大域的な情報を用いる相互結合網方式 Cross-Line

横田隆史[†] 西谷雅史[†,☆] 大津金光[†]
古川文人[†,☆☆] 馬場敬信[†]

大規模並列計算機システムの構築にあたり，効果的な相互結合網を検討することは必要不可欠である．大規模化の実現のためには，機能の集中を避け分散化しながら高性能を達成することが求められる．このため，大規模相互結合網では，独立して動作する多数のスイッチ（ルータ）を用い，それらの協調動作の結果として結合網機能を実現することになる．これまでに，結合網の転送性能を向上させるため，適応ルーティングなど様々な方式が提案されてきたが，さらに効率の良い方式を求めるには広域的に最適な制御手法の検討が必要である．本論文は，大域的な情報をルータ間で流通させることにより効果的なルーティング制御を行う方式 Cross-Line を提案する．Cross-Line は，各ルータにおける仮想チャネルの状況を 1 ビットで表現し，さらにパケットの進行方向の情報のみを収集することで，現実的かつ効果的な広域制御手段を提供するものである．簡単なモデルによる解析的な評価と，シミュレータによる大規模網の評価により，Cross-Line の有効性を明らかにする．

Cross-Line: A Novel Routing Algorithm That Uses Global Information

Takashi Yokota,[†] Masashi Nishitani,[†,☆] Kanemitsu Ootsu,[†]
Fumihito Furukawa[†] and Takanobu Baba[†]

Interconnection network is an inevitable component for massively parallel systems. Such systems exclude centralized control and require huge number of decentralized modules in their interconnection networks. Furthermore, the networks should be powerful and efficient. An ordinal interconnection network is composed of many independent routers that can cooperate as a communication subsystem. Many routing algorithms are proposed in the past, although, they did not use wide-area information. In this paper, we propose a new routing algorithm Cross-Line, that makes efficient use of global information across the network. By limiting one-bit information of each virtual channel and by limiting distribution area, the algorithm achieves global routing control. Analytical and simulation results reveal the effectiveness of the algorithm.

1. はじめに

大規模並列計算機システムの効果的な構築にあたっては，相互結合網に関する考察を抜きに考えることはできない．どのような処理機能を単位（ユニット）として扱うのか．それらをどのように接続するのか．そこで通信はどのように行われるのか．その結果，期待する性能が得られるのか．また，システムは経済的に適当なコストで構築可能なのか，といった課題に適切に応えられなくてはならない．このように相互結合網は並列計算機の構築にあたって非常に重要な位置を占めており，これまで様々な方式が提案され現実の並列計算機に適用されてきた[1),2)]．

本論文では，メッセージ通信により並列計算を行うシステムのなかで，演算ノード間を結ぶ転送路によりノード間でメッセージ（パケット）を転送していく直接網を対象として，転送効率の向上のためのルーティング方式を論じる．

パケットは，送信元から受信先に向けてノード間を転送される．この転送のしかた（ルーティング）により，網の性能が変化する．ルーティング手法は，転送の経路が送信元と受信先の組合せにより一意に決まる決定的（非適応）ルーティングと，転送途中で網の状況に応じて変化する適応ルーティングとに大別される．多くのトポロジでは，送信元と受信先の間に複数の経路を設定できる．適応ルーティングは，輻輳などの網

† 宇都宮大学
Utsunomiya University
☆ 現在，株式会社日本総合研究所
Presently with The Japan Research Institute, Limited
☆☆ 現在，帝京大学
Presently with Teikyo University

の状況により適切な経路を選択することで全体の性能を改善する．

網の状況により適切な経路を選択することが適応ルーティングの基本的な考えであり，既往研究により有効性はある程度示されている．これらの既存手法は現実的かつ効果的なものも多いが，一方で，相互結合網全体の状況から見て真に適切な経路選択を行う保証がないという問題を残している．既存手法の多くは，網を構成する個々のルータにおいて，そのルータ自身の内部状態，ないしその隣接ルータの局所的な情報に基づいて状況を推測し，経路選択を行っている．これらは局所的な問題の解決になっているかもしれないが，必ずしも網全体レベルでの最適な解にはなっていない．

これは相互結合網の構築のしかたに起因する根源的な問題である．直接網では各ノード上にパケットの転送制御を行うルータを設け，ルータ間の接続によりシステムを構成する．すなわち，小規模のスイッチ（ルータ）が分散配置され，スイッチ（ルータ）ごとに独立して制御されている．大規模並列システムを考えた場合，ただ 1 つの制御機構を設け集中的に制御することは現実的ではないため，ルータの単位で分散制御することを前提にせざるをえない．

このような多数のルータの独立動作による分散制御を行わなければならないという制約があっても，網全体にわたる適切な転送制御を行えるだけの情報を流通させることができれば，既存の適応ルーティングの手法を大きく改善できる効果を得られることが期待できる．そのためには，ルータのごく近傍に限られていた情報を現実的な範囲で拡大し，個々のルータで大域的な状況を把握し，より適切な経路判断ができるようにすることが肝要である．

本論文では，大域的な情報によるルーティング制御により性能の向上を目指したルーティング方式 Cross-Line を提案する[3),4)]．Cross-Line では，パケットの転送経路の選択肢となっている方向について，そのまま「直進」した場合の経路にあたるノードでの輻輳状況を広域情報として収集する機能を持つ．この広域情報に従って経路選択を行うことにより，輻輳に対する強い耐性を備えたルーティングを行えるものと期待できる．さらに，広域情報をパケットが転送されない時間を用いて転送することにより，専用の情報経路の増設などの大きなハードウェア負担を招くことなく実現可能である．

以下，本論文は次のような構成をとる．まず 2 章において本論文で前提とする事項を明らかにし，大域的なルーティング制御のための基本的な検討を行い，ルーティング方法の基礎的な設計方針を示す．その結果を受けて 3 章において具体化し，新しいルーティング方式 Cross-Line を示す．またそこでは，簡単なモデル化により，Cross-Line が大域的な輻輳回避能力を備えることを解析的な方法で示す．次いで 4 章では，専用のシミュレータにより評価した結果を示し，Cross-Line の有効性を明らかにする．5 章で関連する研究について述べ，本論文での新規性を明確にし，最後に 6 章でまとめる．

2. 大域的な情報によるルーティング制御

まず本章において，本論文で前提とする事項について明らかにし，そのうえで大域的な情報によるルーティングのための基本的な検討を行う．

2.1 前　　提

(1)　結合網の形式

本論文は，大規模並列計算機システムを直接網により構築する場合に，相互結合網ルータで行われるパケット転送制御の手法を扱う．パケット転送制御の手法の中でも，特に，各ルータ上で，パケットの転送方向の選択肢が複数ある場合に，どの方向に転送するのが適切かをそのときの状況に応じて判断し切り替える手法について議論する．つまり，適応ルーティングにおいて，パケットの転送出力方向を判断する手法に限っての議論を行う．これは，routing policy[5)]（ないし selection function）に属するものである．

本論文で対象とするのは，ノード間を複数のリンクにより直接接続することでシステムを構築する形式の直接網である．各ノード上には，計算を司るプロセシングユニットと，結合網機能を提供するルータユニット（単にルータと称する）がある．各ノードのルータを相互に接続することでシステム全体にわたる相互結合網を構築する．ルータおよびプロセシングユニットは通信のためのポートを持つ．ルータ間およびルータ・プロセシングユニット間は，各々のポートの間を単方向性の接続リンクで接続する．ここにあげたポート以外の通信路は用いない．たとえば，相互結合網の制御を行う専用の通信路などは設けない．

また，議論の簡便化のため，網のトポロジは k-ary n-cube に代表される均質な構造をなすものを前提とする．

(2)　転送の単位と方式

パケットを単位としたメッセージパッシングにより通信を行うシステムを仮定する．転送の単位は，目的のデータと送信先などの情報とをひとまとめにしたパケットであり，送信元から送信されたパケットは受信

先に到着するまでの間に廃棄されることがなく，したがって再送の機能も考えない．また転送の途中で複数のパケットに分解したり，複数のパケットを併合したりせず，基本的に送信元で生成されたパケットをそのままの形で受信先に配送する．

また，パケットを単位に配送を行うパケットスイッチングを基本とし，サーキットスイッチングは除外する．パケットスイッチングであれば，それ以上の配送制御方法は問わない．すなわち，wormhole（WH），virtual cut-through（VCT），store-and-forward（SF）のいずれであっても，本論文での議論には本質的には影響しない．

