2007－HPC－111 (38)
2007／8／3

tagged-VLANを用いたPCクラスタ向けマルチパスネットワークにおける動的ルーティング

三浦信一[†] 岡本高幸[†]
朴泰祐[†] 塙敏博[††]

VFREC-Netは VLAN 技術を用いることで，安価な Layer-2 の Ethernet スイッチのみで高性能なマルチパスネットワークを構成できるクラスタ向けネットワークである．しかし既存の VFREC-Net では，それらのマルチパスを制御するルーティングテーブルが静的に決定されていたため，アプリケーションの通信パターンによっては用意した複数の経路が有効利用されないという問題があった．この問題を解決するために，アプリケーションレベルでこのルーティングテーブルを動的に変更可能なフレームワークを開発した．このフレームワークを用いることで，動的通信負荷分散アルゴリズムを適用可能となり，結果として VFREC-Net がより理想的な形で運用可能になる．これに加えて本論文では MPI プログラム上から直接ルーティングテーブルを変更可能にするためのインタフェースを開発した．ネットワーク経路の負荷分散が問題になる NPB Kernel CG において本システムを適用し，最適なルーティングテーブルになるようにアプリケーション中から動的に設定することで，ルーティングテーブルが静的な場合と比較してより高い性能を示すことが確認された．

A Dynamic Route Control System for PC Clusters with Multi-path Network Using tagged-VLAN Technology

SHIN'ICHI MIURA ,[†] TAKAYUKI OKAMOTO ,[†] TAISUKE BOKU [†]
and TOSHIHIRO HANAWA[††]

VFREC-Net is a network construction technology which allows to configure a multi-path network routing on inexpensive Layer-2 Ethernet switches based on tagged-VLAN technology. Current VFREC-Net system implies a problem on traffic balancing when the communication pattern of the application does not fit the network topology, due to its static routing scheme. We have developed a framework to solve this problem by allowing dynamic routing table rewriting from the application level. When an appropriate algorithm to optimize the VLAN-id allocation according to the application's behavior, it enables to balance the traffic on Fat-Tree topology with user-level controlling of VLAN. We also provide an API library for MPI programming to use above framework, and confirmed its effectiveness through the communication optimization on NPB Kernel-CG benchmark.

1. はじめに

一般の PC クラスタの多くは，ノード間を接続するネットワークとして Ethernet を採用している．特に Gigabit Ethernet（以後，GbE）はそのコストパフォーマンスの高さから多くのクラスタ環境で使用されている．一般的にコストパフォーマンスの良い Layer-2 GbE スイッチは，24 ポート程度の比較的小規模なものになるため，クラスタの規模がこれを上回る場合，複数台のスイッチを tree 構造等で相互に結合して用いる．クラスタの性能をノード数に合わせて向上させるためには，このスイッチ間のバンド幅もあわせて増強する必要があるが，Ethernet ではその性質上ネットワーク上にループ構造を作ることができず，スイッチ間はただ一つのパスで結ばなければならない．例外として LACP[1] 等の trunk 技術による複数パスの利用が考えられるが，トポロジの制限（2 台のスイッチ間での平行結線のみのサポート）から大規模化には対応することができない．そのため，大規模な HPC クラスタで Ethernet を用いる場合にはスイッチ間のバンド幅が問題となる．この問題を解決する一つの方法として VLAN ルーティング法[2] が提案されており，これを用いることで，いままでの Ethernet を利用したクラスタネットワークに存在したトポロジの制限を無くし，より柔軟なネットワーク構成が可能になる．我々はこの

† 筑波大学大学院 システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba
†† 筑波大学 計算科学研究センター
Center for Computational Sciences, University of Tsukuba

VLAN ルーティング法を発展・拡張させた，VFREC-Net を開発している[3]．VFREC-Net は VLAN ルーティング法で得られる高いネットワークバンド幅をユーザに提供し，典型的な HPC ベンチマークにおいて単純 tree 構造のネットワークに比べ高い実効性能を得ることができる．しかしルーティングテーブルを一意に固定しているため，一部のアプリケーションで十分な性能が得られないことが明らかになっている．そこで本論文では，ルーティングテーブルの設定方法に柔軟性をもたせ，動的に変更可能にすることでこの問題を解決する方法を提案・実装する．

