Irregular Network における Adaptive Routing の提案

舟橋 啓 † 鯉渕 道紘 ‡ 上樂 明也 ‡

† 三重大学 工学部
‡ 慶應義塾大学 理工学部
analysis@am.ics.keio.ac.jp

近年，Network of workstations(NOWs) に代表されるように個々の計算機を高速なネットワークで結び，並列処理を行う研究が盛んに行なわれている．通常ネットワークは wiring flexibility，scalability の要求からトポロジに制限のない irregular network が用いられるが，既存の irregular network のルーティングアルゴリズムは，チャネル使用制限が厳しく，リンクを有効に使うことができない．そこで本論文では自由度の高い adaptive routing である Left-first turn routing(L-turn routing) を提案する．L-turn routing は従来のものと同様にネットワークをツリーに見立て (spanning tree) ルーティングを行うが，ツリーに見立てる際の工夫によりチャネルの使用制限を大幅に緩める．このことにより，ツリーの root 近辺のノードにトラフィックが偏る問題も解決できる．シミュレーション結果より，L-turn routing は効率の良いルーティングを行うためパケットの平均 hop 数が小さく，高い性能を示すことがわかった．

Adaptive routing on irregular networks

Akira Funahashi† Michihiro Koibuchi‡ Akiya Jouraku‡

†Dept. of Information Technology, Mie University
‡Dept. of Computer Science, Keio University
analysis@am.ics.keio.ac.jp

Network-based parallel processing using commodity personal computers has been widely developed. Since such systems require high degree of flexibility and scalability of wiring, a high-speed network with an irregular topology is often needed. In traditional routing algorithms for irregular networks, available paths and virtual channels are considerably restricted in order to avoid deadlocks. In this paper, we propose a novel routing algorithm called Left-first turn routing(L-turn routing), which has an enough freedom in irregular networks by building a specific spanning tree. Simulation results show that L-turn routing greatly improves the performance compared with the traditional routing algorithms in all cases.

1 はじめに

近年，PC/WS の性能向上により Network of workstations(NOWs) [1],[2],[3],[4] などの高速なネットワークで接続された PC や WS で並列処理を行う研究が盛んに行なわれている．これらは同規模の並列計算機よりもコストが安い点が特徴で，将来に渡って高性能計算の主力マシンの一角を占めると予想される．

通常これらのネットワークには wiring flexibility，scalability の要求から irregular network が用いられる．irregular network はトポロジフリーのためルーティングアルゴリズムに対する要求が厳しく，従来の並列計算機で用いられている固定トポロジに比べて，性能上不利な点が多い．

また，irregularity がデッドロックの除去を困難なものにしており，既存のルーティングアルゴリズムはいずれもデッドロックフリーを実現するため，厳しいチャネル使用制限を課している [1],[3],[5],[6]．そのため一部のリンクしか有効に使えず，トラフ

ィックが偏る，パケットの平均 hop 数が増える，などの問題が生じる．

そこで本論文では自由度の高いルーティングアルゴリズム Left-first turn routing(L-turn routing) を提案する．L-turn routing は従来のものと同様にネットワークをツリーに見立て (spanning tree[1]) ルーティングを行うが，ツリーに見立てる際の工夫により，チャネルの使用制限を大幅に緩めることを可能にしている．

L-turn routing は up/down routing[1]，及び prefix routing[6] をエミュレートでき，様々な経路を選択可能であるため全体のチャネル利用率が向上し，traffic が分散される．このため，ツリーの root 近辺のノードに traffic が偏る問題も解決できる．

また，L-turn routing に Silla らが提案した irregular network におけるバーチャルチャネルの追加法である minimal adaptive routing を併用することにより，更なる性能の向上を達成した．

以降，第 2 章では様々な irregular network の形態を隠蔽するモデルについて述べ，第 3 章では irregular network における既存のルーティングアルゴリズムについて述べる．そして，第 4 章では L-turn routing の提案を行い，第 5 章 にてシミュレーション結果を提示，考察を行う．また，第 6 章で今後の課題と結論を述べる．

2 Network Model

irregular network のルーティングはプロセッシングエレメントと switch の接続，switch 同士の接続等，様々な接続ケースを考えなければならない．