(3) 転送制御の方法

パケットの転送制御は，ready/busy 信号を用いて転送を on/off させる方法を前提に考える．credit ベースなどの制御手法でも議論の本質には影響しない．基本的に，バーチャルチャネル（VC）の単位で配送制御を行い，ルータの各入力ポートに VC 単位でパケット受信用のバッファ（パケットバッファ）を持ち，バッファに空きがあれば前段ルータからの転送を可能とする．むろん ready/busy 信号は VC ごとに設定する．バッファに空きがなければ転送はブロックされ，他の VC に切り替えられる．

(4) 転送経路

パケットが送信元から出て受信先に至るまでの経路は，最短経路を通るものとする．すなわち，輻輳を避ける目的であっても，最短でない迂回経路を通ることはしない．これも本論文での議論において本質的な事項ではないが，徒らに議論が複雑化するのを避けるために，最短経路での適応ルーティングを前提とする．

(5) デッドロック防止策

議論の簡単化のためデッドロックを発生しない系を仮定する．相互結合網をデッドロックフリーにするか否かは，本論文で対象としているパケットの転送方向を決めるための議論と直交する概念であるため，以降では議論の簡単化のためデッドロックを発生しない系を仮定する．

このほか，本論文で行う議論の対象以外の部分，たとえば，適切な VC の選択方法などについても，本論文では議論の複雑化を避けるために簡素なモデルを基盤にした議論や評価を行う．

2.2 設計方針

まず本論文で目標とする，相互結合網の広域的に最適な制御の考え方を明確にする．

パケットの転送がブロックされている状態が，相互に隣接する複数のルータにまたがって生じている状態を，**輻輳状態**と定義する．そして，輻輳状態にある部分を，本論文では**輻輳領域**と呼ぶ．輻輳領域は，パケットの転送パターンや，網上にあるパケットの一時的な偏りによって，部分的に発生するものと考えられる．網全体が輻輳状態になるのは，当初部分的に発生した輻輳領域が，パケットの転送ブロックが連鎖することで拡大した結果と考えればよい．

本論文では，こうした輻輳領域を広域的な制御によって避けるルーティング手法を考える．すなわち，パケットの転送途中において，輻輳領域が分かっており，ルーティング経路の変更により輻輳領域の通過を避けられるならば，転送効率が向上するとともに輻輳領域が拡大するのを防止できるものと考える．

そこで，既往の適応ルーティング手法ではルータのごく近傍に限られていた情報を現実的な範囲で拡大し，これにより個々のルータで輻輳領域に関する大域的な情報を把握して，より適切な経路判断ができるようにする．そのために，本論文では以下の事項を検討していく．

(1) 結合網の中のどのような事項をルーティングのための情報として扱うべきなのか．
(2) ルータはその情報をどのように収集するのか．
(3) そして，各ルータでどのようにして適切なルーティングを判断するのか．

本節では，特に基本的な (1)，(2) の項目について，以下で検討する．

2.2.1 収集する情報

輻輳領域では，パケット転送がブロックされている状態，すなわち，ルータの入力ポートにあるパケットバッファが busy 状態になっている．逆に，非輻輳領域では，一時的にパケットバッファが busy になる場合があるが，平均的には ready 状態が支配的であると考えられる．このため，パケットバッファの ready/busy を表す情報が，輻輳領域を検出するために最も有効である．

パケットバッファの ready/busy 状態により輻輳領域を検出する場合，busy 状態が比較的長時間続くのか，すぐに解消するのかを判断しなければならない．しかし本論文では，このような厳密な輻輳領域の検出は行わず，単にパケットバッファが busy 状態であれば輻輳状態と見なす．これは以下の理由による．

たとえば，パケットバッファの容量がパケットのサイズよりも十分大きい場合，busy 状態になるということは，そのバッファに多くのパケットが転送された結果，輻輳に近い状態になっていることを表している．wormhole ルーティングのようにバッファのサイズが

小さい場合は，1 つのパケットのブロックが広範囲のルータの転送に影響する．このため，輻輳領域の検出に正確さがやや失われると考えられるが，定性的には busy 状態を検出したルータが輻輳傾向にあることに変わりない．すなわち，busy 状態の継続時間を測る必要はなく，パケットバッファが busy 状態であることをもって輻輳状態と見なしても大きな損失はない．

以上から，輻輳の情報を，ルータのパケットバッファの ready/busy の情報で置き換えて表現することができること，また，ルーティング制御のためにはそれだけの情報で十分であることが分かる．すなわち，パケットバッファの状態を表す 1 ビットの情報を設け，その情報を結合網の広域情報として流通させ，各ルータでのルーティングの判断に使用する手段を提供すればよい．

2.2.2 輻輳情報の収集・分配・共有方法

このようにして輻輳を表現するための情報を絞っても，結合網全体を表現しようとすると相当の量になるため，必要なだけの輻輳情報を収集・分配する方法が必要である．

そこで我々は，パケットが「直進」した場合を仮定し，その直進経路上にあるルータの輻輳情報（すなわち入力バッファの ready/busy 情報）のみを集めることを考えた．たとえば 2 次元トーラスの場合には，パケットが x 軸方向に直進した場合の経路上にあるパケットバッファの状態と，y 軸方向に進んだ場合のそれを収集する．両者の比較により，空いていると判断できる方向にパケットを配送する．こうすることで，たとえば $N = n \times n$ のシステムにある $O(N)$ の情報のうち，実際に収集するものを $O(\sqrt{N})$ に抑えることができる．さらに収集に要する時間も同様に短縮されるため，効果的な配送制御を行えることが期待できる．

3. Cross-Line アルゴリズム

3.1 VCionfo：バーチャルチャネル情報

前章の基本設計を受けて，具体的な実現方法を考える．前章までは，相互結合網のトポロジを規定しておらず，さらにパケットの進行方向に関して「直進する」といった抽象的な表現を用いていたが，本章以降では，議論の簡素化のため，具体的に，2 次元トーラスに絞って議論を進める．

我々は，パケットの「直進」とは，ある VC を使用しているパケットがルータのある出力ポートから出力され，転送先のルータにおいても同一方向の出力ポートから出力されていくことと定義する．2 次元トーラスにおいては，たとえば，x 軸の正の方向に出力されたパケットが以降のルータにおいても x の正の方向に出力されることを指す．このように情報の収集範囲を「直進」の方向に限ることにより，あるルータから見て，x 軸 y 軸に沿った 4 つの方向にある他のルータの入力パケットの ready/busy 情報を収集すればよい．ここで，x 軸に沿った正の方向を $x+$，負の方向を $x-$ とする．y 軸に関しても同様に $y+$ 方向と $y-$ 方向を規定する．

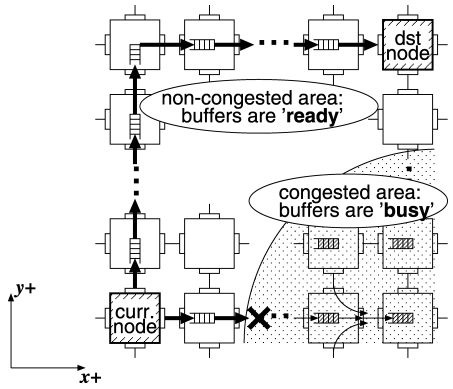
2.1 節 (3) で述べたように，ルータ間の転送は VC ごとに設けられたパケットバッファの ready/busy 信号で制御される．このため，各ルータは隣接ルータでの ready/busy 情報を，ハンドシェーク信号の状態として直接センスするができる．

たとえば，$x+$ 方向の出力ポートのハンドシェーク信号を輻輳情報として $x-$ 方向に転送する．この情報を受けた $x-$ 方向の隣接ルータは，そこから $x+$ 方向に 2 つ先の位置にあるルータの状態を知ることになる．このようにして転送されてきた輻輳情報と，自ルータの出力ポートのハンドシェーク信号とをまとめて，さらに $x-$ 方向にあるルータに転送すればよい．

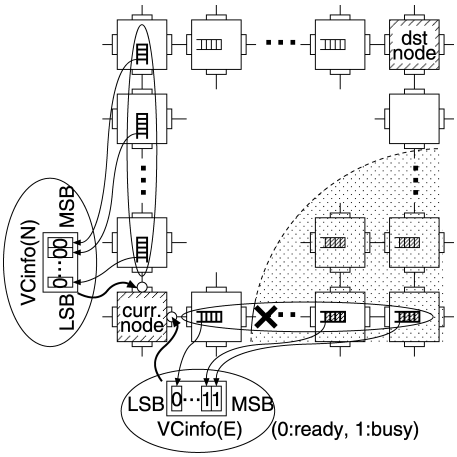
こうして $x+$ 方向から送られてきた輻輳情報を左に 1 ビットシフトし，最下位ビットに自ルータのハンドシェーク信号を付加する．個々のルータで検出した輻輳情報は，ルータ間で転送されるたびに 1 ビット左にシフトされる．これによって，転送情報のビット位置から輻輳情報の該当位置を特定することができる．このようにして，x 軸，y 軸に沿って情報を転送するだけで，$x+$，$x-$，$y+$，$y-$ の各方向についての輻輳情報を構築し，流通させることが可能になる．こうして得られた情報を **VCinfo**（virtual channel information）と呼ぶ．

図 1 に，その基本的な考えを図示する．輻輳状態にある領域（図 1 (a) 中ハッチングで示した領域）ではパケットバッファが busy になっており，非輻輳領域では ready になっている．各々のバッファの ready/busy 情報を 1 ビットで表現し，$x-$，$y-$ 方向に伝播する．この情報を受けたルータ（図中 `curr． node`）では，自ノードから見て $x+$ 方向，$y+$ 方向にあるバッファの状況を把握することができる．図 1 の例では，輻輳領域を避けるには $x+$ ではなく $y+$ 方向にパケットを進めるべきであることが分かる．

VCinfo は，ルータの各出力ポートの各仮想チャネルごとに設置する．パケットをある出力ポートの仮想チャネルから出力しようとしたとき，そこに設置してある VCinfo を参照することで，そのパケットが「直



(a) 輻輳領域とバッファの状態



(b) バッファの状態を保持

図 1 進行方向にあるバッファの状態を表す VCinfo
Fig. 1 VCinfo represents buffers' status in a straight direction.

