Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

ネットワーク転送時におけるノード消費電力削減

児玉祐悦[†1]　高野了成[†1]　岡崎史裕[†1]
工藤知宏[†1]　伊藤智[†1]

データセンタの省エネルギー化を推進するために、IT 機器による生産性を加味した電力利用効率の指標が求められている。そのような指標を策定するために、処理内容による消費電力のモデル化が重要となる。その一歩として、ネットワーク転送時のノードの消費電力のモデル化を試みた。その際、ペーシングによる帯域制御を行ったところ、転送バンド幅を減少させても消費電力が増加する場合が観測された。これは割り込み削減機構に因るものであり、この割り込み遅延時間を制御することにより、消費電力を削減することができた。ネットワーク転送時の消費電力のモデル化には、転送バンド幅だけでなく、割り込み回数をパラメータとすることが有効であった。

Power reduction of nodes with network traffic

Yuetsu Kodama ,[†1] Ryousei Takano ,[†1]
Fumihiro Okazaki ,[†1] Tomohiro Kudoh [†1]
and Satoshi Itoh[†1]

To improve the energy efficiency of data centers, the new metrics for data center efficiency are required to include productivity that is a useful work produced in a data center. To propose a new metric, we will create a model of power consumption for productivity. As the first step, we measured the power consumption of nodes when they communicate using network. In this measurement, we observed that the power consumption increased when the effective bandwidth was decreased with rate controlling by pacing. This phenomenon was caused by interrupt coalescing, and by controlling the delay time of interrupt the power consumption can be decreased. We also found that the number of interrupts is a good parameter to estimate the power consumption of nodes with communication.

1. はじめに

持続的社会の実現に向けて省エネルギーへの取り組みが広く行われており、情報機器が集約されているデータセンタにおいてもその省エネルギー化が強く求められている。データセンターの電力削減を進めていく上で、電力利用効率の指標を定めることは重要である。しかし、現在広く用いられている指標である PUE[1)] は

$PUE =$ データセンタ全体の消費電力/IT 機器による消費電力

で定められており、データセンタでどのような処理が行われるかは評価されない。そのため、同じ処理を行う場合、IT 機器の消費電力のみが削減されると、かえって PUE が悪化してしまう。そこで、データセンタで行った生産性 (プロダクティビティ) を考慮した指標が求められている。

そのような指標を考えるうえで、どのような処理を行ったらどれだけの消費電力となるかをモデル化することが重要である。我々は、その一歩としてネットワーク処理に要する消費電力をモデル化することを目的として、各種条件でのノードの消費電力を測定した。その際に、トラフィックをペーシングした場合に、データ転送のバンド幅を低下させても、消費電力が増加する場合があることが分かった。本稿では、その原因を明らかにすると共に、ペーシングを用いた場合に消費電力を削減する手法について示し、その効果の評価を行う。また、このようなデータ転送時の消費電力のモデル化を行う上で適切なパラメータについて検討を行う。

2. ネットワークデータ転送時のノードの消費電力のモデル化

2.1 評価環境

ノード間でデータ転送を行ったときの消費電力のモデル化を行うために、図 1 の環境で消費電力の測定を行った。ノードは DELL 社のブレードサーバ M1000e に搭載されたブレード M600 で、クアッドコアの Xeon E5420(2.5GHz) を 2 ソケット搭載している。1 つの筐体に 16 台のブレードを搭載可能であるが、使用した環境では 8 台のブレードが搭載されており、このうちの 2 台を用いた。OS は CentOS 5.2(kernel 2.6.18-92) を用いている。

各ブレードはパススルーモジュールで Force10 社のネットワークスイッチ C300 に接続さ

†1 (独) 産業技術総合研究所
National Institute of Advanced Industrial Science and Technology (AIST)