しかし，接続ケースに応じてルーティングアルゴリズムを考えるのは得策ではない．そこで，それらの差を隠蔽するネットワークモデルへの抽象化の例として NOWs を取り上げ簡単に説明する．

NOWs は一般的に switch-based network であるが，各 switch はプロセッシングエレメントと個々に直結しているため，パケットを目的地のプロセッサのある switch にルーティングできれば，短時間でデッドロックせずに各プロセッサに到着することができる．よって，ルーティングアルゴリズムは switch 間のルーティングに焦点を当て検討することができる。

switch 間の interconnection network I は corresponding graph G で以下のように表せる．

$$I = G(N, C)$$

ここで，N は switch の集合，C は switch 間の bidirectional link を表している．これにより，図 1のような irregular network は図 2のように表すことができる．図 2のようにネットワークを一般

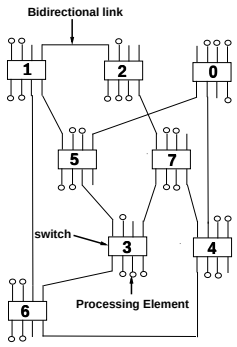

図 1: switch-based interconnection network

化し，ルーティングアルゴリズムを検討していくことにより，ルーティングアルゴリズムは switch 間接続にとどまらず，直接網にも適用することができる．

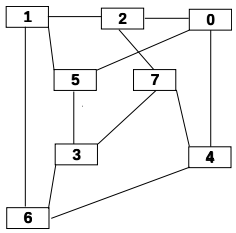

図 2: corresponding graph G

3 既存のルーティングアルゴリズム

irregular network におけるルーティングアルゴリズムはここ数年来，様々な提案がなされており，up/down routing などは既に実装されている [1], [3]．本章では irregular network における代表的な 4 つのルーティングアルゴリズムを deadlock avoidance-based routing と deadlock recoversy-based routing の二種類に分類し，説明する．

3.1 deadlock avoidance routing

3.1.1 primitive up/down routing

primitive up/down routing は単純な方法だが，irregular network のルーティングアルゴリズムの基礎となるものである．

まず，ネットワークを以下の規則に基づきツリーに見立て (spanning tree)，その後ツリーの階層構造を利用したルーティングを行う．

[1]正確には n 個の頂点に対し $(n-1)$ 個の辺を持つ全域グラフのことであるが 本論文では n 個の頂点に対し m 個の辺を持つツリー状の全域グラフと広義する $(n < m)$.

1. ネットワークの中から任意にツリーの root node を選ぶ
2. root node と隣接するノードをツリーに付け加える
3. ツリーの各ノードから隣接するノードをツリーに付け加えていき，全ノードがツリーに入るまでこの作業を繰り返す

上記のアルゴリズムにより，ツリー上の root node 以外のノードは親ノードを唯一つ持つようになっており，親ノードと子ノード以外へのリンクはツリーを作成する際，無視される．例えば図 3 における破線のリンクは実際には存在するが，ツリー上では無視される．

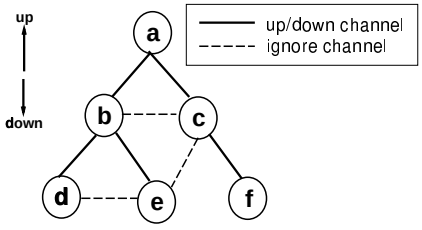

図 3: irregular network のツリー表現

ルーティングアルゴリズムは単純で，パケットの目的地が現ノードの子，孫ノードでない場合，親ノードへパケットを転送 (up channel) し，目的地が子，孫ノードの場合，そのノードの方向 (down channel) へパケットを転送する．

つまり，ルーティングは非最短型の deterministic routing であり，一部の実在するリンクを利用することができない．

パケットは必ず root node 方向 (up channel) に必要数移動した後，root node から遠ざかる方向 (down channel) に必要数移動するため，up channel と down channel の間に循環構造をとることがないのでデッドロックフリーが保証される．primitive up/down routing はデッドロックフリーなため，deadlock avoidance-based routing に分類される．