2. VFREC-Net

2.1 VFREC-Net の概要

VLAN ルーティング法[2] は，tagged-VLAN（IEEE 802.1q)[4] を用い，各ノードから送信するパケットの VLAN ID（以後，VID）を制御することで，Ethernet を用いた場合でもさまざまなトポロジを構築可能にする．物理的にループのあるネットワーク構成を，リンク毎に異なる VLAN を割り当て，論理的にループのない複数のネットワークに展開分割し，これらのネットワークをノード側から明示的に使い分けることで，Ethernet のスイッチ間バンド幅ボトルネックを解決する．我々の開発している VFREC-Net（VLAN-based Flexible, Reliable and Expandable Commodity Network)[3] は，この VLAN ルーティング法を改良し，システムレベル（デバイスドライバ）で実装したうえで，より PC クラスタ向けに利用しやすくしたものである．文献[2] で提案されている VLAN ルーティング法が，通信で使用する VID 毎に複数の仮想デバイスとそれらに割り当てられた複数の IP アドレスを用いるのに対して，VFREC-Net は独自のドライバ（VFN ドライバ）を利用することで，ただ一つの仮想デバイスと，それに割り当てられたただ一つの IP アドレスのみで実現されている．そのため，通常のソケット API を用いたほぼすべてのプログラムが一切の変更無しで使用可能であり，OpenMPI[5] などの並列プログラミング環境に対してもそのまま適用可能である．VFREC-Net では通信経路（VID）の決定に送信先の MAC アドレスを用いる．どの MAC アドレスに対してどの VID を用いるかというルーティングテーブルはドライバの初期化時に与えられる．VFREC-Net のデフォルトのルーティングテーブルの設定では，各ノードで標準に用いる VID を割り当てた上で，それらのノードに優先度を設定する．そして，それらの通信のペアで優先度の高いほうの VID を用いる．これは，Ethernet スイッチの学習アルゴリズムにより，一組のノードペアの通信には同一の VID を使用しないといけないためである．

VFREC-Net では，VLAN ルーティング法で提案されているすべてのネットワークトポロジを構築できる．

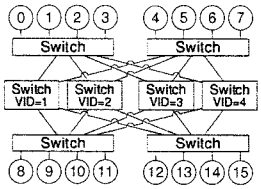
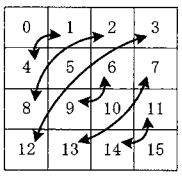


図 1　VLAN Based Fat Tree 構成のネットワーク



組合せ	最適化前	最適化後
1-4	2	2
2-8	3	3
3-12	4	4
6-9	3	4
7-13	4	3
11-14	4	2

図 2　NPB Kernel CG における問題となる通信の VID 割当

特にこの手法を用いることで可能になるマルチパスを用いた Fat Tree 形のトポロジである VLAN Based Fat Tree[2]（以後，VBFT 図 1）は，木構造の上位階層のバンド幅を増強した上で，下位層のトラフィックを均等に分散可能とするため，多くの HPC アプリケーション及びランダムトラフィックに向いていると考えられる．よって VFREC-Net では VBFT を基本的なネットワークトポロジとして実装および評価している．

2.2 VFREC-Net におけるルーティング上の問題点

NAS Parallel Benchmarks（以後，NPB）の一部のベンチマークにおいて，VFREC-Net がデフォルトとして与えるルーティング設定では通信パケットが特定の経路に偏る問題があった．以下では NPB Kernel CG を取り上げ，実際に VFREC-Net の開発中に解析した VLAN によるルーティングとトラフィックの偏り問題の実例について述べる[3]．

図 1 に示すようなノード数 16，上位層のスイッチを 4 台持つ VBFT 構成上で NPB Kernel CG における通信を考える．図中の数字は各ノードのランク番号を示している．VBFT を用い，スイッチ間の経路数を 4 とすることでルートを制御する VID の数は 4 となる．各ノードに 1 ～ 4 までの VID をサイクリックにノード番号の若い順に割り当て，ノード番号が若いノードほど優先度が高くなるように設定する．Kernel CG における行列のランク番号へのマッピング，問題となる通信パターンとそのルーティングテーブルを図 2 の「最適化前」に，実際の通信経路を図 3（a）にそれぞれ示す．