© 2009 Information Processing Society of Japan

Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

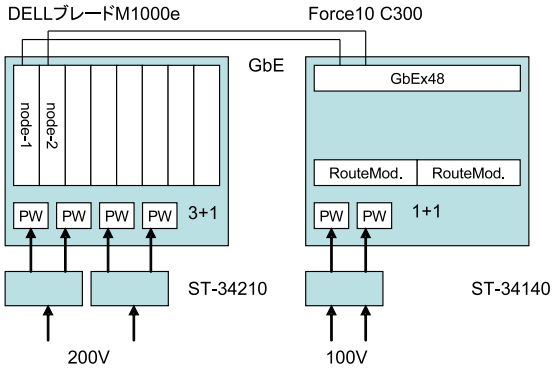

図 1 計算機環境

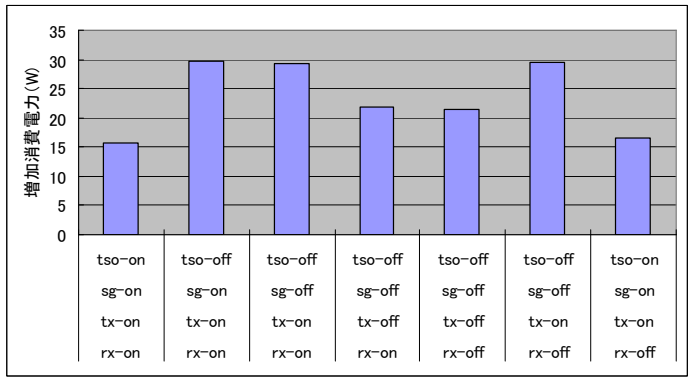

図 2 NIC のオフローディング機構による消費電力の比較

れている。C300 は 7 枚のインタフェースボードが搭載可能であるが、使用した環境では 48 ポートの 1000Base-T のインタフェースボード 1 枚と、冗長構成の 2 枚のルートモジュールが搭載されている。また、詳細なバンド幅測定やネットワークでの帯域制御を行うために、ノード 1 とスイッチの間を GtrcNET-1[2] を介して接続している。

ブレードサーバの電源モジュールは 200V 入力で最大 3+3 の冗長構成が取れるが、利用した環境では 4 台を 3+1 構成として用いている。電力測定には 200V 1 系統の測定が行えるシナジェテック社の ST-34210 を用いた。ブレードサーバへの 2 入力を 1 組として ST-34210 を 2 台用いて測定し、その結果を合算してブレードの電力とした。また、ネットワークスイッチの電源モジュールは 100V 入力で 1+1 の冗長構成となっている。こちらの電力測定には 100V 4 系統の測定が行える同じくシナジェテック社の ST-34140 を用い、各入力を測定して、その結果を合算してスイッチの電力とした。ST-34210/140 は 1 秒毎の電力の測定結果を HTTP 経由で取得することが可能である。

スイッチの消費電力は、ギガビットイーサネット 48 ポートがリンクアップした状態で 407W、全てのポートに 64 バイトの UDP パケットを双方向に最大転送レート (ワイヤレート) で流した状態の消費電力が 411W で、ポートあたり 0.1W 以下の上昇しか観測されなかった。そのため、以下ではノードの消費電力のみを測定した。

8 ノードがアイドル状態の時のブレードサーバの消費電力は 1120W であり、以下の測定はこれから増加した消費電力のみを示した。1 ノードを shutdown したときの電力低下は約 100W であった。ただし、アイドル状態であっても消費電力は増減し、1 秒毎の消費電力を 1000 秒間測定したときの標準偏差は 2.8W ほどであり、測定結果を考察する際には注意が必要である。

2.2 帯域制御を行わない場合の消費電力

データ転送には、ネットワークの転送評価に使われる iperf を用いた。2 ノード間で iperf を 1000 秒間実行したときの転送速度は 942Mbps で、アイドル時からの消費電力の増加の平均は 15.6W であった。この転送速度は iperf の出力であり、アプリケーション層における転送速度である。ギガビットイーサネットレベルでは、ワイヤレート 987Mbps が安定して出ていることを、GtrcNET-1 による測定で確認している。ただし、ブレードでは各ノード毎の電力測定が行えず、送信ノード/受信ノードでの電力増加は切り分けができていない。

各ノードに搭載されているネットワークインタフェースチップ (Broadcom BCM5708) では各種オフローディング機構がサポートされている。上で述べた値は、以下の機能全てを有効にしたときの値である。これらの機能はデフォルトで全て有効であった。これらを変更した場合の消費電力の変化は図 2 の通りである。tso は TCP セグメント、sg はスキャッタギャザ、tx は送信チェックサム、rx は受信チェックサムの各オフローディングが有効かどうかを示す。ただし、tso と sg は tx が off の時には on にできない。

tso を on にしたときの消費電力削減効果は大きく、tso のみ off にしたときと比べて 14W ほど削減している。tso をオフした場合は tx も off にしたほうが電力が削減されているが、詳細は不明である。tx あるいは rx のみを on にした場合の消費電力に対する影響は小さい。

