社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

複数ホストリンクを用いたNoC向け低遅延トポロジ

河野　隆太† 　藤原　一毅†† 　松谷　宏紀† 　天野　英晴† 　鯉渕　道紘††

† 慶應義塾大学大学院 理工学研究科　223–8522 神奈川県横浜市港北区日吉 3–14–1
†† 国立情報学研究所　101–8430 東京都千代田区一ツ橋 2–1–2
E-mail: †{kawano,hunga}@am.ics.keio.ac.jp, ††{ikki,koibuchi}@nii.ac.jp, †††matutani@arc.ics.keio.ac.jp

あらまし　近年のメニーコア・アーキテクチャでは，コアの数は増加の一途を辿っており，コア間の通信遅延がアプリケーションに与える影響が益々大きくなってきている．コア間の通信にはパケット・ネットワーク構造（Network-on-Chip，NoC）が広く用いられるため，コア間トポロジが通信遅延に大きく影響する。そこで、本研究報告では、end-to-end 通信遅延を削減するために、規則的なルータ間トポロジに対し、複数リンクを単一コアとランダムに選択した複数ルータに接続する方法を提案する。フリットレベルのネットワークシミュレーションの結果，ランダムコアリンクを用いた我々のトポロジは，従来のトポロジに比べ，平均遅延を最大 27%減少させた．
キーワード　チップ内ネットワーク, トポロジ，相互結合網

A low latency topology for NoC using multiple host links

Ryuta KAWANO†, Ikki FUJIWARA††, Hiroki MATSUTANI†, Hideharu AMANO†, and Michihiro KOIBUCHI††

† Graduate School of Science and Technology, Keio University　Hiyoshi 3–14–1, Kohoku-ku, Yokohama, Kanagawa, 223–8522 Japan
†† National Institute of Informatics　Hitotsubashi 2–1–2, Chiyoda-ku, Tokyo, 101–8430 Japan
E-mail: †{kawano,hunga}@am.ics.keio.ac.jp, ††{ikki,koibuchi}@nii.ac.jp, †††matutani@arc.ics.keio.ac.jp

Abstract　In recent many-core architectures, the number of cores has been steadily increasing. Therefore, network latency between cores has become a more important issue for applications. Because packet network structures (Network-on-Chip, NoC) are widely used for core-to-core communications, a topology among cores has a major impact on network latency. Therefore, in this research, to reduce end-to-end communication latency, we propose a method to build network topologies by adding multiple links between a single core and randomly selected multiple routers on a regular topology of routers. Results obtained with flit-level discrete event simulation show that our random-core-link topologies achieved the average latency up to 27% lower than that of baseline topologies.
Key words　Network-on-Chip (NoC), topology, interconnection networks

1. はじめに

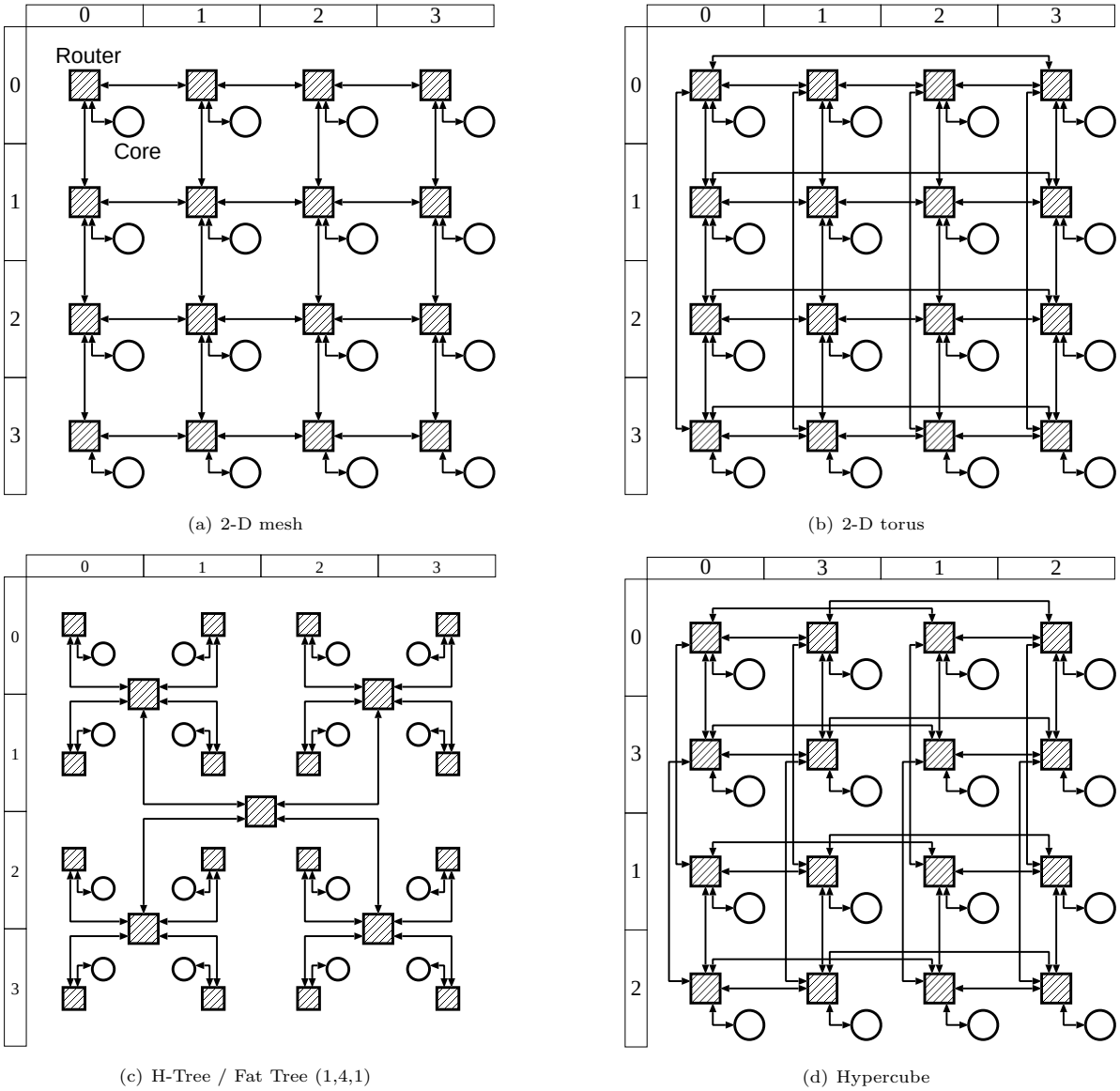
近年のメニーコア・計算機アーキテクチャでは，コア数が増加の一途を辿っている。そのため、コア間の通信遅延がアプリケーションに与える影響が益々大きくなってきている．コア間の通信にはパケット転送を用いたネットワーク構造（Network-on-Chip，NoC）[1] が広く用いられるため，コア間ネットワークトポロジが通信遅延に大きく影響する。