問題は 1 台のスイッチ内で閉じる通信ではなく，複数のスイッチを経由するもので生じている．図 3（a）中の丸で囲んだ部分が示すように，ノード 2–8 間と 6–9 間の通信が，また，ノード 3–12 間・7–13 間および 11–14 間の通信が，スイッチ間経路で衝突している．これらは，VBFT によって用意された複数の経路が有効に活用されていないことを示し，本来得られるべきネットワークバンド幅増大の効果が性能向上に結びつ

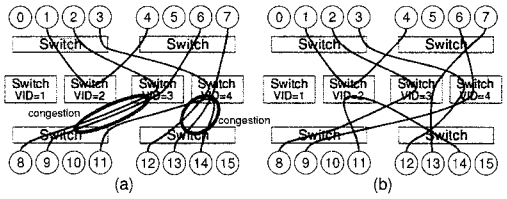


図 3 通信経路で見た衝突例（a）と衝突回避例（b）

かない．実際のベンチマーク結果においても，Kernel CG では VBFT における速度向上が不十分であった．そこで，図 2 に合わせて示した「最適化後」のように，使用する経路で衝突しないよう，特別なルーティングテーブルを設定した上で実験を行ったところ，性能が大きく改善できることを確認した．このときのネットワークの経路の使用状況を図 3（**b**）に示す．この場合にはスイッチ間経路で衝突する通信がないことが分かる．ここで重要なのは，ルーティングテーブル（VID）の決定方法である．最適化前後で 4 つの VID のうち 3 つしか利用していない点に変わりはなく，VID の均等利用という意味ではどちらも等価であり負荷の分散がうまくいっているように見える．しかし実際の経路利用状況を見てみると，最適化後では通信経路の衝突が回避され，最適化前よりも並列処理性能が向上している．VBFT の構成で得られる大きなバイセクションバンド幅を効率良く使うためには，使用する VID を均等に割り振るだけでなく通信パターンを注意深く観察しスイッチ間の経路で通信が衝突しないように VID を設定する必要がある．

2.3 通信負荷の効率的分散

これまでの VFREC-Net で想定してきた VBFT は，アプリケーション上の通信トラフィックを，Fat-Tree 網の上位階層にフラットに分散させることを目的に実装されてきた．しかしながら，ある種のアプリケーションにおいてはその性質上，通信トラフィックに偏りが生じ，これが VBFT の構成とマッチングしないことにより，本来 VBFT が持つバイセクションバンド幅増強の効果が十分発揮されない可能性がある．この問題に対する解の一つは，VLAN ルーティングにおいて VBFT の構造は変えず，VID の割付け方法を変更することにより，物理ネットワーク上での通信負荷分散を改善するというものであり，その可能性は前節で述べた解析により実証された．そこで，アプリケーションに適した VID の割付けを行うことにより，ネットワーク上の経路制御を積極的に行うことを提案する．本手法では，スイッチ自体の設定変更は行わず，送受信ノード内での VID 選択方法の変更のみで事実上のルーティング変更が可能である．これは，ハードウェアとソフトウェアの両者に跨ったルーティングという，VLAN ルーティングならではの特徴を利用するもので，小さなオーバヘッドで大きな効果を生むことが期待できる．

3. 通信最適化手法の検討

前述のように，既存の静的なルーティングテーブル決定方法では，偏りを持った通信パターンに対応することが難しい．そこで，初期設定のルーティングテーブルを用いつつ状況に応じて適宜ルーティングテーブルを更新し，通信量が多いノード間の通信がなるべく同一のパスを通らないように設定する動的な通信経路選択を実現するシステムを用意する．これを用いて，アプリケーション毎，あるいはアプリケーション内での処理フェーズ毎に，各ノード対が使用する VID の割当を動的に変更することで問題に対応する．ただし，動的ルーティング変更手法には，「そもそもどのように最適な経路分布を発見するのか」という，アルゴリズム的な問題が存在する．最適通信経路分布問題は，VLAN ルーティング法に限らずネットワークでは一般的な問題であり，本論文はこれに対する回答を与えることは目的としない．その代わり，最適な VID の分布が見つかった場合に，これを簡単にアプリケーションに適用可能とするシステムのフレームワークを提供する．具体的には，これまで VFREC-Net システムにおいて初期化ファイルによって固定テーブルとして提供されていたルーティング情報を，アプリケーションレベルから動的に変更することを可能とするシステムを構築することが本論文の目的である．

