU 使用量が多いことで、CPU 処理がボトルネックとなり、NIC の性能を十分に活用できない。一方、負荷分散方式は、データ送信量に基づき、データ送信プロセスと NIC を対応付け、NIC 負荷の均一化を行う。データ送信プロセスと NIC を対応付けることにより、ソフト割り込みに要する処理時間を削減できるため、CPU 使用量が少ない。しかし、対応付け更新のタイミングによっては、データ送信プロセスを複数の NIC にうまく対応できず負荷を分散できない問題がある。

文献 [3] は、複数 NIC を連携制御する手法として Linux で実現されている Ethernet チャネルボンディング [2] の balance-rr モードと balance-tlb モードの定量的な評価結果を述べ、その問題点を明らかにしている。

本稿では、文献 [3] が明らかにした問題への対処として、複数 NIC 間での新しい負荷分散法を提案する。具体的には、提案法は、NIC の I/O 性能がボトルネック（過負荷）のためにデータ送信プロセスのデータ送信量が抑制されているか否かを確認し、この結果に応じてデータ送信プロセスと NIC の対応付けを変更する。つまり、過負荷の NIC に対応付くデータ送信プロセスのうち、最も優先度の低いデータ送信プロセスを最も余力のある NIC に対応付けを変更する。これにより、優先度が高いデータ送信プロセスのデータ送信量を増加でき，複数 NIC のスループットの向上が期待できる。また、提案法を評価した結果を報告する。

2. 複数の NIC を仮想的に 1 つに見せる負荷分散法の問題点 [3]

2.1 分散方式

複数の NIC を連携制御し、1 台の計算機からデータを送信することにより、高速な通信が実現されている。

文献 [1][2] の手法は、複数の NIC を仮想的に 1 つに見せて、NIC 間の負荷分散を行うことにより、データを送信す

[1] 岡山大学 大学院自然科学研究科
Graduate School of Natural Science and Technology, Okayama University, 3-3-1 Tsushima-naka, Kita-ku, Okayama, 700-8530, Japan.
[a)] tani@cs.okayama-u.ac.jp
[b)] yamauchi@cs.okayama-u.ac.jp
[c)] sato@cs.okayama-u.ac.jp

る。複数の NIC を仮想的に 1 つに見せる負荷分散法は、大きく以下の 2 つに分類できる。

(1) ラウンドロビン分散方式

ラウンドロビン分散方式は、送信パケットをラウンドロビンで NIC に振り分ける。この方式の例として、FreeBSD の Lagg 機能 [1] における Round-robin モード、Linux の Ethernet チャネルボンディング [2] における balance-rr モードがある。

(2) 負荷分散方式

負荷分散方式は、データ送信量に基づきデータ送信プロセスと NIC を対応付け、NIC 負荷を均一化する。この方式の例として、Linux の Ethernet チャネルボンディング [2] における balance-tlb モードと balance-alb モードがある。

2.2 特徴

Linux の Ethernet チャネルボンディングは、両分散方式を実現している。このため、Ethernet チャネルボンディングの balance-rr モードと balance-tlb モードを対象とし、両分散方式を比較する。ここでは、データ送信処理に着目して比較するため、データ送受信処理の負荷分散を行う balance-alb モードを比較対象としない。

文献 [3] では、以下を明らかにしている。

(1) 100Mbps NIC を使用した場合、NIC の I/O 処理がボトルネックになるため、balance-rr モードと balance-tlb モードのスループットは、同程度である。

(2) 上記（1）において、balance-rr モードは、ソフト割り込みの発生回数が balance-tlb モードと比較して多く、CPU 使用量が多い（問題 1）。

(3) 複数の 1Gbps NIC を使用した場合、balance-rr モードでは、CPU 処理がボトルネックになる。また、両モードとも、複数の 1Gbps NIC を使用しているにも関わらず、スループットが NIC1 枚分のスループット未満であり，評価に使用した計算機のハードウェア性能がボトルネックになっているためと考えられる。