進」する経路にあるルータの当該チャネルの状況を知ることができる．デッドロックの防止のため，転送途中で仮想チャネル番号が変更されることがある．たとえば，パケットがあらかじめ決められた date-line を越えて転送された場合にチャネル番号を変更する方式がこれに該当する．この場合には，VCinfo の内容もチャネル番号の変更に対応した適切なものをいれる．

輻輳情報の伝播には，特別なハードウェアの追加を前提とはせず，通常のパケットの転送のためのポートと接続リンクを共用することを仮定する．相互結合網，特に大規模な直接網において，接続リンクを常時100%使用していることは稀である．送るべきパケットがない状態，あるいは，隣接ルータのバッファが busy であるために転送がブロックされている状態が少なからず存在する．このように接続リンクがパケットの転送に使用されていないときを使って，輻輳情報を転送すればよい．通常のパケット転送を優先するために，輻輳情報の転送により通常パケットの転送が待たされることはない．逆に輻輳情報の伝播の遅れが問題となる可能性があるが，実質上，致命的とはならないことが，たとえば文献 6) などで示されている．

3.2 VCinfo によるルーティング制御

上述のように収集した VCinfo を用いてパケットのルーティング制御をする方法を考える．まず，VCinfo の各ビットが表す内容を定義する．0 を ready，1 を busy とする．VCinfo の最下位ビットは，隣接ルータの入力バッファの ready/busy を表す情報であり，これは当該ルータにおいて，対応する出力ポートのハンドシェーク信号から直接知ることができる．VCinfo のそれ以外のビットについては，前節に示した方法によってパケットの進行方向と逆向きに転送されてきたものを用いる．

ルーティング制御は以下のように行う．まずパケットが自ノード宛かを確認する．自ノード宛の場合は，プロセシングユニットに向けて出力する．自ノード宛でない場合には，出力方向の選択肢があるか否かを判断する．選択肢がなく唯一の方向にしか進めなければ，その方向に向けて転送する．

x 軸，y 軸のどちらにも進む可能性がある場合は，該当する出力・仮想チャネルにある VCinfo を，最下位ビットから順に比較してパケットの転送方向を決める．下位ビットから順に比較していき，ビット値の違う箇所が見つかった時点でやめ，そのとき ready であった方向に転送する．ただし，比較するビット数は，当該ルータから受信先まで間の x 軸，y 軸の各方向についてのホップ数の小さい方を上限とする．上述のとおり，VCinfo の最下位ビットは隣接ルータの入力バッファの ready/busy 状態を表しているため，一方の仮想チャネルが busy のため転送がブロックされており他方が ready である場合には，VCinfo の最下位ビットの比較のみで転送方向が決まる．

このようにして VCinfo をもとにルーティング方向を決めるアルゴリズムをプログラミング言語風に表現した例を**図 2** に示す．この図は，どの方向のポートからパケットを出力するかを決めるアルゴリズムを抽象的に表したものである．簡略化のため，x 軸 y 軸のどちらの方向か（$x+$ か $x-$ かなど）は区別していない．ルータの位置（`cx`，`cy`）とパケットの宛先（`dx`，`dy`）から一意に求められるためである．また，`goto_X`，`goto_Y` は，それぞれ x 軸方向，y 軸方向の出力ポートに転送することを意味している．`next_x()`，`next_y()` は，パケットが x 軸，y 軸方向に直進する場合の隣接

```
route( int cx, int cy, int dx, int dy ){
  // (cx, cy) ... ルータの位置
  // (dx, dy) ... 受信先ノードのアドレス
  int  wx = cx;
  int  wy = cy;
  for( i=0 ; i<MaxHops ; i++ ){
    // x 軸, y 軸それぞれのホップ数を越えていないか確認
    if( wy==dy || wx==dx )
      break;
    // VCinfo を最下位ビットから比較する
    if( VCinfo_x[i]==READY && VCinfo_y[i]==BUSY  )
      goto_X;
    if( VCinfo_x[i]==BUSY  && VCinfo_y[i]==READY )
      goto_Y;
    wx = next_x(wx);
    wy = next_y(wy);
  }
  // VCinfo の比較では決まらなかった
  // → x 軸, y 軸のホップ数の多い方向へ進む
  if( wx==dx && wy!=dy )  goto_Y;
  else                    goto_X;
}
```

図 2 Cross-Line のルーティングアルゴリズム例
Fig. 2 An example routing algorithm of Cross-Line.

のアドレスを求める関数であり，たとえば $x+$ 方向の場合 `if( ++X >= n ) X=0;` となる．また VCinfo に付せられた [] は，LSB を 0 としたビット位置を表している．

3.3 Cross-Line ルーティングの妥当性

上で示した方法により，結合網全体にわたる適切なルーティング制御が行えることの検証を行う．

ここで，4 章で用いているホットスポット通信のように，系の特定の領域（たとえば中心）に向かう通信が多い状況を仮定しよう．この場合，パケットは中心部に向かって集まる傾向を示すため，中心部に近いほどパケットの密度が高くなる．ルーティング途中のパケットはパケットバッファに保たれるため，ブロックによりバッファに滞留する度合いが高い中心部ほど，パケットバッファが busy になっている可能性が高い．パケット密度の高い領域を避けるように制御できるルーティングアルゴリズムほど，輻輳回避能力が高いといえる．

そこで，結合網の輻輳の度合を表現するための指標（輻輳度）を仮定し，これにより網で発生している輻輳をモデリングする．この輻輳度はノード位置の x, y 座標の値をもとにした関数 $C(x,y)$ で表現できるものとする．さらに，$C(x,y)$ がある一定の値（閾値）よりも大きい場合にルータの入力バッファが busy 状態に，下回っている場合には ready 状態となるものとする．輻輳度関数 C と各ルータでの入力バッファの状況，および VCinfo 各ビットの値との関係を示す概念図を，**図 3** に示す．この図は，輻輳度が高ければパケットバッファが busy になるが，そうでなければ ready である，という性質を定性的に表現したものである．

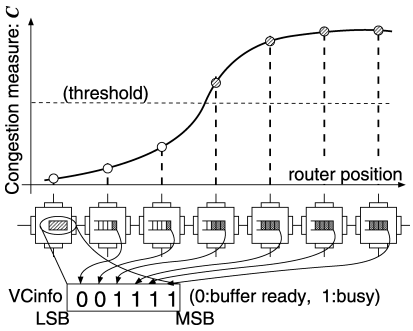


図 3 輻輳度 C と VCinfo の関係
Fig. 3 Cogestion measure C and VCinfo.