3.1 ルーティングテーブル変更手法

通信パターンに応じてルーティングテーブルを変更する際，VFREC-Net システムがどのようにこれを決定するかについて，以下の 2 通りの方法があり得る．

ユーザからの指示によるコントロール

ユーザプログラム自身からの明示的な指示によってルーティングテーブルの変更を行う方法である．多くの場合で，ユーザは自身のプログラムの通信パターンを把握可能であり，それをネットワークトポロジと比較することで最良なルーティングテーブルを設定できる．またプログラム中で通信パターンが大きく変わる場合でもその時点で最適なルーティングテーブルに変更することもできる．一方で，ユーザはトポロジと VID の関係を把握する必要があり，またプログラムの可搬性は小さくなる．

自律的なシステムによるコントロール

ユーザによる特別な指示なしに，システムが自律的に状況に適したルーティングテーブルを導きだし，その結果を用いてルーティングテーブルを設定する方法である．ネットワークの全トラフィックを監視できる管理サーバを用意し，その管理サーバが特定の評価関数に基づいてアダプティブなルーティングテーブルを求め，すべてのノードがその指示に従ってルーティングテーブルを変更する．この管理サーバがすべてのノードから通信量を取得しそれらを合成することでどの経

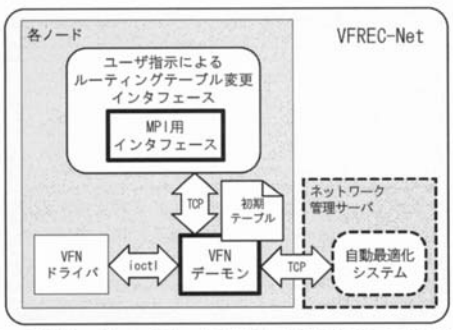


図 4 動的ルーティングテーブル変更システムを含む VFREC-Net システムのフレームワーク

路が最も混雑しているかを求める．これにより，どのようなアプリケーションを動作させた場合においても，それに応じたアダプティブなルーティングが設定され，ユーザはそれに対して，特別な意識をする必要がなくなる．一方で，ルーティングテーブルの変更タイミングなどによって，最適化されるまでの時間が必要であり，また適切な評価関数が設定されていなければ，さらに良いルーティングテーブルがあっても最適解に近づかない場合も考えられる．

4. 実　　装

本論文では，前述の 2 つの手法を実現するためのフレームワークを開発し，そのうえでまず前述のユーザ指示によるコントロール手法を実装・評価する．このフレームワークのイメージを**図 4** に示す．今回開発したものは，専用のデーモンプログラム **VFN デーモン**，およびユーザ指示によって動的にルーティングテーブルを変更する **MPI 用制御ライブラリ**である．

4.1 VFN デーモン

ルーティングテーブルを動的に変更する機能を実現するために，各ノードで専用のデーモンプログラム（VFN デーモン）を動作させる．VFN デーモンの最も大きな役割は各ノードで動作しているデバイスドライバ（VFN ドライバ）を管理することである．VFN ドライバは VFREC-Net の機能を提供するための核となるデバイスドライバであり，これは OS やユーザから見た場合，通常の Ethernet デバイスドライバとして振舞い，Linux カーネルモジュールの形で提供される．そのため，OS やユーザプログラムに一切の変更なしで TCP・UDP/IP などのソケットプログラムを利用できる．内部のルーティングテーブルの変更には *ioctl()* を通じた特別な操作が必要になるため，VFN デーモンがこれらの処理を代表して行う．動的なルーティングテーブルの変更を可能にし，またアダプティブなルーティングテーブルを決定するために VFN デーモンは以下に示す機能を提供している．

(1)　初期ルーティングテーブルの設定
(2)　ルーティングテーブル変更
(3)　各ノードへの送信量の収集
(4)　Network Interface の各種情報収集