従来、チップ内ネットワークのトポロジの研究はルータを頂点としたグラフにモデル化し、直径および平均距離などの指標に基づいて最適化し、生成されることが多かった。しかし、この生成グラフに基づくネットワークトポロジは、すべての end-to-end 通信遅延に対して、最初と最後の 1hop、すなわちコアールータ間の遅延が別途加算されることになる。

そこで、我々は、既存のルータ間トポロジに、コアから直接複数の (ショートカット) リンクを（ランダムに選択した）異なるルータに接続するトポロジを提案する。最近の我々の成果から、相互結合網においてランダムにコアールータ間を接続することでトポロジの平均距離、直径ともに劇的に改善されることが分かっている [2]。そこで、我々はこれらの成果をふまえ、チップ内ネットワーク向けに、配線密度、および各配線長を一定以下に抑える制限を課したランダムトポロジを追求する。ただし、上記の研究 [2] や特殊なトポロジ [3] と異なり、コアの軽量化のため、各コアは中間ノードとして (自身へ、あるいは自身からのパケット以外の) パケット処理、つまりルーティングは行わないこととする。

本論文で得られた知見は以下である。

- ランダムに選択したルータとコア間リンク（以後ランダムコアリンクと呼ぶ）をコア当たり 3 本追加することで、64 ルータネットワークにおいて 37 %の最悪通信遅延と 51 %の平均通信遅延の削減が達成できることが分かった。このランダムコアリンクはマンハッタン距離で 6 コア長以内という制約を課しても、この遅延削減効果は得られた。
- ランダムコアリンクは、ルータ間トポロジとして、2 次



(a) 2-D mesh

(b) 2-D torus

(c) H-Tree / Fat Tree (1,4,1)

(d) Hypercube

図 1　典型的な接続網の 2 次元レイアウト (16 コア)

元メッシュのみならず、2 次元トーラス、次数 5 のツリーに対して適用した場合にも通信遅延の削減効果が高いことが分かった。

- ネットワークシミュレーションの結果、従来のトポロジと比べ，遅延を最大 27%減少させた.
- ランダムコアリンクを用いたトポロジは，同程度の総配線長で構成されるルータ間にリンクを付加したトポロジに比べ，総配線長に対する遅延を大きく削減できた.