輻輳度関数を，たとえば転送途中のパケットの存在密度など具体的な指標で表現することも可能であるが，本論文では議論の一般化を図るため，輻輳の測度として抽象化した表現形式を用いる．

VCinfo 内のビットの並びにより，x 軸，y 軸に沿った $C(x,y)$ の傾きを近似することができる．VCinfo の最下位ビットから検索したとき，ready 状態があまり連続せずに busy 状態が見つかった場合を考える．busy 状態にあるルータの近辺で輻輳度 $C(x,y)$ が閾値を超えていること，さらに最下位ビットからの ready の連続数が少ないことから，当該軸に沿って $C(x,y)$ の傾きが大きいことが推測される．逆に ready の連続数が多く，VCinfo の上位ビットまで busy 状態がなければ，それは，輻輳度 $C(x,y)$ の傾きが緩慢か，あるいは傾斜が負の方向になっていることを表していると解釈してよい．

つまり，パケットの送出先候補となっている方向に対して，対応する VCinfo の下位ビットから比較を行うことで，近似的ながら，輻輳度 $C(x,y)$ の傾斜がより緩慢な方向を求めることができる．**図 4** にこの例を示す．

このように，VCinfo に保持された情報を比較してパケットの適切な転送方向を判断するやりかたは，パケットの転送経路上にあるすべてのルータで行われる．すなわち，パケットは転送経路上の各ルータにおいて，輻輳度の増加の傾きの少ない方向を選択して進んでいくことになる☆．

☆ 明示していないが，むろん，2 つ以上の転送方向の選択肢があることが条件である．

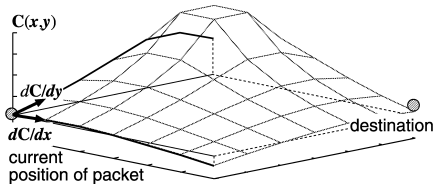


図 4 輻輳度の傾きを比較する
Fig. 4 Comparing gradients of congestion measure.

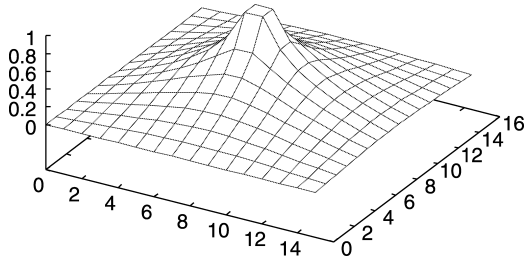


図 5 評価に用いた輻輳度分布
Fig. 5 Congestion measure distribution used for evaluation.

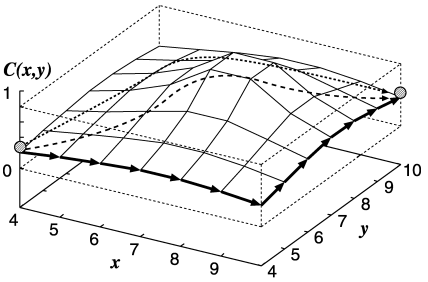


図 6 経路積分の例
Fig. 6 Example of path integral.

表 1 ルーティングアルゴリズムと経路コスト
Table 1 Path costs of some routing algorithms.

ルーティングアルゴリズム	経路コスト
dimension order	5.504965e+04
random walk	5.628108e+04
adaptive	5.245490e+04
cross-line	4.766727e+04
(optimal)	3.800551e+04

VCinfo を用いてルーティング制御をすることの効果を定量的に表現するために，以下の評価を行った．

16×16 の 2 次元トーラスを仮定し，輻輳度の分布を**図 5** に示すように設定した．$x = 0$，$y = 0$ のノードでは $C = 0$，中心部の 4 つのノード $(7,7)$，$(7,8)$，$(8,7)$，$(8,8)$ で $C = 1$ とし，他のノードの値は

$$\frac{\partial^2 C}{\partial x^2} + \frac{\partial^2 C}{\partial y^2} = 0$$

を差分法により求めた値とした．これは，中心部にあるノードに向かってパケットの転送が集中するホットスポット通信の状況をモデリングしたものである．

パケットが送信元から出て受信先に達するまでの経路上にある各点での輻輳度 C の総和を求める．この総和値を求める演算は，輻輳度 $C(x,y)$ の値を送信元から受信先までの経路に沿って積分する，経路積分に相当する．この総和値は，パケットが輻輳度の低い経路をとるほど小さくなる．そこで，以降，この値を**経路コスト**と呼ぶ．この経路コストの大小により，パケットの配送制御の適切さを比較できる．すなわち，経路コストが大きいとき，パケットは輻輳度の高い部分を通ったことになる．経路積分の例を**図 6** に示す．図は $(4,4)$ から $(10,10)$ に向かう場合の例であり，図のような輻輳度分布になっている．経路コストが最小になる経路を，太線で表示している．

16×16 の中の 2 つのノードのすべての組合せについて，ルーティングアルゴリズムに従ってパケットが進行する経路に沿った経路コストを求め，それらの総和を求めた．ルーティング方法によっては送信元から受信先までの経路（往路）と，逆に受信先ノードから送信元に至る経路（復路）が異なる場合があるため，2 ノードの組合せそれぞれについて，往路と復路の両方の値を算出して加えている．結果を**表 1** に示す．表中，dimension order は x 軸方向に先に進む決定的なルーティング方式，random walk は，最短経路の範囲内で途中経路でのパケットの転送先をランダムに決めた場合，adaptive は隣接ノードとの ready/busy 情報のみを使った場合である．なお，optimal は送信元・受信先の間のすべての最短経路について経路コストを計算し，そのなかから最小のものを選んだ場合の値であり，理論的最小値となる．この方法は現実には実現しえないが，比較のため表示している．なお，random walk のみ 100 回の試行の平均値を用いている．他の値は一意に決定するため，複数回の試行をする必要はない．Cross-Line と adaptive では，与えられた輻輳度 $C(x,y)$ をもとに，図 3 に示した方法で各ノードの VCinfo を設定した．ここで用いている閾値は，全ノードの輻輳度 C の平均値である．他の dimension order，random walk，optimal では，VCinfo の値をまったく使わない．

表 1 より，Cross-Line は optimal には及ばないものの，dimention order に比べ約 14%の改善が得られていることが分かる．ルーティングの際に参照する情報の範囲を隣接ルータに限定した adaptive では対 dimension order 比で 5%ほどの改善にとどまってい

ることと比較すると，Cross-Line により高い輻輳回避能力が実現できていることが分かる．

4. 評　　価

4.1 評 価 対 象

4.1.1 ルーティングアルゴリズム

Cross-Line のルーティングアルゴリズムは図 2 に示したとおりである．ただし，VCinfo を参照する最大数（図中 `MaxHops`）を任意に設定できるようにしている．以降では，VCinfo の参照量を最大 n ビットに制限した場合を CrossLn(n) と表す．CrossLn(full) は制限を加えない場合である．なお，ビット数を陽に表さない場合は CrossLn(full) を意味する．

また，比較のため，以下のルーティング方法を用いた．

決定的ルーティング（det.） VCinfo をまったく参照しない場合である．ルーティングアルゴリズムを**図 7** に示す．図中，`dist(x1,x2)` は，x 軸 (y 軸) 方向の x1，x2 の距離を計算する関数である．このルーティングアルゴリズムによると，パケットは $\texttt{dist}(c_x, d_x) \approx \texttt{dist}(c_y, d_y) > 0$ のとき，ジグザグの経路をとる．

次元順（dim.order） x 軸方向にそれ以上進めなくなるところまで進み，その後 y 軸方向を進むルーティングである．アルゴリズムを**図 8** に示す．

理想状態（ideal） VCinfo の代わりに該当するルータの busy/ready 情報を直接参照する方法である．これは実現不可能な方法であるが，Cross-Line で VCinfo の転送が理想的に行われた場合（状態が変化しても遅延ゼロで反映される場合），すなわち Cross-Line の理想的な場合として，比較のために用いる．

評価に用いたルータの構成を**図 9** に示す．ただし，図では仮想チャネル数を 3 として表示している．隣接ルータ間は 2 本の単方向のリンク（往・復）で接続される．ルータは，入力ポートにチャネルごとのバッファを持つ．ルータ間の通信は，チャネルごとに設けられた ready/busy 信号により制御される．チャネルごとにバッファの空き容量があれば，当該ビットが ready となる．

パケットと VCinfo は同じポートを使って転送される．ただし，VCinfo が転送されるのは，ポート上に有効なパケットデータがない場合のみに限られる．このため VCinfo の転送が，パケットの転送に干渉することはない．前段から送られてきた VCinfo は，1 ビット左にシフトしたのち該当するバッファ（VCinfo buffer）に保存される．VCinfo バッファの最下位ビットには，

```
route( int cx, int cy, int dx, int dy ){
  // (cx, cy) ... current router
  // (dx, dy) ... destination node
  if(  dist(cx,dx) >= dist(cy,dy) )  goto_X;
  else                               goto_Y;
}
```

図 7 決定的ルーティング
Fig. 7 Deterministic routing.

```
route( int cx, int cy, int dx, int dy ){
  // (cx, cy) ... current router
  // (dx, dy) ... destination node
  if(  cx != dx )  goto_X;
  else             goto_Y;
}
```

図 8 次元順ルーティング
Fig. 8 Dimension-order routing.