各ノードのクロック周波数変更による電力制御 (CPUfreq) はいくつか選択可能である。上

© 2009 Information Processing Society of Japan

Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

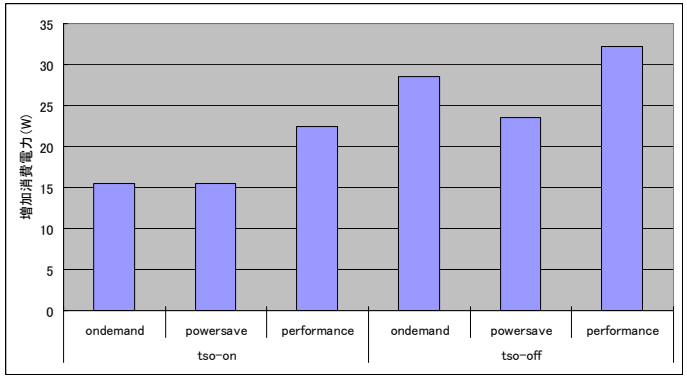

図 3 電力制御方式による消費電力の比較

で述べた値は、デフォルトの ondemand 制御の場合で、低負荷時のクロック周波数は 2.0GHz, 高負荷時の周波数は 2.5GHz であり、適宜 CPU 負荷に応じて切り替わる。その他の電力制御として、powersave および performance があり、これらを用いたときの消費電力の増加は図 3 の通りである。powersave とは常に最小のクロック周波数 (ここでは 2.0GHz) を用いる制御であり、performance とは逆に常に最高のクロック周波数 (2.5GHz) を用いる制御である。また、NIC の TCP セグメントオフローディングをオフにした場合の消費電力増加も参考として示した。

TCP セグメントオフローディングがオンの場合は、ondemand と powersave はほぼ同じで、CPU 負荷が低くほぼ最小クロック周波数で動作しているものと思われる。performance では 7W ほどの増加がみられる。一方、TCP セグメントオフローディングをオフにすると、ondemand は powersave に比べて 5W ほど増加し、一部高クロックで動作しているものと思われる。また、performance はさらに 3.6W ほど増加する。以下では、ondemand の動作を基本とする。

2.3 帯域制御を行った場合の消費電力

データセンタ内である程度のサイズのデータ転送を行う場合、ネットワークの輻輳が起きなければ、データ転送に起因する消費電力量は上記の電力と転送時間の積でおおよそ求まると考えられる。一方、データセンタの外とのデータ転送の場合には、ネットワークがボトルネックとなっていてネットワークインタフェースの性能を出しきることはできない。そこで、送信側で帯域制御を行い、送信帯域の変化により消費電力がどう変化するかについて評価を行った。

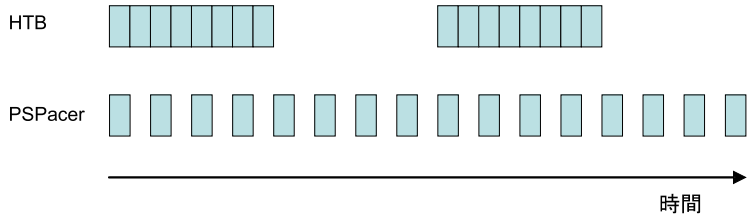

図 4 HTB と PSPacer のパケットの流れ

帯域制御としては、HTB と PSPacer[3] を用いた。HTB は Linux で標準にサポートされている帯域制御ツールで、トークンバケットベースの制御により帯域制御を行う。HTB で帯域制御したトラフィックを GtrcNET-1 で詳細に測定したところ、8 ミリ秒ごとに帯域制御を行っていることが観測された。例えば 500Mbps に帯域を制御する場合、最初に 4 ミリ秒連続的にトラフィックが流れて、その後 4 ミリ秒送信を停止することを繰り返す。これにより平均的な帯域を制御している。この制御間隔は Linux の HZ の設定により異る。また、最新のカーネルでは高精度タイマを用いてより細粒度の制御も可能になっている。

PSPacer は産総研で開発したパケット単位で帯域制御を行うソフトウェアであり、HTB と切り替えて利用することができる。PSPacer は、パケット間に適切なサイズのギャップパケットを挿入することにより、帯域制御を行っており、ソフトウェアのみでパケットレベルの精密な帯域制御が行える。この実装では、送信ノードはダミーパケットも含めると常にワイヤレートの送信を行っている。ギャップパケットは途中のスイッチで破棄され、受信ノードでは指定された間隔でパケットが到着する。