3.1.2 prefix routing

Wu と Sheng らにより提案された prefix routing [6] は primitive up/down routing で利用できなかった一部のリンクを利用できるようにしたもので，ルーティングアルゴリズムは，primitive up/down routing とほぼ同じである．

prefix routing ではツリー (spanning tree) を作成する際，各ノードとチャネルにラベルを付け，primitive up/down routing では無視したリンク (図 3 中の ignore channel) を cross channel としてツリーに組み込み，ルーティングに利用する．

ルーティングは，デスティネーションノードのラベルの prefix を持つチャネルがあればそれを選択し，無ければ，ラベル'e' のチャネル (単なる up channel) を選択し，root 方向へと向かう．これにより結果として，primitive up/down routing と同じ経路と，経路をショートカットすることが可能な cross channel を選択することが可能となっている．

例えば図 3においてノード **e** からノード **f** にパケットを転送する場合，primitive up/down routing では，**e**→**b**→**a**→**c**→**f** と 4 hop 必要だが，prefix routing では **e**→**c**→**f** と 2 hop で到着することができる．

prefix routing では prefix を調べるだけでルーティングができるので，up/down routing より小さなルーティングテーブルを実現可能であるという特徴も持つ．また，prefix routing は up channel と down channel の間に循環構造をとることがなく，また，up channel と down channel の 2 種類の経路で全てのパケットのルーティングを行うことができるためデッドロックフリーであり，deadlock avoidance-based routing に分類される．Autonet[1] で用いられている up/down routing も prefix routing に非常に近い．また prefix routing をベースに一工夫した手法もある [7].

3.1.3 minimal adaptive routing

primitive up/down routing と prefix routing は down channel を使用した後，up channel の使用を禁止することにより cyclic dependency を排除し，デッドロックフリーを実現したが，Silla らは既存のルーティングアルゴリズムを escape path に使い，各リンクに fully adaptive なバーチャルチャネルを追加する minimal adaptive routing を提案し [3]，[5]，cyclic dependency を除去することなしに，デッドロックフリーを実現した．以下にルーティングアルゴリズムを簡単に述べる．

まず，各リンク間に 2 本のバーチャルチャネル (original channel, new channel) を用意する．そして，ルーティングは原則として new channel を用いて行なう．new channel の使用条件は，目的地までの最短経路を取ることのみである．ただし，各ノードにおいて選択可能な new channel が全て使用されている場合，original channel に切り換えてルーティングを行う．original channel では既存のルーティングアルゴリズムを用いる．一度 original channel を使用したパケットは二度と new channel を使用することはできない．

original channel は，既存のルーティングアルゴリズムを用いているためデッドロックフリーが保証されており，ネットワーク全体に渡る escape path になるため，デッドロックフリーが保証される．

minimal adaptive routing はパケットが new channel を使用した場合最短経路をとるため効率が良く，既存の非最短型のルーティングアルゴリズムの性能を大きく向上することができる．

3.2 deadlock recovery routing

上記のルーティングアルゴリズムは deadlock の発生を防ぐことを前提として提案されており，deadlock avoidance-based routing と呼ばれる．一方，deadlock の発生を許し，deadlock が発生した場合にはそのパケットを結合網中から除去するといったアプローチをとるものもある [8]，[9]．このような deadlock recovery-based routing は各物理リンクに対し充分なバーチャルチャネル数を用意し，ネットワークに注入するパケット数を制御することによりパフォーマンスの向上を狙う．deadlock recovery-based routing は結合網のトポロジとパケット長，更に deadlock recovery の機構によって性能が大きく変化するため [10] 結合網に最適なバーチャルチャネル数を求めるのは非常に困難である．

3.3 既存のアルゴリズムのまとめ

一般的にルーティングアルゴリズムの自由度が高ければ高いほど，高いスループットを実現できる．しかし，既存のルーティングアルゴリズムは循環構造を作る要因についての対処が大がかりなものであり，デッドロックフリーの実現方法と自由度のトレードオフにおいて，デッドロックフリーの実現方法を優先したものである．