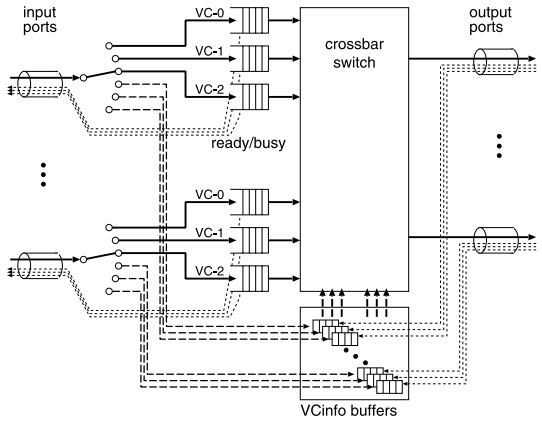


図 9 ルータの構成
Fig. 9 Router organization example.

該当する出力ポートの ready/busy 信号が入る．

パケットはいったんチャネルごとに設けられたバッファに保存され，バッファの先頭にあるパケットがクロスバスイッチにより出力先を決められる．クロスバスイッチは，パケットバッファからの要求を受け，VCinfo を参照して適切な出力ポートを決める．同一の出力ポートに対して，同時に複数のチャネルから出力要求があった場合は，転送ブロック状態の最も長いチャネルを優先して選択する．

4.1.2 仮想チャネル制御

本評価は Cross-Line によるルーティング機能の有効性を確認するために行うものである．結合網の性能は，仮想チャネルの使用方法によっても大きく変化するため，本評価にあたり仮想チャネルの選択方法は，上記ルーティングアルゴリズムのすべてで同一とした．

パケットの送信元を中心にした座標系を考え，送信元→受信先のベクトルがどの象現にあるかにより最初の仮想チャネル番号を決める．第 1，第 3 象現であれ

ば 0，第 2，第 4 象現であれば 1 とする．その後は，パケットが date-line を越えるときに，チャネル番号を 2 だけ増分する．date-line は，システムを $N = n \times n$ の 2 次元トーラスとし，ノードの位置をゼロオリジンの 2 次元座標で表現したとき，$(x, n/2-1)$–$(x, n/2)$，$(x, n-1)$–$(x, 0)$，$(n/2-1, y)$–$(n/2, y)$，$(n-1, y)$–$(0, y)$ の間とした．x，y は，$0 \leq x, y < n$ の整数である．パケットは送信先に到着するまでに最大 2 回 date-line を越えるため，合計 6 本の仮想チャネルを使用している．

4.2 評価環境

二次元トーラス網を模擬するシミュレータを C++ により作成し，上記のルーティングアルゴリズムを実装し，評価を行った．システムのサイズは $N = 32 \times 32$ である．パケット長は 4（フリット）とした．転送はパケット単位で行われるが，隣接ルータ間でのハンドシェークはフリット単位で制御する．このためにパケットが転送途中の状態でも busy によりブロックされることがある．隣接ルータ間でパケット転送がブロックされた場合，当該チャネルの転送は凍結し，別の転送可能なチャネルに切り替える．凍結したチャネルが転送を再開するのは，そのチャネルの転送が 1 パケット分終了したとき，あるいは，そのチャネルがブロックされたときで，かつ，当該チャネルのパケットバッファに空きが生じた場合（ready になった場合）である．各フリットは，ルータ間を最短 1 サイクルで移動する．

評価は，シミュレーション開始後 100,000 サイクル間を無視し，それ以降 200,000 サイクルまでに得られた評価項目を用いている．スループットは，200,000 サイクル時の受信パケット総数から 100,000 サイクル時のそれを引いたものである．

転送パターンは，各ノードが一定の周期でランダムに選ばれた相手ノードにパケットを送信するランダム通信と，ランダム通信の送信先のうち 5%を特定のノード（中心 $(n/2, n/2)$ のノード）にしたホットスポット通信を用いた．パケットバッファの容量は 3 フリットである．バッファ容量がパケット長（4 フリット）未満であるため，wormhole に近い転送特性となることを意図している．

4.3 ルーティングアルゴリズムの比較

ランダム通信の場合の評価結果を**図 10** に，ホットスポット通信の場合の結果を**図 11** に示す．転送スループット（normalized accepted traffic）は，実際にシミュレーション時間内に受信したパケットの総数を正規化して表現したものである．シミュレーション時間 T（本評価の場合 $200,000 - 100,000 = 100,000$ サイクル），システムのサイズ N（本評価の場合 $N = 32 \times 32$），受信パケット総数を P，パケット長 l として，転送スループットは $P/(T/l * N)$ となる．

各図とも，パケットの生成間隔をパラメータとし，パラメータごとに得られた転送スループットと平均レイテンシをプロットし，隣接パラメータ間のプロットを線で結んだものとなっている．パケット生成間隔が広いとき，網は空いている状態であり，転送スループッ

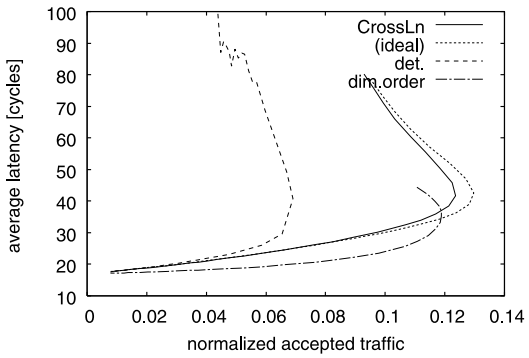
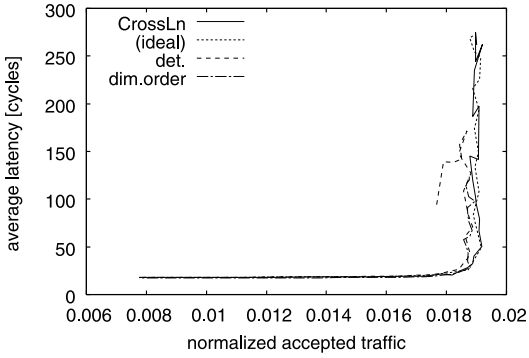
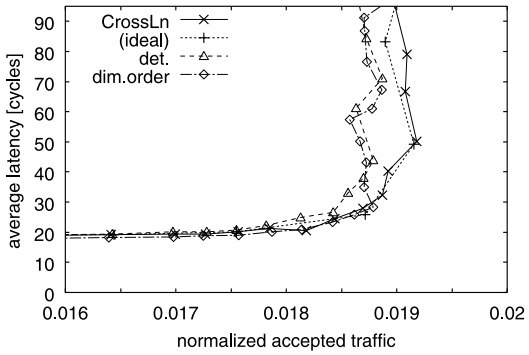


図 10 ランダム通信特性（32 × 32）
Fig. 10 Random traffic performance.



(a) 転送スループット対平均レイテンシ



(b) (a) の部分拡大

図 11 5%ホットスポット通信特性（32 × 32）
Fig. 11 5% hot-spot traffic performance.

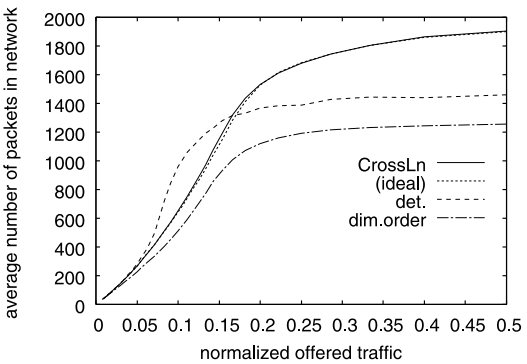
トが小さく，平均レイテンシも小さい．パケット生成間隔を狭めていくと，網を流れるパケットの量が増え，転送スループットが増すが，それとともにパケット間の衝突によるブロックが生じやすくなり，平均レイテンシが漸増する．さらに生成間隔を狭めると，転送スループットが減少に転じるとともに，平均レイテンシが急激に上昇するようになる．生成間隔をこれ以上に狭めると，網が輻輳状態となり，転送スループットは減少し，平均レイテンシは増加していく．最大転送スループットが大きく，平均レイテンシが低いものが望ましい．

図 10 から，ランダム通信時に得られる最大転送スループットを比較する．決定的ルーティングと Cross-Line を比べることで，適応ルーティングの有無による効果を測ることができる．決定的ルーティングに対し Cross-Line は（0.122/0.068=）1.79 倍の最大転送スループットを実現している．Cross-Line では，理想的な場合（ideal）と比較しても約 93%（= 0.122/0.130）の性能が得られている．Cross-Line では，各ノードの仮想チャネルの情報（VCinfo）を，パケット転送をしていないときを利用して伝播させるために，各ノードの状態が変化してからそれをルーティングに反映できるまでに遅延が生じる．離れた位置にあるルータの情報ほど古く不正確な情報になり，Cross-Line の転送性能に影響するものと考えられるが，この結果から，遅延の影響が大きな性能低下をもたらすわけではないことが分かる．