(4) balance-tlb モードは、データ送信プロセスと NIC の対応付けにおいて、NIC の過負荷がデータ送信量に与える影響を考慮していない。このため、うまく負荷分散できない場合がある（問題 2）。

3. 過去の NIC 負荷とプロセスのデータ送信量を考慮した負荷分散法

3.1 要求条件

2.2 節で述べたように既存の負荷分散法には、以下の 2 つの問題がある。

（問題 1）ソフトウェア割り込みが多発すると、CPU 使用量が増加する。

（問題 2）データ送信プロセスと NIC の対応付け更新のタイミングにより、データ送信プロセスのデータ送信量を NIC 対応付けに反映できない事態が発生する。

しがたって、新たな負荷分散法に対し、（問題 1）に対処するために、以下の要求がある。

（要求 1）データ送信プロセスと NIC の対応付けの変更回数を抑制する。これにより、ソフトウェア割り込みを抑制し、CPU 使用量の増加を抑える。

また、（問題 2）は、具体的には次のような原因で発生する。balance-tlb モードの場合、データ送信プロセスと NIC の対応付け解除は定期的に行われる。その後、新たなデータ送信要求に基づいて、データ送信プロセスと NIC を対応づける。この時、過去のデータ送信プロセスのデータ送信量を基に対応付けを決定する。このため、NIC の I/O 性能ボトルネックの影響によりデータ送信を抑制され、過去のデータ送信プロセスのデータ送信量が少なくなっている場合、この点を反映できない。そこで、以下の要求がある。

（要求 2）過去の状態において、NIC の I/O 性能ボトルネック情報（過負荷情報）、およびデータ送信プロセスのデータ送信量の抑制情報を考慮する。

3.2 NIC 対応付けの方法

データ送信プロセスと NIC の対応付け方法を**図 1** に示す。（A）は balance-tlb モードの場合、（B）は提案する方法である。

balance-tlb モードは、対応付けられた NIC がない場合、データ送信プロセスを最も余力のある NIC に対応付ける。提案法では、過去 T 秒間に一部の NIC が過負荷の場合、過負荷の NIC に対応付けられていたデータ送信プロセスのうち、最も優先度の低いデータ送信プロセスを最も余力のある NIC に対応付ける。これにより、優先度が高いデータ送信プロセスのデータ送信量を増加できる。提案法を（B）を用いて以下に説明する。