今回はこれらの VFN デーモンの機能を実装するために，既存の VFN ドライバにログの収集機能，ドライバ内部のルーティングテーブルのセット／リセット機能を追加した．VFN デーモンは TCP を用いて各種ライブラリや外部のシステム管理サーバなどからの接続を受付け，それらのシステムが要求する処理（たとえばルーティングテーブルの変更など）を実行する．

4.2 MPI 環境における動的ルーティングテーブル制御ライブラリ

本論文ではプログラミング環境として MPI を用いることを前提とし，MPI プログラム上からルーティングテーブルを変更するインタフェースを提供する．ユーザが通信パターンを把握していることを前提とし，与えられたネットワークトポロジから特定の経路で通信が集中しないように，各ランク番号のペアごとに使用する通信経路すなわち VID を決定する．この環境をユーザに提供するために MPI 上からルーティングテーブルを制御する以下の 3 つの関数を用意した．

VFN_InitMPI()　ユーザはランク番号のペアから使用する通信経路（VID）を決定する．この関数は必要なルーティングテーブルの初期化処理を行う．MPI が動作していることを前提にしているため，そこで必要な通信は MPI を介して行う．なおこの段階のドライバ内のルーティングテーブルには，予め用意された標準のルーティングが設定される．

VFN_FinalizeMPI()　この関数が呼び出されると，所持しているランク番号と MAC アドレスのテーブルが即座に開放され，また，プログラム中で変更されたルーティングテーブルを初期状態にもどす．

VFN_SetRouteMPI(rank1, rank2, vid)　ユーザの指示によりローカルのルーティングテーブルに設定されている VID を *vid* に変更する．*rank1* および *rank2* は通信を行う 2 ノードのランク番号であり，これらの 2 ノード間では必ず同じ VID を用いるため，source, destination の区別はされない．初期化関数によって作成されたテーブルを元に，MPI のランク番号を実際の MAC アドレスに変換し，ローカルホストで動作しているデーモンにドライバ内のルーティングテーブル変更要求を行う．ルーティングテーブルの設定は，この関数が呼ばれたのちに即座に行われる．この関数は *VFN_InitMPI()* と *VFN_FianlizeMPI()* の間でのみ動作する．各ノードではローカルな VFN デーモンに対してソケット接続し，ルーティングテーブルの設定の場合はそのノードのランク番号に関するルーティング情報のみを設定する．ルーティングテーブルを変更するタイミングにより，ノードごとに使用する VID が一時的に異なり，それが原因でフラッディング送信がおこる可能性がある．フラッティングが継続して発生する

表 1　評価環境

CPU	Intel Xeon 3.0GHz EM64T 1-way
Memory	DDR2/400 1.0 Gigabytes
NIC	Intel PRO/1000 MT Dual Port Server Adapter （PCI-X 64bit/133MHz）1 ポートのみ使用
OS	Linux Kernel 2.6.20
MPI	LAM ver.7.1.3
Compiler	GCC ver.4.1
スイッチ	DELL PowerConnect 5224 Gigabit Ethernet 24 Port

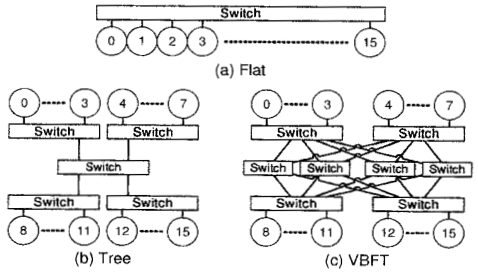


図 5　実験ネットワーク

と，ネットワークのスループットを低下させる原因になるが，この状況は一時的なもので，通信相手の VID 変更が終了すると共に解消する．フラッディングが行われる時間が短時間であるならばネットワークへの負荷も小さい．そこで変更要求前後に MPI 同期関数（*MPI_Barrier()* 等）で同期をとることで，このフラッディングによって生じるネットワークのスループットの低下を最小に抑えることもできる．

5. 評　　価

本評価の目的は，実装したシステムの動作を確認することである．加えて，事前にルーティングテーブルを最適になるように特別に固定した場合と比較し，本システムを用いて動的にルーティングテーブルを変更した場合でも同等の性能が得られることを確認する．評価環境として**表 1** に示すノード構成で計 16 ノードのクラスタを構築した．評価に用いたネットワークを**図 5** に示す．それらは，(a) すべてのノードを 1 台のスイッチに収容した Flat 構成，(b) 4 台のスイッチに 4 ノードずつ接続しスイッチ間を Tree 接続した Tree 構成，(c) 4 台のスイッチにそれぞれ 4 ノードずつ接続しスイッチ間に 4 つの経路を用意した VBFT 構成の計 3 種類のネットワークである．