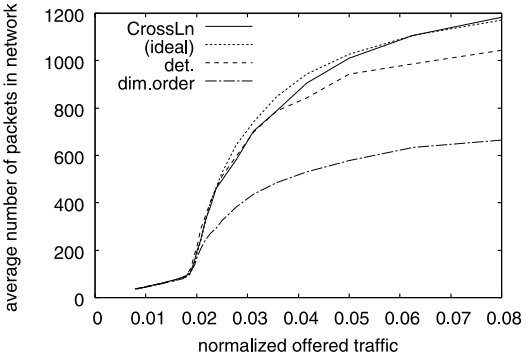
網の性能が飽和状態になる付近では，輻輳状態が網全体に及んでいたり，通信パターンに従った固定的な渋滞パターンが生じていたりするわけではなく，局所的にパケットが流れやすくなっている非輻輳領域が生じる．輻輳領域では，パケットどうしがブロックし合うためにパケットの移動度が低い状態になっている．このため，輻輳領域そのものが系内を移動する速度，すなわち系内の輻輳状態が変化する速度は，非輻輳領域でのパケットのそれと比べると格段に遅くなる．したがって各仮想チャネルの状態が VCinfo として転送される速度のほうが，輻輳領域が移動する速度よりも速いと考えられる．Cross-Line が理想的な場合（ideal）に比べ大きく性能低下しないのは，このような理由のためと考えられる．

また，Cross-Line は，次元順ルーティングと比較しても，約 4%（= 0.122/0.118）良好な最大転送スループットを得ている．ただし，平均レイテンシは次元順ルーティングの方が低い結果になっている．

5%ホットスポット時の性能（図 11）では，ランダム通信性能（図 10）と比較して顕著な性能差が見られない．最大転送スループットを表す部分（図 11 (a) の右下部分）を拡大し，同図 (b) に示す．最大転送スループットは，Cross-Line およびその理想形（ideal）が明らかに高くなっている．また，パケット生成間隔が長く網が飽和していない状態での転送レイテンシは，いずれのルーティングアルゴリズムでも有意な差は見られない．ランダム通信では次元順ルーティングが明らかに短かった．



(a) ランダム通信の場合



(b) ホットスポット通信の場合

図 12 網上のパケット総数（32 × 32）
Fig. 12 Number of packets in the network (32 × 32).

4.4 網の状態の考察

こうした特性が何に起因するのかを解明するため，ルータのパケットバッファにあるパケットの総数を調べた．投入スループット（normalized offered traffic）に対する網上のパケット総数を**図 12** に示す．ここで，投入スループットは，パケット長を各ノードでのパケット生成間隔で割った値である．網上のパケット総数は，シミュレータでの評価時間内（開始後 100,000 サイクル以降，200,000 サイクルまで）で，100 サイクルごとに全ルータのパケットバッファにあるパケット数を求め，それらの平均値を求めたものである．

ランダム通信の場合（図 12 (a)），ホットスポット通信の場合（図 12 (b)）とも，Cross-Line のパケット総数が大きく，しかも，投入スループットの増加にともない，増加していることが分かる．決定的ルーティングや次元順のルーティングでは図 12 (a) のようにほぼ一定量に飽和している．

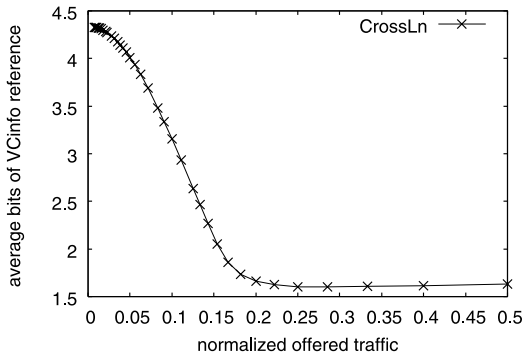
上記の結果からランダム通信の場合（図 12 (a)）について網の状態を考察することにより，前節で得られた転送特性の理由を解明する．

4.1.1 項で述べたように，決定的ルーティングではパケットの現在位置から受信先までの x 軸方向/y 軸方向の距離が近いとき，1 ホップごとに進行方向が x 軸 y 軸で切り替わるジグザグの経路をとる．このために，ルータの中でパケットの x 軸 $\leftrightarrow$ y 軸の転送方向の切替えが多く発生する．これに対し，次元順ルーティングで x 軸 $\leftrightarrow$ y 軸の転送方向の切替えが起きるのは，1 回の転送経路上でたかだか 1 回である．しかし，x 軸方向，y 軸方向にホップする回数は，最短経路ルーティングを行うため，どちらのルーティング法でも同じである．

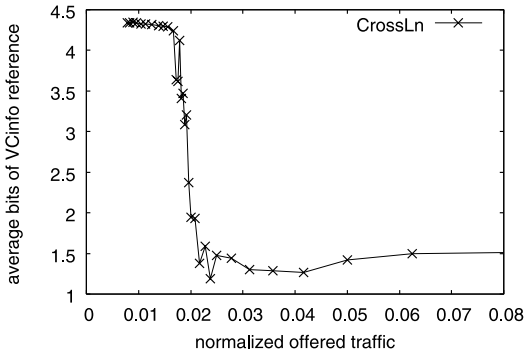
このことから，決定的ルーティングと次元順ルーティングの転送性能の差は，このような転送方向の切替え頻度の差によることが分かる．次元順ルーティングにおいて，x 軸方向のパケットバッファに入ったパケットがクロスバスイッチ上でブロックされるのは，そのパケットが当該ルータで y 軸に進行方向を変えるときのみである．y 軸 $\rightarrow$ x 軸の順で転送方向を変えることはないためである．y 軸方向に進行しているパケットは，同じルータ上で x 軸 $\rightarrow$ y 軸に進行方向を変えるパケットと衝突の可能性があるが，進行方向切替えの頻度が高くないため，衝突の頻度も低いと考えられる．一方，決定的ルーティングではジグザクの経路をとるため，上記のような衝突の頻度が高い．このような衝突機会の差により，次元順と決定的ルーティングとで平均レイテンシに大きな差が生じる（図 10）と説明できる．

また，決定的ルーティングではパケットどうしの衝突にともない転送ブロックが多く発生し，x 軸方向，y 軸方向ともにパケットバッファ内にパケットが多く滞留すると考えられる．このために網上のパケット総数は，次元順よりも決定的ルーティングのほうが多くなる．図 12 (a) はこのことを表している．

Cross-Line は決定的ルーティングをベースにしているため，上記のようなパケットの進行方向の切替えに起因する衝突が多く発生すると考えられる．しかし，網が輻輳状態にないとき（図 12 (a) で投入スループットが 0.15 以下のとき），適応ルーティングによる経路選択が働くために，パケット進行方向の切替えの頻度が抑えられ，また，輻輳領域を回避する方向にパケットを進める．このために，網上のパケット総数も平均レイテンシも，決定的ルーティングに比べ抑えられている．しかし輻輳状態では，網上パケット総数が決定的ルーティングを大きく超えている．これは，適応ルーティングにより，局所的に生じた非輻輳領域にパケットが送られるためと考えられる．つまり，適応ルーティングがうまく働くほど，非輻輳領域まで満遍なくパケットが充填する結果となり，網上のパケット総数と，平均レイテンシの増加をもたらす．



(a) ランダム通信



(b) 5%ホットスポット通信

図 13 VCinfo の参照量（32 × 32）
Fig. 13 Number of referred VCinfo bits (32 × 32).

ホットスポット通信の場合（図 12 (b)），全体の転送スループットが低いために，網が輻輳していない状態では有意な差を確認できないが，輻輳状態（図 12 (b) 中投入スループットが 0.02 以上のとき）では，ランダム通信の場合と同じことがいえる．

4.5 VCinfo の参照特性

Cross-Line では実際にどの程度の VCinfo を参照しているのか，そして VCinfo は何ビットあれば実際上十分なのかを調べた．

図 13 に VCinfo の参照ビット数の平均を示す．投入スループットが大きくなり，網が輻輳傾向になると参照ビット数が減少することが分かる．ランダム通信ではその減り方がなだらかだが，ホットスポット通信では急激に減少していることが分かる．

VCinfo の参照ビット数を制限しての評価結果を**図 14** に示す．同図 (a) がランダム通信時の特性であり，図 14 (b) が 5%ホットスポット通信時の特性である．図 14 (b) では，各プロットの差が分かりにくいため，同図 (c) で投入スループットを横軸，平均レイテンシを縦軸として表現している．