（処理 1）データ送信プロセス（以降、当該プロセスと略す）からのパケット送信要求が発生

（処理 2）TCP/IP のプロトコル処理を実行

（処理 3）当該プロセスと対応付けられている NIC があるか確認

（処理 4）過去 T 秒間に一部の NIC が過負荷であり、当該プロセスは過負荷の NIC に対応付いていたかを確認

（処理 5）上記（処理 4）を満たす場合、過去 T 秒間に当該プロセスは過負荷の NIC に対応付く最低優先度のプロセスであったかを確認

（処理 6）上記（処理 5）を満たす場合、当該 NIC に対応付いていた他プロセスの対応付けは未変更であるかを確認

（処理 7）上記（処理 6）を満たす場合、当該プロセスを過去 T 秒間で最も余力のあった NIC に対応付け

（処理 8）対応付けた NIC の余力から当該プロセスの過去 T 秒間の送信データ量を減算

（処理 9）上記（処理 4）を満たさない場合、当該プロセ

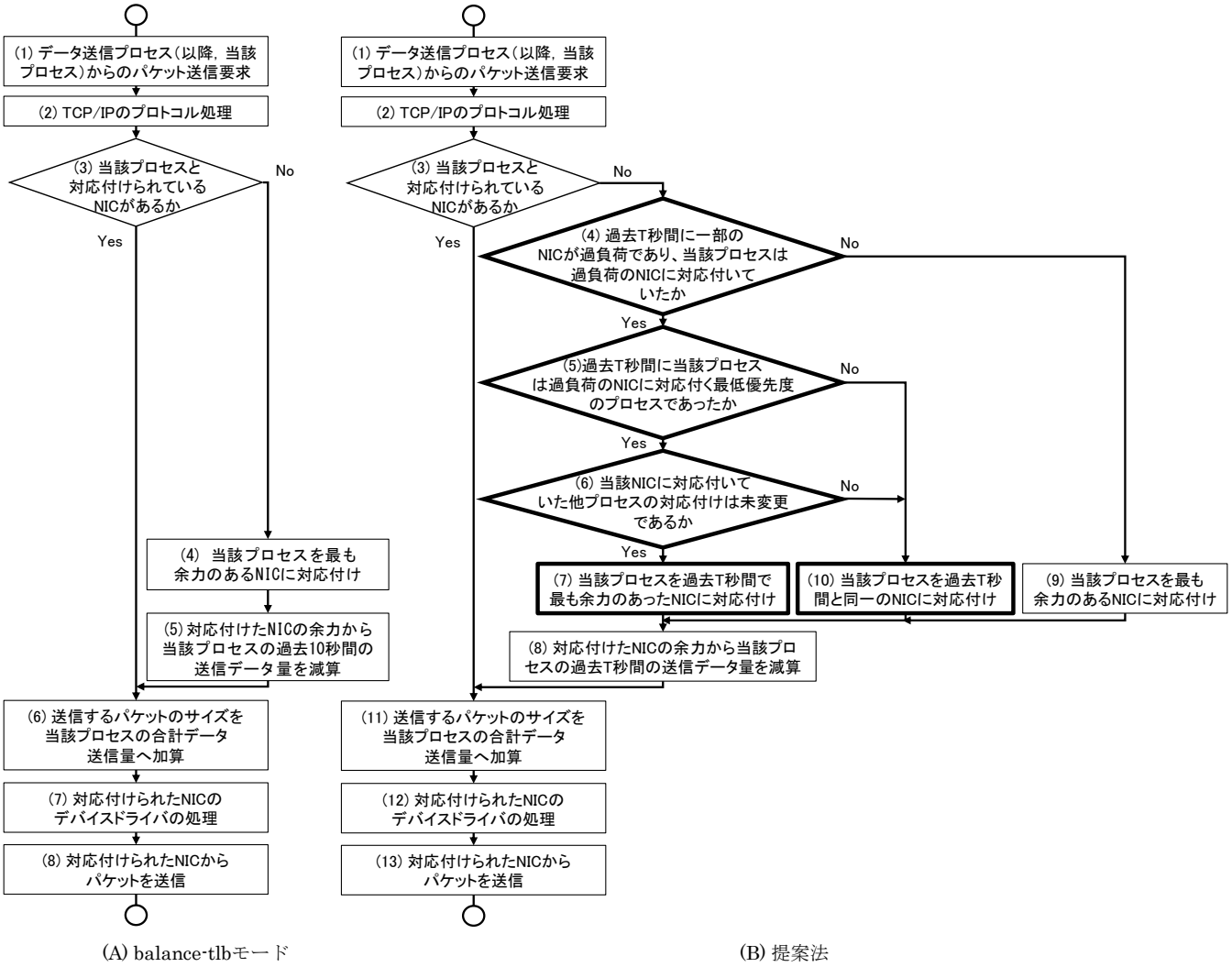


図 1 データ送信プロセスと NIC の対応付け方法

スを最も余力のある NIC に対応付け

（処理１０）上記（処理５）または（処理６）を満たさない場合、当該プロセスを過去 T 秒間と同一の NIC に対応付け

（処理１１）送信するパケットのサイズを当該プロセスの合計データ送信量へ加算

（処理１２）対応付けられた NIC のデバイスドライバの処理

（処理１３）対応付けられた NIC からパケットを送信

上記において、（処理４）は NIC の過負荷情報を利用しており、処理（５）と（6）により、最低優先度でないプロセスは同じ NIC に対応付け、最低優先度のプロセスを別 NIC に対応付けている。