図 4 に 500Mbps に制御したときの HTB と PSPacer のパケットの流れを示した。HTB では連続したパケット流 (バーストトラフィック) とパケットが流れていない時間が交互に現れる。それに対し、PSPacer では、パケットとパケットの間隔が設定帯域に合わせて適切に制御されている。このように、パケット単位で設定帯域が制御され、バーストトラフィックが起きない帯域制御をペーシングという。

送受信ノード共に 100Mbps きざみで帯域制御を行った場合の消費電力の変化を図 5 に示す。また、比較として帯域制御を行わない場合の消費電力を再掲した。ただし、HTB および PSPacer を設定する場合は、TCP セグメントオフローディングを無効にする必要があるため、帯域制御を行わない場合も、TCP オフローディングを無効にした場合を示した。

© 2009 Information Processing Society of Japan

Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

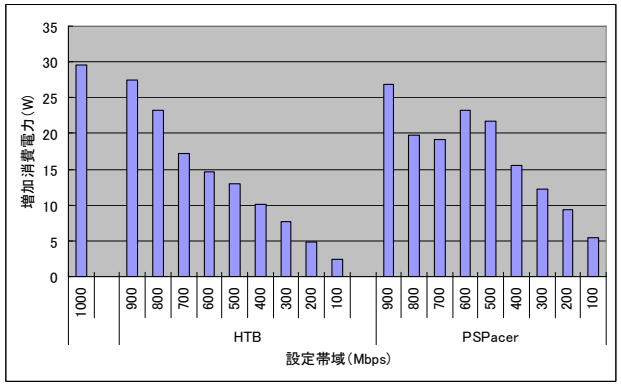

図 5 送信ノードによる帯域制御時の消費電力増加

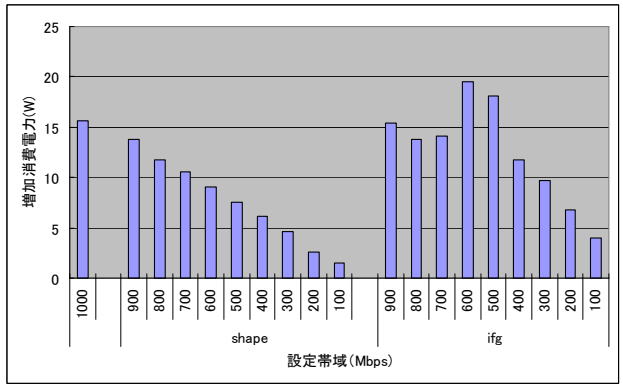

図 6 GtrcNET-1 による帯域制御時の増加消費電力

前節で示したとおり、これは有効にした場合に比べて約 2 倍の値である。PSPacer の 10 ギガビットイーサネットに向けた改良では TCP オフローディングを有効にすることも可能となっているが、現在リリース準備中であり、ここではダウンロード可能なバージョン 2.2.1 を用いている。

HTB による帯域制御では、制御帯域が小さくなると、消費電力もほぼ比例して減少しており、100Mbps の場合は制御なしのときのおよそ 1/10 程度である。ただし、単位転送量あたりの消費電力では、制御なしとほぼ同じである。

PSPacer を用いた場合も、制御帯域が小さくなると消費電力も減少している。しかし、消費電力は単調には減少しておらず、600Mbps にピークが生じている。また、HTB よりも消費電力が多く、100Mbps に制御した場合で 5.4W と HTB の約 2 倍となっている。

PSPacer の場合に消費電力が制御帯域に比例しなかったり、HTB に比べて増加している原因が、PSPacer の実装に因るものかどうかを詳しく調べるために、HTB および PSPacer と同等の帯域制御をネットワーク上で行い、各ノードでは制御なし、TCP セグメントオフロード有効で転送を行ったときの消費電力を測定した。ネットワーク上での帯域制御には GtrcNET-1 の shape および ifg 機能を用いた。shape はトークンバケットによる帯域制御で制御間隔も変更できるが、今回は HTB に合わせて 8 ミリ秒とした。ifg は PSPacer と同様にパケット間隔を制御する方式で、ギャップパケットではなく直接パケット間隔を制御する。これらを用いて、双方向の帯域制御を 100Mbps きざみで行った場合の消費電力の変化を図 6 に示す。

shape では、制御帯域にほぼ比例した消費電力となり、100Mbps では制御なしのほぼ 1/10 になっている。また、単位転送量あたりの消費電力は、制御なしとほぼ同じである。