具体的には，primitive up/down routing，prefix routing ともノード間のチャネルを up channel，down channel の 2 つに分け，down channel を使用後，up channel を使用できないという厳しい制限を課している．また，minimal adaptive routing もこれら 2 つのルーティングアルゴリズムに併用する場合，virtual channel レベルでの自由度を高める限定的な効果しか期待できない．

一方，deadlock の発生を許す deadlock recovery-based routing ではルーティングアルゴリズムの自由度は高いが，deadlock の検出機構，フローコントロール，充分なバーチャルチャネル数等のネットワーク資源の確保など解決すべき問題が多い．

4 L-turn routing

deadlock recovery-based routing には解決すべき問題点が多く残されており，また，どのようなトポロジに対しても有効であることを保証するのは困難だと考えられるため，本研究では deadlock avoidance-based routing でのパフォーマンスの向上を目的とする．

既存の deadlock avoidance-based routing は自由度が低く，そのため root node にトラフィックが偏る，チャネルが空いていても最短経路を取りにくい，などの問題点がある．これは irregularity がデッドロックフリーを保証することを困難にしていることが最大の原因であるが，irregular network をツリーに見立てる際の工夫 (L-R tree) により自由度を飛躍的に上げることが可能である．

このツリー (L-R tree) を用いて我々は論理的なパケットのターン方向を管理し，cyclic dependency を除去する Left-first turn routing(L-turn routing) を提案する．また，L-turn routing はバーチャルチャネルを追加して minimal adaptive routing と併用することにより更なる性能向上も果たすことができる．

以降，まず irregular network をツリー (L-R tree) に見立てる手順を説明し，その後 L-turn routing の提案を行う．

4.1 L-R tree の作成

prefix routing 等で述べたツリーでは root node に向かうチャネルを up channel，root から遠ざかるチャネルを down channel と分類した．そして，prefix routing と primitive up/down routing は up/down という階層構造を基にデッドロックフリーを実現した．

しかし，我々は自由度の高いルーティングを実現するため発想を逆転し，up/down の概念を捨て，left/right という概念を導入したツリーである L-R tree を提案する (図 4)．ここで以下のように語を定義する．

- **left node:** L-R tree において，各階層の一番左側のノード．図 4においては **a,b,d** がこれにあたる．
- **right node:** L-R tree において，left node 以外のノードのこと．図 4においては **c,e,f,g,h,i** がこれにあたる．

以下に，L-R tree の構成法を述べる．

1. ネットワークの中から任意に L-R tree の root node を選ぶ．
2. root node に隣接したノードの一つを left node，他を right node として L-R tree に追加する．例えば図 4では root node は **a**, left node は **b**, right node は **c** である．
3. L-R tree の left node，right node について以下の作業を行う
   (a) right node において，まだ L-R tree に組み込まれていない隣接ノードを right node として L-R tree に追加する．

(b) left node において，まだ L-R tree に組み込まれていない隣接ノードのうち，1 つを left node，他を right node として L-R tree に追加する．例えば図 4ではノード **b** の隣接ノードのうち，left node は **d**, right node は **e,f** である．

4. 3. の手順を全ノードが L-R tree に組み込まれるまで繰り返す．
5. L-R tree に漏れている irregular network のリンクを追加する．

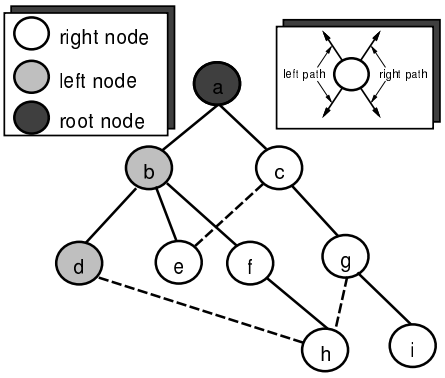

図 4: L-R tree の例 (ノード数 9)

図 4における破線のリンクは，L-R tree の構成法の手順 5. で追加したリンクである．

4.2 L-turn routing algorithm

本節では前節で述べた L-R tree を利用する Left-first turn routing (L-turn routing) の提案を行なう．

第 3 章で説明した既存のルーティングアルゴリズムは，ツリーの階層間の移動に厳しい制約があり，自由度が低くなることが問題であるが，L-turn routing はツリーの階層間の移動の制限を無くした点が特徴である．