3.3 期待される効果

提案手法は、主に以下の３つの効果を期待できる。

(1) データ送信プロセスと NIC を対応付けるため、CPU 使用量を抑制可能（(要求１）を満足）

balance-tlb モードと同様に、データ送信プロセスと NIC を 1 対 1 対応させるため、パケットごとに NIC を変更する balance-rr モードと比較して、ソフト割り込みの発生回数が減少し、ソフト割り込みに要する処理時間を低減できる。これにより、CPU 使用量を抑制できる。

(2) 各 NIC の性能を効率的に使用可能（(要求２）を満足）

複数のデータ送信プロセスが走行しており、一部の NIC が過負荷の場合は、当該 NIC に対応付く最も優先度の低いデータ送信プロセス 1 つを最も余力のある NIC に対応付けることで、各 NIC が過負荷にならないように制御する。これにより、各データ送信プロセスのスループットが NIC の過負荷により抑制されないように制御できる。

(3) 優先度の高いデータ送信プロセスのスループット低下を抑制可能

優先度の高いデータ送信プロセスと優先度の低いデータ送信プロセスが過負荷の NIC に対応付いている場合に、最も優先度の低いデータ送信プロセスを最も余力のある NIC に分離することにより、優先度の高いデータ送信プロセスのスループット低下を抑制できる。

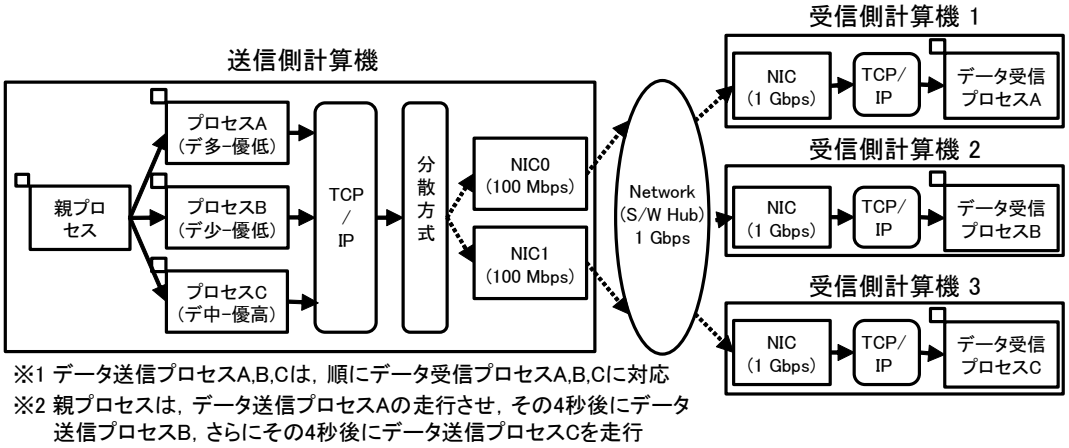


図 2　評価環境

表 1　評価に使用した計算機

計算機	CPU	メモリ	NIC
送信側	Pentium 4 (2.4 GHz)	1,024 MB	DE500-BA（100 Mbps）
			Intel PRO 1000MT
受信側	Celeron D (2.8 GHz)	256 MB	Intel PRO 1000GT

4. 評価

4.1 評価項目と条件

提案法について、スループット（データ送信量）と CPU 使用量を評価する。なお、balance-tlb モードは、10 秒ごとにデータ送信プロセスと NIC の対応付けを更新する。このため、提案法においても、T = 10 秒とした。また、NIC の過負荷状態は、I/O 性能の 90%以上を使用している状態とした。