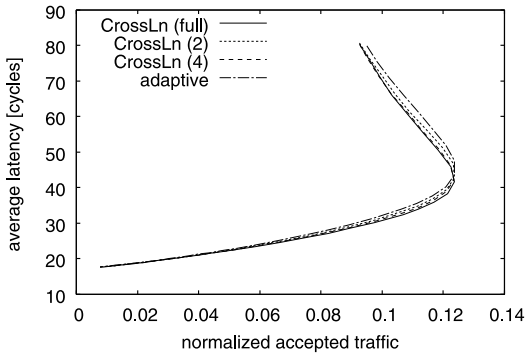
図 13，図 14 において，CrossLn(n) の括弧内の数字 (n) は，Cross-Line において VCinfo の最大参照ビット数を表している．CrossLn(full) は参照ビット数の制限をしていない場合である．また，参照ビット数 1 に制限した場合は，出力ポートでの ready/busy 状態を見てルーティングするタイプの適応ルーティングと等価になる．このため図中では adaptive と表示している．

図 13 によると，輻輳状態での参照ビット数の平均値は大きくない（2 ビット以下）が，転送特性（図 14）によれば CrossLn(4) と CrossLn(full) に有意な差が認められることから，VCinfo の参照は 4 ビット以上あったほうが良いことが分かる．これは，輻輳状態において，大多数の場合はどのルータのバッファもフル状態になっているため VCinfo の内容に差が生じず，有効参照ビット数も少なくなるが，系の一部に輻輳の緩和が生じ，各ルータの VCinfo の内容の差となって表れる状況が起きると，Cross-Line のアルゴリズムにより輻輳を回避する方向が適切に求められているため，と考えられる．

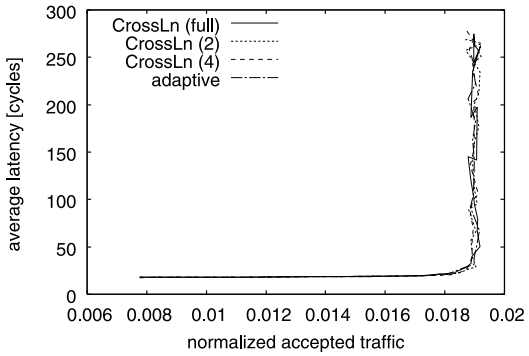
4.6 適応ルーティングとしての性能

図 14 に示されている adaptive と CrossLn を比較することにより，出力ポートでの ready/busy 状態を見てルーティングするタイプの適応ルーティングと Cross-Line との比較を行う．これによって，適応ルーティング法としての Cross-Line の有効性を確認する．

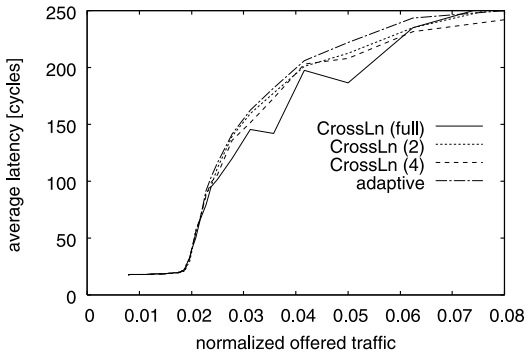
ランダム通信では，図 14 (a) から分かるように，最大転送スループットに大きな差はないが，平均レイテンシは，adaptive よりも Cross-Line のほうが小さくなっている．ホットスポット通信においても，図 14 (c) から，平均レイテンシの差が顕著であることが分かる．輻輳状態（図 14 (c) 中投入スループットが 0.02 以上の範囲）で adaptive よりも Cross-Line のほうが明らかにレイテンシが小さい．ただし，同図から分かるように，Cross-Line では効果のばらつきが大きい．同図中で CrossLn(full) が大きく波打っているのは，ばらつきの大きさを表している☆．



(a) ランダム通信



(b) ホットスポット通信



(c) ホットスポット通信（投入スループット対平均レイテンシ）

図 14 最大参照ビット数の違いによる転送性能の違い（32 × 32）
Fig. 14 Communication performances under maximum VCinfo bits limitation.

以上から，Cross-Line が輻輳領域を迂回してルーティングする能力が単純な適応ルーティングよりも優れていることが分かる．これは，3.3 節で行った評価の結果（表 1）と符合しており，Cross-Line の輻輳回避能力がシミュレーションにおいても確認されたもの

☆ CrossLn(full) のみ，試行ごとのばらつきが大きいため，あえてこのようなグラフにしている．

と考えられる．

5. 関連研究

本論文で提案の Cross-Line は，各ルータ上でパケットの転送方向の選択肢が複数ある場合に，適切な方向を選択する手法，すなわち，routing policy（selection function）に属するものである．

routing policy に関しては，文献 5) で Maximum Shortest Path（MSP）の最適性が主張されている．この手法は，最短経路の適応ルーティングにおいて，経路の選択肢を最大限に保つポリシの下でルーティング制御を行うものである．たとえば，2 次元メッシュのトポロジでは，送信先ノードに向かってジグザグの経路をとる．これは，x 軸方向・y 軸方向の経路の選択肢を最大限に担保するようにルーティングした結果である．本論文で示した Cross-Line ルーティング（図 2）では，VCinfo 値の比較で経路が決定できなかった場合，上記手法と同様に経路選択肢を最大に保つようにパケットを進めるため，ジグザグの経路をとる．このために，本論文の評価で用いた Cross-Line の手法は，MSP に VCinfo による経路選択の機能を組み合わせたものということもできる．

鯉渕らは，カウンタを用い網の混雑状況をモニタすることで適切な方向を選ぶ load-dependent selection function，および，使用されていない時間が最も長いチャネルを選択する LRU（Least Recently Used）selection function を提案している[7]．これらの方法によれば，網の転送負荷を適切に分散することが可能であるが，選択された方向で輻輳領域を回避できることが担保されているわけではない．一方，Cross-Line は，各仮想チャネルの情報を伝播することで輻輳領域を回避することを主たる目的にしている点が異なる．仮想チャネルの情報がそれを必要とするルータに伝播し VCinfo 値として反映されるまでの遅延が問題となるために，輻輳の回避が 100%保証されるわけではないが，遅延が問題にならない距離の範囲内であれば，Cross-Line による輻輳回避が機能することが期待できる．

Thottethodi らは，ルータでパケットの転送に使われるバッファの使用率により結合網の性能（スループット）が大きく変わることを示し，バッファの使用率を指標としてパケットの生成の抑制（スロットリング）を行う方法を提案している[8]．トラフィックの増加とともにバッファに滞留するパケットが多くなる．このバッファ使用率が大きくなると，ある程度まではスループットが上昇するものの，転送パターンごとに定まる閾値を超えると性能が低下する．そこで，ルータのバッファに溜まっているパケットを数え，それらの結合網全体についての総和を求める．この総和値は，ルータのバッファの平均使用率と等価である．閾値を動的に変更するセルフチューニングの機構を持たせることで転送パターンによらない制御方法を実現している．

この方法は，結合網全体の状態をバッファ内にあるパケットの総量として表現している．つまり広域情報を扱った手法といえるが，システムでただ 1 つの値しか扱わないため系の平均的な状態しか検知できない．このため局所性のある輻輳に対して効果を望めない．また，セルフチューニングによる効果が実際に得られるまでの遅延が大きい問題もある．この手法はパケット生成の抑制に適用するものであるから，パケットの転送方向を選択する Cross-Line の方式とは，本来，直交するものである．

So らによる SSR（Speculative Selection Routing）[9] では，各ルータ内に滞留しているパケットの数を求め（Local Utilization Counter，LUC），その情報を近隣のルータに設けた専用のバッファ（Network Condition Buffer，NCB）に配布している．各ルータは LUC 値の配布の範囲に対応する数の NCB を備え，配布された LUC はそれぞれ別個の NCB に保存される．パケットの転送方向は，そのルータ内部の状態と NCB をもとにして決められる．ルータ内部にあるパケットの数を扱う点で上述の Thottethodi らの方法と類似するが，配布範囲を限定し，局所的な転送制御に用いる点で異なる．

SSR は LUC の配布対象の拡大に大きな問題があると考えられる．LUC はカウンタ値であるため，複数ビットを必要とする．個々の LUC を近隣ルータに配布する必要があるため，転送すべき情報量が多い．一方，Cross-Line では，VC ごとに 1 ビットあれば十分であり，さらに複数個の情報をまとめて転送しても差し支えない．

さらに，配布範囲の拡大にともなってルータ内で保持しなければならない NCB の数が増すことも問題である．パケットの転送方向を決めるために，NCB の値を参照することが必要であるから，転送方向決定のコストが大きくなる．実際，文献 9) ではホップ数 1 ないし 2 の範囲が現実的であるとしている．これに対して Cross-Line では，情報を ready/busy の 1 ビットに絞り，さらに配布範囲を直進経路上のみに制限することで，SSR に比べより広域の情報を効率的に扱っている．