まず，以下のように語を定義する．

- **left path:** L-R tree において，現在のノードよりも左側に存在するノードへのリンクのこと．図 4において，ノード **e** からノード **b** へのリンクは left path である．
- **right path:** L-R tree において，現在のノードよりも右側に存在するノードへのリンクのこと．図 4において，ノード **e** からノード **c** へのリンクは right path である．

すると L-turn routing は以下のような制限で表される．

right path を使用した後，left path を使用してはならない

L-turn routing は上記の条件を守れば，いかなる経路も選択可能であり，自由度の高い非最短型ルーティングを実現する．

定理 1： L-turn routing はデッドロックフリーである

証明： L-R tree において right 方向へ進んでいるパケット (つまり right path を使用) は，left 方向へ方向転換することが出来ないため，目的地に到着するまで right path しか使えない．よって right path は循環することがないため，デッドロックフリーである．

また，left 方向へ進んでいるパケット (つまり left path を使用) は一度だけ right 方向へ方向転換できるが，right 方向のチャネル (right path) はデッドロックフリーであるため left 方向のチャネルもまた，デッドロックフリーである．

よって，L-turn routing は，left 方向のチャネル (left path) と right 方向のチャネル (right path) で構成されたネットワーク上ではデッドロックフリーである．

L-turn routing は right path と left path 間の循環構造を論理的に断つことにより，リンク間の双方向チャネル 1 本で adaptive routing を実現している．また，L-turn routing は任意の spanning tree 上では全ノードへの経路が保証されないが，L-R tree 上での L-turn routing では，spanning tree の各階層において left node を一つのみ選択し，残りすべてを right node とするという構成を取ることにより，

(a) left node からは right path のみを進み続けることにより必ず root node に到達できる．
(b) left node の子孫である right node からは，left path のみを進み続けることにより left node に到達できるので (a) より，必ず root node に到達できる．
(c) それ以外の right node からは，left path のみを進み続けることによ root node に必ず到達することができる．

ことが保証される．これは primitive up/down, prefix routing の up 方向のエミュレートが可能であることを示している．また，root node に到達した後には，上記の L-tree の構成により，

- left node もしくは left node の子孫である right node から root node に到達した場合には，right path のみを進み続けるだけで，必ず目的ノードに到達できる．
- その他の right node から root node に到達した場合には，right path → left path の順にチャネルを使用しなければ到達できない目的ノードは絶対に無いため，必ず目的ノードに到達することができる．

ことも保証される (down 方向移動のエミュレート)．更に，prefix routing におけるショートカットを行なう場合には，right path → left path の順にチャネルを使用することは無いので，ショー

トカットを必ず使うことができることも保証される．よって，L-R tree 上での L-turn routing では primitive up/down，prefix routing をエミュレートすることが可能である．

L-turn routing は up channel，down channel の概念がないため，図 4においてノード **d** からノード **i** へパケットを転送するような場合，prefix routing や primitive up/down routing では **b**→**a**→**c**→**g**→**i** と 5 hop 必要だが，L-turn routing では **h**→**g**→**i** と 3 hop で到着することができる．更に，L-turn routing は primitive up/down routing や prefix routing で用いた経路 (**b**→**a**→**c**→**g**→**i**) をも先に述べた制約を満たすため選択可能である．これは，L-turn routing は spanning tree の作成に工夫をしたことに因る．

L-turn routing は非最短型の adaptive routing であるが，自由度が高いため全体のチャネル利用率が向上し，traffic が分散される．また，パケットの平均 hop 数を抑えることもできる．

また，L-turn routing は自由度が高いため途中経路において複数のチャネルを選択可能になることが多く，出力チャネルの選択アルゴリズムである output selection function を慎重に決める必要がある [11]．irregular network の場合，理想の output selection function とは全てのパケットが最短経路を取り，なおかつ全てのノードのチャネル利用率を均一にすることである．この条件が満足されれば結合網のチャネルを有効に利用でき結合網の性能を最大限に引き出すことができる．