一方、ifg では、900Mbps の時には制御なしとほぼ同じ消費電力であり、600Mbps では制御なしを越える高いピークがみられる。また、100Mbps の時も制御なしの 1/5 程度となり、単位転送量あたりの消費電力では、制御なしと比べて約 2 倍となっている。この傾向は、PSPacer を用いた場合とほぼ同じである。そのため、これらの現象は、PSPacer で連続的に NIC からパケットを送出しているという実装上の影響ではなく、パケットペーシングを行っていることの影響であると考えられる。

この 600Mbps で消費電力が増加する原因を考察したところ、パケット受信処理における割り込み削減機構 (Interrupt Coalescing) の影響であると推定された。基本的に、NIC がパケットを受信すると、割り込みにより OS に伝える。しかし、ギガビットイーサネットでは、パケットが到着するたびに割り込みを起こすと、I/O 負荷が高くなり過ぎるため、パケット受信が完了しても一定時間待ち、その時間内に受信を完了したパケットを一回の割り込みで OS に伝える手法が提案されている。これを割り込み削減機構といい、多くの NIC で実装されている。確認したところ、上記の評価で用いたドライバ tg3 では、この時間が 18 マイクロ秒に設定されていた。

ifg および shape による帯域制御の場合の割り込み回数を測定したところ、表 1 の通りであった。shape では、帯域が減少するにつれて割り込み頻度は減少し、どの帯域でも 1 割り込みあたりおよそ 6 パケットを処理していた。6 パケットは 1 度の割り込みで処理する最

© 2009 Information Processing Society of Japan

Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

表 1 受信ノードの割り込み頻度

制御帯域 (Mbps)	ifg		shape	
	割り込み/秒	パケット/割り込み	割り込み/秒	パケット/割り込み
900	12588	5.89	12564	5.90
800	15985	4.12	11187	5.89
700	19768	2.92	9823	5.87
600	49122	1.01	8438	5.86
500	40929	1.01	7060	5.83
400	32847	1.00	5680	5.80
300	36219	0.68	4305	5.74
200	24617	0.67	2920	5.64
100	12374	0.67	1530	5.38

大パケット数として、別途ドライバで設定されている。帯域が減少するにつれて割り込みあたりの平均パケット数が若干減っていくのは、帯域が減少するにつれてバースト長が短くなり、バーストの最後の 6 パケット未満の処理の割合が増えるためと思われる。

一方、ifg の場合は、帯域が減少しても割り込み頻度は逆に増加し、600Mbps の場合には 1 パケット毎に割り込みが生じていた。このときパケットの先頭の間隔は 20 マイクロ秒と設定の 18 マイクロ秒を越えている。このような割り込み回数の増加による I/O 負荷の増加が消費電力増加の原因であると思われる。なお、300Mbps 以下で割り込みあたりの平均パケット数が 0.67 と 1 より小さくなっているのは、ACK パケットの送信が受信割り込み時に行われなくなり、別途送信用の割り込みが生じているためと思われる。ACK パケットはデータパケット 2 パケットにつき 1 パケット送信されており、データ 2 パケットにつき受信割り込み 2 回、送信割り込み 1 回の計 3 回の割り込みが発生している。このため、割り込みあたりの平均パケット数は 0.67 となる。また、900Mbps から 700Mbps までは徐々に割り込みあたりの平均パケット数が減少している。これは受信バンド幅が高い場合には、割り込み遅延制御ではなく、一定時間間隔で割り込みを発生させているためである。

割り込み削減機構は CPU 負荷と遅延のトレードオフにあることは知られているが、消費電力への影響が思いのほか大きいことが分かった。

この割り込み遅延時間は ethtool コマンドで設定可能である。設定パラメータはドライバにより異るが、tg3 では rx-usecs および rx-usecs-irq による変更が可能であった。そこで受信割り込み待ち時間を変更して、消費電力の評価を行った結果が図 7 である。各折れ線が割り込み遅延時間に対応している。各点は 400 秒実行したときの平均の消費電力である。参考として帯域制御を行わない場合の点を no-rc として示している。