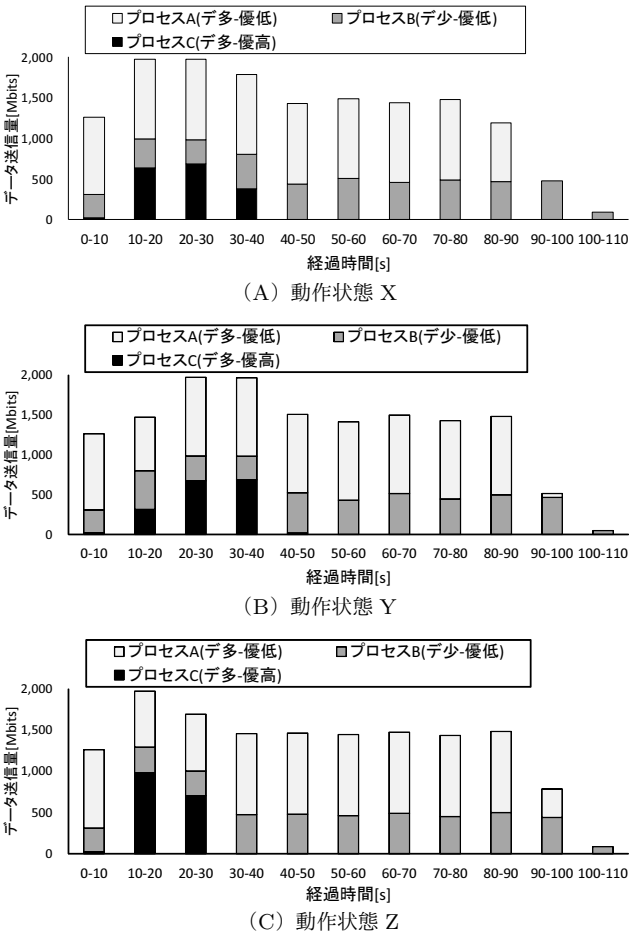
4.2 評価環境

評価環境を**図 2** に示す。送信側計算機上に 2 枚の 100 Mbps NIC を搭載し、各受信側計算機と 1 対 1 対応するデータ送信プロセスを 3 つ走行させる。また、評価に使用した計算機を**表 1** に示し、各データ送信プロセスのデータ送信量と優先度を**表 2** に示す。各データ送信プロセスは、送信データ量と優先度を変えている。

測定では、データ送信プロセス A を走行させ、4 秒後にデータ送信プロセス B、さらに 4 秒後にデータ送信プロセス C を走行させる。この測定を 100 回行い、各データ送信量を計測し算出する。また、CPU 使用量は、3 つのデータ送信プロセスが同時に走行している区間を対象とする。

4.3 考察

データ送信プロセスのデータ送信量を**図 3** に示す。100 回の測定により、3 つの動作状態を観測できた。具体的には，プロセス優先度が高いプロセス C のデータ送信に着目して分類する。この送信処理が 40 秒で終了した状態において、早い時間帯（10 秒から 20 秒の間）でプロセス C のデータ送信量が増加した状態を動作状態 X とし、遅いものを動作状態 Y とする。また、この送信処理が 30 秒で終了した状態を動作状態 Z とする。各動作状態のデータ送信プロセスと NIC の対応関係を**図 4** に示す。



（A）動作状態 X

（B）動作状態 Y

（C）動作状態 Z

図 3　データ送信プロセスのデータ送信量

表 2　データ送信プロセスのデータ送信量と優先度

データ送信プロセス	データ送信量	優先度
A（デ多-優低）	(i) 1KB のデータを 100 万回連続で送信	低（120）
B（デ少-優低）	(i) 1KB のデータを 1 万回連続で送信した後，1 秒間 sleep() (ii) (i) を 50 回繰り返し	低（120）
C（デ中-優高）	(i) 1KB のデータを 20 万回連続で送信	高（110）