5 評価

提案した L-turn routing，そして既存の prefix routing，primitive up/down routing，minimal adaptive routing，deadlock recover-based routing の各ルーティングアルゴリズムを C++ で記述したフリットレベルシミュレータを用いて評価を行った．シミュレーションに用いた条件を表 1に示す．

表 1: シミュレーション条件

パケット長	128 flit
実行時間	50,000 clk
バーチャルチャネル数	1 2 (minimal adaptive 併用) 3 (deadlock recovery のみ)
ネットワークサイズ	9 or 16 nodes
トラフィックパターン	uniform

シミュレーションで初めの 5,000 クロックはネットワークが安定せず，想定した負荷に達していないと考えられるため無視して評価を行った．1

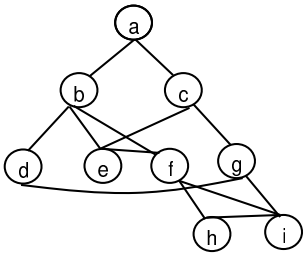

図 5: 9 nodes irregular network

ノードにつき 1 プロセッシングエレメントと仮定した．ネットワークのトポロジについては，irregular network での性能評価としてランダムに 9 ノードを接続したもの (図 5) と 4×4 2D torus を用いた (図 6)．バーチャルチャネルの本数は用いるルーティングアルゴリズムによって異なる．トラフィックパターンに用いた uniform traffic では各パケットの目的地ノードはランダムに決定されており，等確率に分散されている．今回の評価で用いた deadlock recovery-based routing はデッドロックが起きない場合にネットワークの性能を最大限引き出すことができる．よって，本稿ではネットワーク自身が持つポテンシャル性能と deadlock avoidance-based routing の比較を行うためにデッドロック検出機構を設けていない．結合網の直径の数だけバーチャルチャネルを用意すればデッドロックは発生しないが，今回の評価では他のルーティングアルゴリズムとの比較のために 3 本のバーチャルチャネルを用意して評価を行った．deadlock avoidance-based routing (up/down, prefix, L-turn) はそのルーティングアルゴリズム単体での評価 (バーチャルチャネル 1 本) と minimal adaptive routing を併用したとき (バーチャルチャネル 2 本) の評価を行った．

irregular network では各パケットがネットワークにかける負担を抑えることが性能向上につながる [3]．よって L-turn routing の選択可能な経路の中から最短のものを選択するようにした．複数の最短経路が選択可能な場合，今回の評価では単純にある決められた優先順位により選択することにした．

5.1 9 nodes irregular network

9 nodes irregular network での評価を図 7，図 8に示す．図 7はリンク間の双方向チャネルが 1 本の時，図 8はリンク間の双方向チャネルが 2 本で，各アルゴリズムに minimal adaptive routing を併用した時のものである．また，横軸はスループット，縦軸はレイテンシを表している．