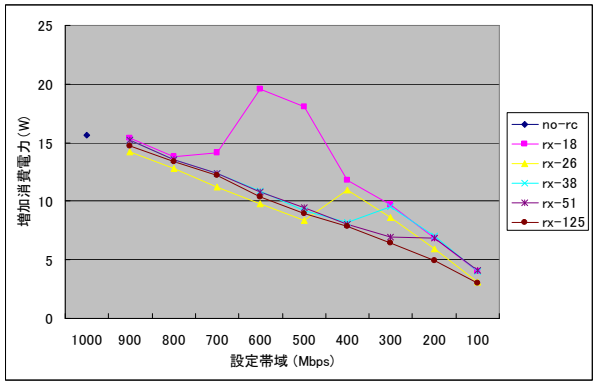

図 7 受信割り込み待ち時間変更による増加消費電力

割り込み遅延時間を大きくするにしたがって、途中のピークの位置がバンド幅の小さい方に移動すると共に、その際の消費電力の増加も小さくなっている。もっとも大きな効果は、600Mbps に制御した場合に割り込み遅延を 18 マイクロ秒から 26 マイクロ秒に変更した場合で、19.6W から 9.8W へ約 10W の電力削減を実現した。これは 2 ノードで 220W の通信時消費電力を 210W に削減するもので、約 5%の削減に相当する。割り込み遅延時間を 125 マイクロ秒にすると、100Mbps までは shape の場合とほぼ同じような消費電力になっている。このように、適切な遅延時間の設定によりペーシングを行ったときでも消費電力を軽減できることが分かった。

一方、割り込み遅延時間を大きくすると、それだけ受信完了の時間が遅れる。その影響を確認するため、2 つのノードでデータを交互に送り合うプログラムを実行して、その往復遅延時間を測定した。その結果が図 8 である。図の横軸はデータのサイズ、縦軸は 1 回の往復遅延である。各折れ線が各割り込み遅延時間に対応している。各 10 万回データを往復してその平均を示している。

割り込み遅延時間は送信/受信双方のノードで同じ値を設定しており、割り込み遅延時間とデフォルトの遅延時間 (18 マイクロ秒) の差の 2 倍程度、往復遅延が増加している。

この実験環境では、2 つのノードは 1 つのスイッチを介して接続されており、最小往復遅延は 100 マイクロ秒程度と小さい値になっており、割り込み遅延時間増加の影響が相対的に大きく見えている。一方、データセンタの外との接続の際は、最小遅延もミリ秒オーダとなり、100 マイクロ秒程度の遅延の増加は無視できる場合もあると思われる。また、割り

© 2009 Information Processing Society of Japan

Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

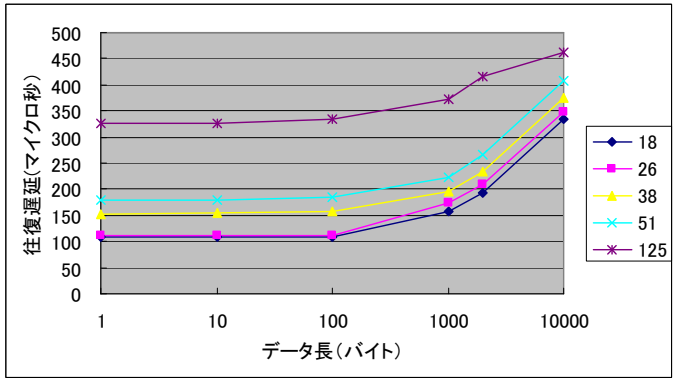

図 8 受信割り込み待ち時間変更による往復遅延への影響

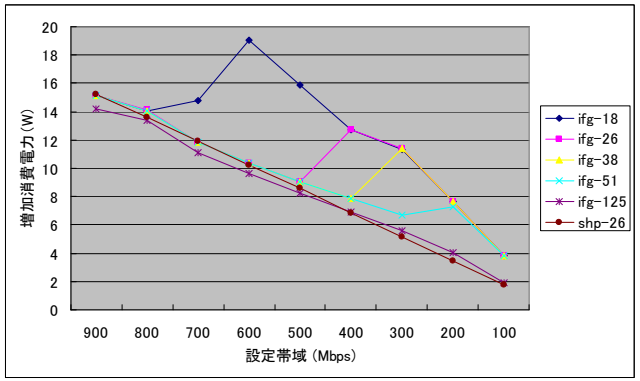

図 9 データ転送バンド幅と割り込み回数を用いた消費電力のモデル化

込み遅延時間を 26 マイクロ秒に変更することは、遅延の増加が 8 マイクロ秒と小さいわりには、消費電力に対する効果は大きい。このように、初期設定値が適切かどうかを確認することは有効である。いずれにせよ、遅延の増加と消費電力の低下はトレードオフになり、用途に応じて適切に設定することが望ましい。