5.1 評 価 内 容

次の 5 つの場合で評価する．

(1)　図 5（a）で示した Flat 環境
(2)　図 5（b）で示した Tree 環境
(3)　図 5（c）で示した VBFT で既存の固定ルーティングテーブルを用いた場合
(4)　図 5（c）で示した VBFT で予め最適化を行ったルーティングテーブルを用いた場合
(5)　図 5（c）で示した VBFT で最適化ルーティングテーブルを本手法によって与えた場合

```
call initialize_mpi    … cg.f 中の mpi 初期化関数
call VFN_InitMPI       … 初期化処理
call VFN_SetRouteMPI(1, 4, 2)    … テーブルの変更処理
…
call VFN_SetRouteMPI(11, 14, 2)
…
(cg.f の内容)
…
call VFN_FinalizeMPI    … 終了処理
call mpi_finalizer      … cg.f 中の mpi 終了関数
```

図 6　Kernel CG に追加した変更 (cg.f)

評価に用いたベンチマークは，問題に示した NPB Kernel CG（ver.3.2 CLASS=B NPROCS=16）である．VBFT 構成におけるルーティングテーブルの最適化では，すでに図 2 に示した最適化例を適用する．(4) では既存の固定ルーティング方法に特別な方法で該当するルーティングテーブルを設定し，(5) の場合では，アプリケーション実行中に *VFN_SetRouteMPI* を用いてルーティングテーブルを設定した．(4)，(5) の 2 つの方法は最終的に設定されているルーティングテーブルがほぼ等価であるため，両者の評価結果は等しくなることが予想できる．この 2 つの方法の評価結果によって新しく実装したシステムが正しく動作していることを確認する．

5.2 アプリケーションプログラムの変更

ルーティングテーブルを動的に変更するために，プログラムソース中で変更用関数を呼び出す必要がある．**図 6** に実際に Kernel CG のソースファイル（cg.f）中に加えた変更を示す．Kernel CG では，MPI の前処理を *initialize_mpi* で行っている．その直後に *VFN_InitMPI* を追加し VFREC-Net での動的ルーティングテーブル変更のための初期化処理を行う．その後にルーティングテーブルが実際に変更できるようになり，ここで図 2 に示した，最適化後のテーブルを記述していく．最後に，MPI の終了処理 *mpi_finalize* の直前に，動的ルーティングテーブルの終了処理 *VFN_FinalizeMPI* を記述した．

5.3 評 価 結 果

すべての通信を同一スイッチ内で閉じることが可能な Flat 構成のネットワークは，他の構成のネットワークと比較して，最も良い結果を得ることができると考えられる．そこで，本評価結果は（1）で得られた結果を基にした相対性能で示す．結果を**図 7** に示す．

予想どおり，すべての通信が同一スイッチ内で閉じる（1）の環境が，最も性能が良い．一方で多くの通信で衝突が発生する（2）は最も性能が悪い．VBFT 構成を用いた（3）の場合，(2) よりも性能が良い．固定されたルーティングテーブルで，且つルーティングテーブルを最適化していない場合でも，複数ある通信経路

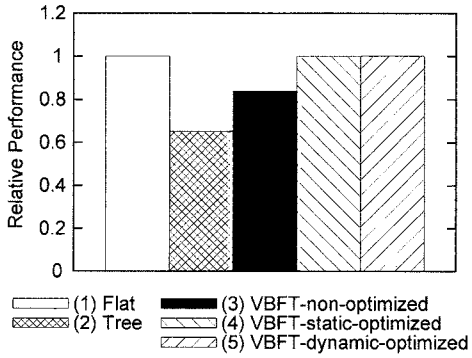


図 7　NPB Kernel CG における評価結果

に分散して送受信が行うため，幾分（2）と比較して結果は良くなる．さらに，固定ルーティングテーブルの VBFT 構成の場合でも，ルーティングテーブルを最適化した（4）はさらに結果は良くなり，その性能は（1）に近づく．このことから VBFT で得られる高いバイセクションバンド幅を，より効率的に使うためには，通信パターンを考慮に入れたルーティングテーブルの設定が重要であることが分かる．