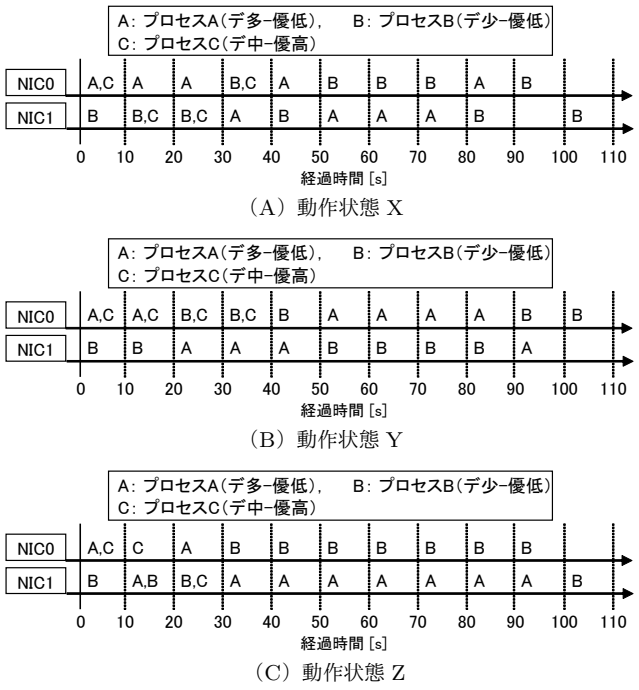


図 4　データ送信プロセスと NIC の対応関係

100 回の測定において、balance-tlb モードは、

動作状態Ｘ：９５回、Ｙ：５回、Ｚ：０回

であった。これに対し、提案法は、

動作状態Ｘ：０回、Ｙ：０回、Ｚ：１００回

であった。また、いずれの動作状態においても最初の 10 秒間は、

- NIC0 のデータ送信量：951Mbits
- NIC1 のデータ送信量：311Mbits

であり、総データ送信量は 1262 Mbits、NIC0 は過負荷状態であった。

データ送信量が多く、かつ優先度が高いプロセスはプロセスＣであり、このプロセスと NIC との対応付けが重要である。10 秒から 20 秒の間の動作状態について以下に説明する。

(1) 提案法は動作状態Ｚである。提案法は、最初の 10 秒における NIC0 の過負荷状態、およびその影響によるプロセスＣのデータ送信量の低下を考慮し、次の 10 秒での対応関係を設定している。具体的には、動作状態Ｚは、プロセスＣに対応付けた NIC0 に、他のプロセス（ＡとＢ）を対応づけていない。このため、

- プロセスＣのデータ送信量：983Mbits
- 総データ送信量：1974 Mbits

である。

(2) balance-tlb モードは、動作状態ＸまたはＹである。balance-tlb モードは、最初の 10 秒における NIC0 の過負荷状態、およびその影響によるプロセスＣのデータ送信量の低下を考慮しないため、動作状態Ｘや動作状態Ｙになっている。両者の状態は、プロセスＣと同じ NIC に対応付けられるプロセスが異なる（ＡまたはＢ）だけで、同じような対応関係である。このため、動作状態Ｘの場合、

- プロセスＣのデータ送信量：630Mbits
- 総データ送信量：1974Mbits

動作状態Ｙの場合、

- プロセスＣのデータ送信量：316Mbits
- 総データ送信量：1472Mbits

である。

したがって、提案法は balance-tlb モードに比べ以下の特徴がある。

(A) 優先度が高いプロセスＣのデータ送信量を大きく増加（1.6 倍あるいは 3.1 倍）できる。

(B) 総データ送信量を増加（同等あるいは 1.3 倍）できる。

また、優先度が高いプロセスＣの走行時間は、

- 動作状態Ｘ：27.4 秒
- 動作状態Ｙ：32.1 秒
- 動作状態Ｚ：19.0 秒

であった。したがって、提案法は balance-tlb モードに比べ、

(C) 優先度が高いプロセスの走行時間を約 2/3 に短縮できる。

といえる。

以上のことから、提案法は（要求２）を満足している。

CPU 使用量は、

- 動作状態Ｘ：１１．４６％
- 動作状態Ｙ：１０．１３％
- 動作状態Ｚ：１１．１８％

であった。３つの動作状態の CPU 使用量は同等であることから、提案法の CPU 使用量は balance-tlb モードと同等である。したがって、提案法は（要求１）を満足している。

5. 関連研究

文献 [4] は、マルチメディアデータの配送に広帯域幅を要求する場合について、各ＮＩＣに適切な比率でＩＰデータグラムを配分する手法を述べている。これにより、通信路ごとの帯域幅が異なることを吸収して、性能の劣化を防いでいる。また、文献 [5] は、複数の経路を利用する場合