また、本評価ではトークンバケット方式の HTB や shape では連続的にパケットが到着するために、ペーシング方式の PSPacer や ifg よりも消費電力が低くなっていた。しかし、データセンタの外との通信では、パケットを連続して送信しても、共有ネットワークを通り、そこで他のトラフィックのパケットと混在することになる。そのため、受信ノードで連続的にパケットが受信されずに、パケット間隔が生じる可能性が高い。その場合、トークンバケット方式でもペーシング方式と同様の割り込み頻度の増大という現象が起きると考えられる。

2.4 消費電力のモデル化

帯域制御を行ったときの消費電力を見積もることを考える。単に、データ転送バンド幅だけを考える方法では、図 6 の ifg のような消費電力を正しく見積もることはできない。そこで、割り込み回数をパラメータとして追加することを検討した。割り込み遅延 18 マイクロ秒の時の消費電力に対して、バンド幅と 1 秒あたりの割り込み回数から回帰分析を行い、以下の係数を得た。

消費電力 $(W) = 1.25 \times 10^{-2} \times$ データ転送速度 $(Mbps) + 1.56 \times 10^{-4} \times$ 割り込み頻度 $(intr/s)$

これをもとに、ifg の他の割り込み遅延時間、および shape による帯域制御の場合の消費電力を見積もった結果が図 9 である。なお、shape については割り込み遅延の違いによる差が小さいため 26 マイクロ秒の場合のみ示した。図 6 と同様のグラフを得ることが出来ることを確認した。誤差の平均、最大値, 標準偏差はそれぞれ、0.5W, 2.7W, 0.5 であった。

プロセッサ利用率などはプロセッサのモデル化で行い、最終的にはそれらを組み合わせることを想定しており、ここでは利用していない。動作周波数変更などの電力制御のモードや、NIC のオフローディング機能の有無などについても、本モデル化では対応できていない。これらも、プロセッサ負荷へ影響を与えると考えられるので、プロセッサのモデル化との統合により対応できるのではないかと考えられる。

3. ま と め

ネットワーク転送時のノードの消費電力のモデル化のために、帯域制御を用いて転送バンド幅を変化させ、消費電力を測定した。帯域制御にパケット単位の制御を行うペーシングを用いた場合に、転送バンド幅を減少させても消費電力が増加する場合が観測された。これは割り込み削減機構に因るものであった。割り込み削減機構は CPU 負荷と遅延のトレードオフにあることは知られているが、消費電力への影響が大きいことが分かった。この割り込み遅延時間を制御することによる消費電力削減効果を評価し、最大で 10W の消費電力削減を確認した。これは 2 ノードの通信時消費電力の約 5%に相当する。また、ネットワーク転送時の消費電力のモデル化には、転送バンド幅だけでなく、割り込み回数をパラメータとする

© 2009 Information Processing Society of Japan

Vol.2009-ARC-186 No.5
Vol.2009-HPC-123 No.5
2009/11/30

ことが有効であることを確認した。今後は、プロセッサやストレージの消費電力のモデル化と組み合わせることにより、データセンタ全体の消費電力のモデル化を行っていきたい。

謝辞 本研究は、独立行政法人新エネルギー・産業技術総合開発機構（ＮＥＤＯ）の委託業務「グリーンネットワーク・システム技術研究開発プロジェクト（グリーンＩＴプロジェクト）」の成果を一部活用しています。

参 考 文 献

1) Christian Belady, ”GREEN GRID DATA CENTER POWER EFFICIENCY METRICS: PUE AND DCIE”, white paper #6 of the Green Grid, 2008.
2) 児玉, 工藤, 佐藤, 関口, “ハードウェアネットワークエミュレータ GNET-1 におけるモジュール設計”, 第一回リコンフィギャラブルシステム研究会論文集, pp.271–276, 2003.
3) 高野, 工藤, 児玉, 松田, 石川, 岡崎, “ ギャップパケットを用いたソフトウェアによる精密ペーシング方式 ”, 情報処理学会論文誌, Vol.47, No.SIG 7 (ACS 14), pp.194–206 (2006).

© 2009 Information Processing Society of Japan