最後に VBFT 構成を用いて初期化時に最適化を施した（4）とプログラム実行時にルーティングテーブルを設定した（5）を考察する．前述のように，この 2 つのルーティングテーブルはほぼ等価であり，実行前に最適化されたルーティングテーブルが設定されているか，実行時に必要な部分のみ最適なルーティングテーブルを設定するかの違いである．今回の Kernel CG では通信パターンがほぼ一定で明確であるため，最適なルーティングテーブルに固定可能であり，その評価結果と実装システムの評価結果がほぼ同じであるため，システムが有効に機能していることが分かる．

6. 関連研究

工藤らの提案した VLAN ルーティング法[2] は，VID ごとに論理的なネットワークを構成しそれらに独自の IP アドレス空間を与えることで，送信に用いる VID を制御しそれによって使用経路を選択している．大塚らの研究[6] ではルーティングを行うための VID のタグ付けをノードではなくスイッチで行う．そのため送受信のノードはスイッチ間のルーティングのために特別なドライバや設定は必要とせず，スイッチの機能のみを用いて VLAN ルーティング法を実現する．これら 2 つの実装においても本論文で述べているようなトラフィックの偏りに対して，経路変更を行うシステムが必要になる．しかし，工藤らの方法ではルートの変更には通信するノード対で使用する IP アドレスを変更しなくてはならず，MPI のような環境でこれらの変更を適用するためには，一度プログラムを終了する必要がある．一方で大塚らの実装でのルート変更処理はスイッチの設定変更で行うことができるが，そのためには，スイッチに対する何らかのインタフェースが必要になる．そのためこれらの 2 つの実装では動的ルーティングテーブルの変更を行うことは現実的ではない．

我々の実装では，ルーティングの管理は送信ノードにおけるパケットへの VID 割当のみで行われ，これはソフトウェア制御で実現されている．そのためルーティングを変更するためには VID への割当を切り替えればよく，通信中においても可能である．その結果，VFREC-Net システムは動的なルーティングテーブル変更が実現できる．

7. おわりに

本論文では VFREC-Net における動的ルーティングテーブルの設定手法を提案し，それを実現するためのフレームワークを開発した．これを用いることで VID の割付けの最適化を行い，ネットワーク上の経路制御を積極的に行うことが可能になる．このようなことから，各種通信負荷分散制御アルゴリズムを VFREC-Net に適用可能になっている．このフレームワークは各ノードで動作する新規に開発したデーモンプログラムである VFN デーモンによって提供される．今回は，このフレームワークに対して MPI 上から制御するライブラリもあわせて開発した．このようなシステムを用い，ユーザが適切にルーティングテーブルを設定することで，より複雑な通信パターンを持つアプリケーションにおいても，ネットワークの持つ性能を最大限に利用できると期待できる．

参考文献

1) IEEE: 802.3ad - Link Aggregation. http://standards.ieee.org/getieee802/802.3.html.
2) 工藤知宏ほか：VLAN を用いた複数パスを持つクラスタ向き L2 Ethernet ネットワーク，情報処理学会論文誌 コンピューティングシステム，Vol.45, No.SIG06(ACS6), pp.35–44 (2004).
3) 三浦信一ほか：VFREC-Net：ドライバ制御による VLAN を用いたマルチパスネットワーク，情報処理学会論文誌 コンピューティングシステム，Vol.47, No.SIG12(ACS15), pp.35–45 (2006).
4) IEEE: 802.1Q - Virtual LANs. http://www.ieee802.org/1/pages/802.1Q.html.
5) Graham, R.L. et al.: Open MPI: A Flexible High Performance MPI, *Proceedings, 6th Annual International Conference on Parallel Processing and Applied Mathematics*, Poznan, Poland (2005).
6) 大塚智宏ほか：スイッチでタグ付けを行う VLAN ルーティング法，情報処理学会論文誌コンピューティングシステム，Vol.47, No.SIG12(ACS15), pp. 46–58 (2006).