図 7より，ルーティングアルゴリズムの自由度がパフォーマンスに大きく影響を与えていることがわかる．L-turn routing はその高い自由度により他の

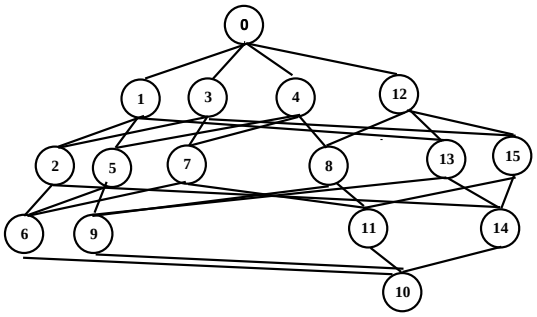

図 6: 16 nodes 2D torus

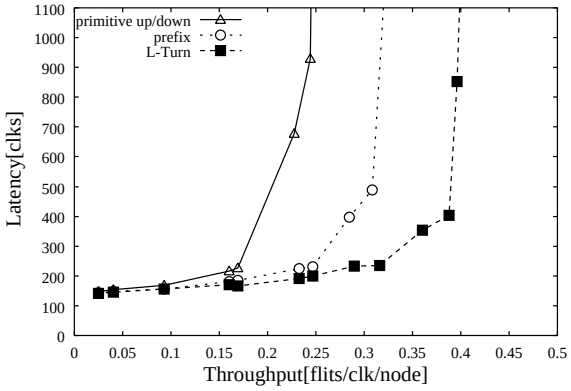

図 7: 9 node, Vch 1 本

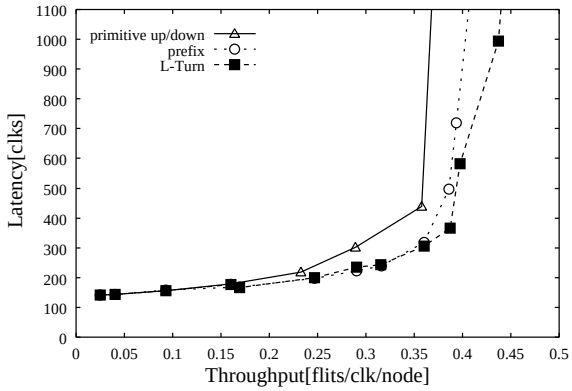

図 8: 9 node, Vch 2 本 (minimal adaptive routing 併用)

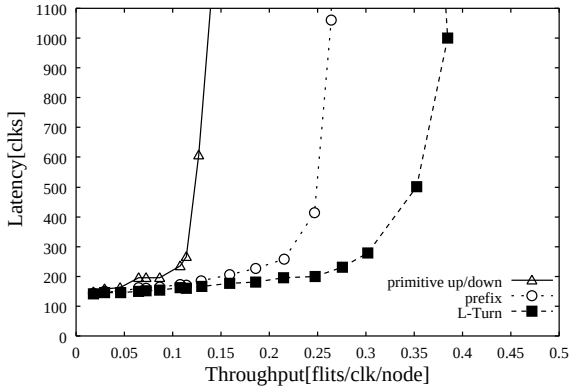

図 9: 16 node, Vch 1 本

2 つの deadlock avoidance-based routing に比べ高い性能を示した．また，deadlock recovery-based routing はバーチャルチャネルの本数を 3 本用意しても deadlock が発生してしまい，9node のトポロジでは評価を取ることができなかった．deadlock recovery-based routing はトポロジ，バーチャルチャネル数に大きく依存するので irregualr network 上では性能が安定しない．図 8より minimal adaptive routing を併用した場合，各ルーティングアルゴリズムの性能差が縮まっていることがわかる．これは多くのパケットが minimal adaptive routing を行うチャネル (new channel) のみを利用し，original channel 上で各ルーティングアルゴリズムを利用するパケットが減少しているからである．しかし minimal adaptive routing を併用した場合にも original channel 上で最も高い自由度を持つ L-turn routing が一番高い性能を示した．

5.2 16 nodes 2D torus

4 × 4 2D torus での評価を図 9，図 10に示す．

図 9はリンク間の双方向チャネルが 1 本の時，図 10はリンク間の双方向チャネルが 2 本で，minimal adaptive routing を各アルゴリズムと併用した時のものである．なお 9 node irregular network と同様，deadlock recovery-based routing のみバーチャルチャネルを 3 本用意している．

図 9より，ノード数が 16 の場合でも L-turn が高い性能を示していることがわかる．また，図 7 と比較すると，ノード数が増加したときに各 deadlock avoidance-based routing の差は顕著となる．これはノード数が増加したことと，規則的な結合網を用いることにより各ノード間のリンクの本数が増加したことによって他のルーティングアルゴリズムより自由度の高い L-turn routing における経路の選択肢が著しく増加したからである．このことより，L-turn routing は代替経路が多く利用可能な結合網において有利であることが言える．また，4×4 2D torus の場合には deadlock recovery-based routing はバーチャルチャネルを 3 本用意すれば deadlock が発生しなかった．また，L-turn routing は deadlock avoidance-based routing にもかかわらず，deadlock recovery-based routing と同等の性能を示していることがわかる．