Singh らによる GOAL（Globally Oblibvious

Adaptive Locally)[10] とその発展形である GAL (Globally Adaptive Load-balanced)[11] は，パケットの送信元 (s_x, s_y)，受信先 (d_x, d_y) を使って表される quadrant を用いることでルーティングを分散させる手法である．その名のとおりに広域的な最適化や負荷の分散を狙っているが，制御のために広域的な情報を用いることはしていない．この点において，結合網の挙動変化を広域的にとらえてパケットの配送経路を決めようとする Cross-Line のアプローチとは根本的に異なる．

このほか，輻輳の際にパケットを逃すためのキューを（通常の転送用とは）別個に備える方式[12] や，専用のバッファ（in-transit buffer，ITG）を設け，そこに一時的にパケットを退避させることで輻輳を解消する方式[13] なども提案されている．これらは，輻輳の発生に対して個々のルータで行う対処方法であり，結合網全体の状況に応じて輻輳の領域を避ける Cross-Line とは異なる．

6. おわりに

独立したルータ間での協調動作によって実現される大規模並列システムの相互結合網では，広域的に最適な制御を行うことが難しい．本論文では，ルータ間に流通させる情報の量と，流通の範囲を絞ることで広域的な制御を現実的な方法で行うことを検討し，その結果，新しいルーティング方式 Cross-Line を提案した．Cross-Line では，各ルータの各ポートにある仮想チャネルごとに 1 ビットの情報を用いる．チャネルのバッファの ready/busy の情報のみである．さらにこうした情報をルータ間で転送することで流通させるが，転送の向きを「パケットが直進する方向」に限定することで，流通範囲を絞っている．その結果，ルーティング制御に用いる広域情報を現実的な範囲で抑えている．

系の輻輳状況をモデル化し，輻輳の度合いを表す指標を導入し，ルーティング経路に沿って指標値を合算することで，輻輳状況下でのルーティングコストを算出した．その結果から Cross-Line の有効性を定量的に示すことができた．

また，シミュレータによる評価の結果から，Cross-Line の効果を確認することができた．特にホットスポット通信のように，偏りのある通信状況で有効であることが分かった．一方，適応ルーティングにより，パケットの進行方向が分散することで，システム中の配送途中パケットが増すことも明らかにした．これによりルータ内のクロスバスイッチにおいて，busy 状態にある複数のパケットバッファからの出力要求を調停する機会が増す．このためにパケットがブロックされる時間が長くなりレイテンシが増す結果となることが分かった．この現象は適応ルーティング一般に起こりうることであるが，従来の相互結合網の教科書[1),2)] では，こうした説明はされておらず，単に局所的な情報のみを用いる弊害として記述されていた．この問題については，本論文で行ったルーティングアルゴリズムの検討とは別のアプローチが必要であり，今後の課題としたい．

謝辞 本研究は，一部日本学術振興会科学研究費補助金（基盤研究（B）14380135，同（C）16500023，若手研究（B）17700047）の援助による．

参考文献

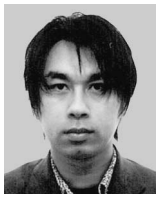
1) Duato, J., Yalamanchili, S. and Ni, L.: *Interconnection Networks: An Engineering Approach*, Morgan Kaufmann Pub. (2003).
2) Dally, W.J. and Towles, B.: *Principles and Practices of Interconnection Networks*, Morgan Kaufmann Pub. (2004).
3) 西谷雅史，絵面 聡，横田隆史，古川文人，大津金光，馬場敬信：動的な情報を用いたルーティングアルゴリズム Cross-Line の検討，情報処理学会研究報告，Vol.2004, No.20, pp.7–12 (2004).
4) Nishitani, M., Ezura, S., Yokota, T., Ootsu, K. and Baba, T.: Preliminary Research of a Novel Routing Algorithm Cross-Line Using Dynamic Information, *Proc. 16th IASTED International Conference on Parallel and Distributed Computing and Systems* (*PDCS 2004*), pp.107–112 (2004).
5) Wu, J.: Maximum-Shortest-Path (MSP): An Optimal Routing Policy for Mesh-Connected Multicomputers, *IEEE Trans. Reliability*, Vol.48, No.3, pp.247–255 (1999).
6) 横田隆史，松岡浩司，岡本一晃，廣野英雄，坂井修一：相互結合網のトポロジを活かしたシステム支援機能とその評価，情報処理学会論文誌，Vol.38, No.4, pp.873–882 (1997).
7) 鯉渕道紘，舟橋 啓，上樂明也，天野英晴：適応型ルーティングにおける Output Selection Function に関する研究，情報処理学会論文誌，Vol.42, No.4, pp.704–713 (2001).
8) Thottethodi, M., Lebeck, A.R. and Mukherjee, S.S.: Self-Tuned Congestion Control for Multiprocessor Networks, *Proc. 7th Int. Symp. on High-Performance Computer Architecture* (*HPCA'01*), pp.107–118 (2001).
9) So, T.C., Oyanagi, S. and Yamazaki, K.: Speculative Selection in Adaptive Routing on Interconnection Networks, 情報処理学会論文誌：コンピューティングシステム，Vol.44, No.Sig 11(ACS

3), pp.147–156 (2003).
10) Singh, A., Dally, W.J., Gupta, A.K. and Towles, B.: GOAL: A Load-Balanced Adaptive Routing Algorithm for Torus Networks, *Proc. 30th Ann. Int. Symp. on Computer Architecture* (*ISCA'03*), pp.194–205 (2003).
11) Singh, A., Dally, W.J., Towles, B. and Gupta, A.K.: Globally Adaptive Load-Balanced Routing on Tori, *Computer Architecture Letters*, Vol.3, No.1, pp.6–9, IEEE Computer Society (2004).
12) Duato, J., Johnson, I., Flich, J., Naven, F., García, P. and Nachiondo, T.: A New Scalable and Cost-Effective Congestion Management Strategy for Lossless Multistage Interconnection Networks, *Proc. 11th Int. Symp. on High-Performance Computer Architecture* (*HPCA'05*), pp.108–119 (2005).
13) Flich, J., López, P., Malumbres, M.P., Duato, J. and Rokicki, T.: Improving Network Performance by Reducing Network Contention in Source-Based COWs with a Low Path-Computation Overhead, *Proc. 15th Int. Parallel and Distributed Processing Symp.* (*IPDPS'01*) (2001).

(平成 17 年 4 月 28 日受付)
(平成 17 年 7 月 19 日採録)

横田　隆史（正会員）

1983 年慶應義塾大学工学部電気工学科卒業．1985 年同大学院電気工学専攻修士課程修了．同年三菱電機（株）に入社，中央研究所，先端技術総合研究所，産業システム研究所に所属．主席研究員．1993 年 12 月から 1997 年 3 月まで新情報処理開発機構（RWCP）に出向．2001 年 4 月より宇都宮大学工学部助教授．計算機アーキテクチャ，設計方法論等の研究に従事．工学博士．ICCD Outstanding Paper Award (1995)，FPGA/PLD Design Conference 審査委員特別賞（2002）各受賞．電子情報通信学会，IEEE 各会員．



西谷　雅史

2003 年宇都宮大学工学部情報工学科卒業．2005 年同大学院博士前期課程修了．同年 4 月より株式会社日本総合研究所入社．技術本部ブレードコンピュータクラスタ所属．

大津　金光（正会員）

1993 年東京大学理学部情報科学科卒業．1995 年同大学院修士課程修了．1997 年同大学院博士課程退学，同年より宇都宮大学工学部助手となり現在に至る．計算機システムの高性能化に関すること，特にマルチスレッドアーキテクチャ，バイナリ変換処理，実行時最適化等に興味を持つ．

古川　文人（正会員）

1998 年宇都宮大学工学部情報工学科卒業．2000 年同大学院博士前期課程修了．2003 年同大学院博士後期課程修了．同年 4 月より宇都宮大学ベンチャー・ビジネス・ラボラトリー非常勤研究員．2005 年 4 月より帝京大学ラーニングテクノロジー開発室助手．博士（工学）．高性能計算機システム，授業改善のためのラーニングテクノロジに関する研究に従事．

馬場　敬信（正会員）

1970 年京都大学工学部数理工学科卒業．1975 年同大学院博士課程単位取得退学．同年より電気通信大学助手，講師を経て，現在宇都宮大学工学部教授．工学博士．1982 年より 1 年間メリーランド大学客員教授．計算機アーキテクチャ，並列処理等の研究に従事．1992 年情報処理学会 Best Author 賞，2002 年 FPGA/PLD Design Conference 審査委員特別賞，PDCS2002 国際会議 Best Paper Award 各受賞．著書 “Microprogrammable Parallel Computer”（MIT Press),『コンピュータアーキテクチャ（改定 2 版)』（オーム社),『コンピュータのしくみを理解するための 10 章』（技術評論社）等．電子情報通信学会，IEEE 各会員．