について、各経路の遅延や遅延揺らぎを考慮してパケット制御を行う方式を述べている。文献 [6] は、無線環境において複数経路を通信するプロトコルを述べている。これらは、通信経路に関するものであり、通信スループットを向上させるものではない。

RI2N++[7] は、受信側計算機に到着したパケット数を基に、各 NIC に割り当てるパケットの割合を変更している。この手法は、提案法と比較して、データ送信プロセス 1 つに複数の NIC からソフト割り込みが発生するため、ソフト割り込みの処理に要する処理時間が長くなり、平均 CPU 使用率が高くなる。

複数の NIC を使用する際、割り込みを処理するコアを考慮することにより、低い平均 CPU 使用率で高いスループットを実現する手法として、MiAMI[8] がある。MiAMI は、同一のコア上でプロセスの走行、および I/O デバイスにより作成された割り込みの処理をすることにより、キャッシュヒット率を向上させ、低い平均 CPU 使用率で高いスループットを実現している。この手法は、提案法と比較して、NIC の過負荷時において、各プロセスの優先度を考慮した負荷分散を行えない。

6. おわりに

過去のデータ送信において、NIC の I/O 性能がボトルネック（過負荷）のためにデータ送信プロセスのデータ送信量が抑制されているか否かを確認し、この結果に応じてデータ送信プロセスと NIC の対応付けを変更する複数 NIC 間での負荷分散法を提案した。

既存の複数 NIC を仮想的に 1 つに見せる負荷分散法の問題へ対処するため、2 つの要求を述べた。1 つは、データ送信プロセスと NIC の対応付けの変更回数を抑制してソフトウェア割り込みを抑制し、CPU 使用量の増加を抑えることである。もう 1 つは、過去の状態において、NIC の I/O 性能ボトルネック情報（過負荷情報）、およびデータ送信プロセスのデータ送信量の抑制情報を考慮することである。

評価により、提案法は 2 つの要求を満足していることを示した。

残された課題として、「過去 T 秒間」について 10 秒以外の場合の評価など詳細な評価、および複数のプロセス優先度が存在する場合の評価や新たな対処方式の明確化がある。

謝辞

プログラムのバグ改修と測定に御協力頂いた岡本裕之君（岡山大学大学院自然科学研究科博士前期課程）に感謝します。

参考文献

[1] The FreeBSD Project: LAGG(4), available from ⟨http://www.freebsd.org/cgi/man.cgi?query=lagg(4)⟩ (accessed 2014-08-06).
[2] Davis, T.: Linux Channel Bonding, available from ⟨http://sourceforge.net/projects/bonding/⟩ (accessed 2014-08-06).
[3] 吉田泰三, 山内利宏, 谷口秀夫: 複数 NIC を連携制御する既存手法の評価, 情報処理学会研究報告, Vol. 2014-DPS-161, No. 11, pp. 1–8 (2014).
[4] 加藤剛史, 桧垣博章: 複数種 NIC による広帯域通信のための TCP 再送抑制手法, 情報処理学会研究報告, Vol. 2005, No. 33(2004-CSEC-028), pp. 111–116 (2005).
[5] 川島佑毅, 峰野博史, 石原 進, 水野忠則: 利用経路を動的に制御する複数経路集約通信方式の評価, 情報処理学会論文誌, Vol. 48, No. 2, pp. 880–891 (2007).
[6] 坪井祐樹, 相田 仁: 無線環境における複数経路通信プロトコルの検討, 情報処理学会研究報告, Vol. 2011-DPS-148, No. 11, pp. 1–7 (2011).
[7] 米元大我, 塙 敏博, 三浦信一, 朴 泰祐, 佐藤三久: トラフィック量に適応する非対称マルチリンク Ethernet トランキング, 情報処理学会論文誌コンピューティングシステム (ACS), Vol. 3, No. 1, pp. 25–37 (2010).
[8] Jang, H. C. and Jin, H. W.: MiAMI: Multi-core aware processor affinity for TCP/IP over multiple network interfaces, *Proc. 17th IEEE Symposium on High Performance Interconnects (HOTI 2009)*, pp. 73–82 (2009).