また，平均 hop 数の評価を表 2に示す．表 2に示す通り，deadlock avoidance-based routing の中で

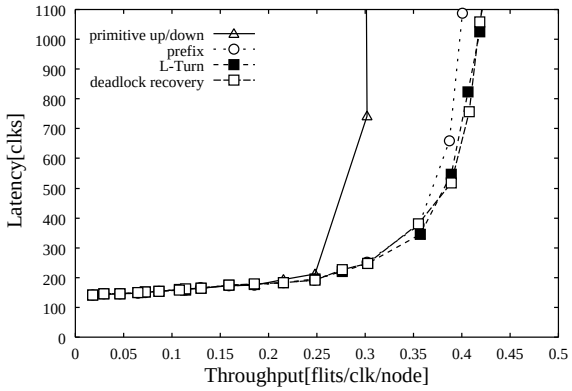

図 10: 16 node, Vch 2 本 (minimal adaptive routing 併用)

表 2: 平均 hop 数

9 nodes	Vch × 1	Vch × 2
up/down	2.73	1.84
prefix	2.03	1.81
L-turn	1.89	1.80
16 nodes	Vch × 1	Vch × 2
up/down	3.33	2.26
prefix	2.50	2.17
L-turn	2.12	2.11
recovery-based	2.11	2.11

は L-turn の平均 hop 数が最も少ない．deadlock recovery-based routing では new channel のみを用いているため最短経路を常に選択し，その結果平均 hop 数を抑えているが，L-turn は deadlock free を保証しつつ，deadlock recovery-based routing に迫る平均 hop 数を達成している．バーチャルチャネルの本数が 2 本になると，どのルーティングアルゴリズムでも最短経路を選択可能な new channel を使用するため平均 hop 数の差が小さくなるが，original channel の平均 hop 数の差がパフォーマンスに影響を及ぼしていることがわかる．

6 まとめ

irregular network において自由度の高い adaptive routing である L-turn routing，及び L-turn routing に必要な L-R tree の提案を行った．

L-turn routing は irregular network を L-R tree に見立て，パケットの論理的なターン方向を管理することにより全ノードへの経路を保証し，かつデッドロックフリーを実現した．L-turn routing は既存の deadlock avoidance-based routing と異なり，ツリーの葉方向へ移動した後に root node の方向へ移動する事ができるため高い自由度を持つ．

今後は L-turn routing の自由度を十分に生かせるような L-R tree の構成法，及び L-turn routing の出力チャネルの選択法 (output selection function) を検討し，更なる性能向上をはかる予定である．また今回は 16 node までの評価を行ったが，より多いノード数，多様なトポロジでの評価も行う予定である．

参考文献

[1] Schroeder et al. Autonet: A high-speed, self-configuring local area network using point-to-point links. *Technical Report SRC research report 59,DEC*, Apr. 1990.

[2] R. W. Horst. Tnet: A reliable system area network. In *IEEE Micro*, Feb. 1995.

[3] F.Silla and J.Duato. Efficient Adaptive Routing in Networks of Workstations with Irregular Topology. In *proc. of CANPC'97*, Feb. 1997.

[4] 西 宏章, 多昌 鷹治, 工藤知宏, and 天野英晴. 仮想チャネルキャッシュを持つネットワークルータの構成と性能. In *JSPP'99* 論文集, 1999.

[5] F.Silla and J.Duato. Is it worth the flexibility provided by irregular topologies in networks of workstations? In *proc. of CANPC'99*, Jan. 1999.

[6] Jie Wu and Li Sheng. Deadlock-Free Routing in Irregular Networks Using Prefix Routing. *DIMACS Technical Report 99-19*, Apr. 1999.

[7] A. Robles J. C. Carlos and J. Duato. flexble routing scheme for networks of workstations. In *proc. of ISHPC 2000*, Oct. 2000.

[8] S. Warnakulasuriya and T. M. Pinkston. Characterization of Deadlocks in Irregular Networks. In *Proc. of ICPP'99*, Sep. 1999.

[9] T. M. Pinkston and S. Warnakulasuriya. On deadlocks in interconnection networks. In *Proc. of ISCA'97*, Jun. 1997.

[10] J. M. Martinez P. Lopez and J. Duato. A very efficient distributed deadlock detection mechanism for wormhole networks. In *Proc. of the HPCA'98*, Feb. 1998.

[11] 鯉渕 道紘, 舟橋 啓, 上樂 明也, and 天野英晴. 適応型ルーティングにおける output selection function. In *JSPP2000* 論文集, May 2000.