以後、2. 章において関連研究を述べ，3. 章において，ランダムコアリンクを用いたトポロジの提案を行う．4. 章において、ランダムコアリンクを用いたトポロジの end-to-end 通信遅延の解析、 5. 章においてネットワークシミュレーションによる遅延およびスループットの測定を行う．6. 章ではルータ間にランダムなリンクを付加したトポロジとの比較を行い，最後に 7. 章においてまとめと今後の課題を述べる。

2. 関 連 研 究

2.1 低遅延通信技術

チップ内ネットワークにおいて、これまで 1 サイクル、あるいは 2 サイクルでパケットを転送可能な低遅延ルータ [4]、トラヒックパターン、負荷に応じて混雑を避ける経路を動的に選択する適応型ルーティング、ワームホールスイッチングの利用など、種々の低遅延通信技術が設計されてきた。特にワームホールスイッチングの利用により、長いパケットの end-to-end 通信遅延に対する経由 (ルータ) ホップ数が与える影響を抑えることができるようになった。

しかし、パケットサイズは通常、極めて短い場合が多い。例えば、TRIPS では，On-Chip Network (OCN) におけるトラフィックはメモリ転送が多く，キャッシュの line size である 64-Byte 転送は 5-flit パケットに，データ転送要求などの細かい通信は 1-flit パケットとして転送する．さらに，Operand Network (OPN) には演算データが流れるが，90% のパケットは 99-bit 以下であり，1-flit パケットに収まる [5] [6]．このような場合、通信遅延を削減するためには、トポロジ自体の改良が重要となる。

2.2 NoC トポロジとレイアウト

直径、平均距離の小さなトポロジとそのレイアウトについて

は、様々な提案が成されている。図 1 に典型的なトポロジを示す。各丸頂点はコア、黒四角頂点はルータを示す。4. 章のグラフ解析では，ルータ間トポロジに 2-D Mesh (図 1(a)), 2-D Torus (図 1(b)), H-Tree (図 1(c)), Hypercube (図 1(d)) を採用し，5. 章のネットワークシミュレーションでは，ルータ間トポロジに 2-D Mesh を採用した．

H-Tree はツリー構造をベースとするもっとも単純なトポロジであり，最上位のルータを除く全てのルータが，上位のルータ向けのリンクを 1 本持ち，下位のルータ向けのリンクを 4 本持つ．ツリーのルート付近のルータは常に混雑しやすいため，二分帯域幅が小さい特徴を持つ．

これらの規則的なトポロジに加えて、ランダムトポロジについても議論がなされている。ランダムグラフを用いたネットワークが，そのスモールワールド性により，従来の規則的なトポロジに比べて直径や平均最短距離を小さくできることが報告されている．このようなネットワークは遅延性能に対して配線長が大きくなる傾向があるため，配線長を抑えるようなレイアウトの工夫があるが [7]、十分な性能が期待される [8].

3. ランダムコアリンク・トポロジ

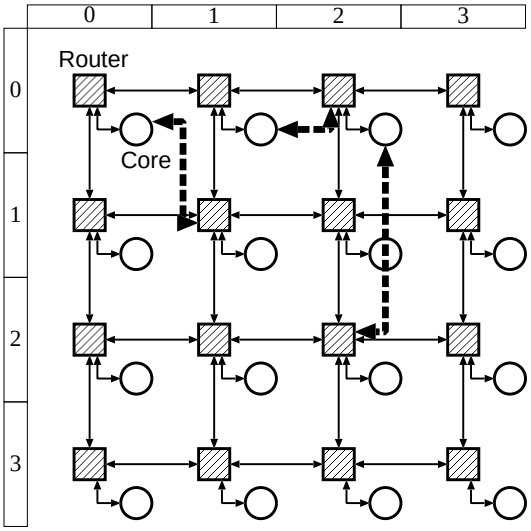


図 2　ランダムコアリンクの接続 (4x4 Mesh)

本論文で提案するトポロジにおけるランダムコアリンクの接続を図 2 に示す．この図においてランダムコアリンクを点線で示しており，通常のルータ間トポロジに対し，各コアからランダムに選択したルータに対し追加リンクを接続する．

コア当たりの追加リンクの本数 (x) は各コア間で等しく，コアからのリンクは互いに異なるルータに繋がるものとする．また，コア・ルータ間のリンク追加に伴うルータの増加ポート数は各ルータ間で等しいこととする．

さらに，追加リンクの長さを制限することによる性能の変化を調べるため，トポロジ内のランダムコアリンクの長さの上限である接続半径 (y) を設定した．ここで，y は，各コアのローカルルータ (ランダムリンクを用いない場合に各コアが接続しているルータ) と，そのコアが接続可能なルータとの間のマンハッタン距離 (コア長) の上限を表す．

4. 解　　析

本章ではグラフ解析を用いることにより，従来のトポロジと，複数ランダムコアリンクを用いたトポロジの評価を行う．具体的には，経由ルータ数及び経路上の配線距離を考慮にいれたコア間の平均 Zero-load 遅延の評価を行う．

今後，評価対象のトポロジは”TOPOLOGY-x-y” と表記する．この表記は，TOPOLOGY がルータ間トポロジ，x がコアあたりのランダムリンクの本数，y がランダムコアリンクの長さの上限 (コア長) を表す．

$x = 0$ の時，存在するコア・ルータ間のリンクは，同じ 2 次元座標に位置するコア・ルータ間のローカルなリンクのみである．

そして，本章では，以下に示すランダムコアリンクを用いたトポロジについて，グラフ解析を用いた遅延評価を行う．

- 2D-MESH-x-y: ルータ間のトポロジは Mesh であり，各コアは x 本のリンクをマンハッタン距離が y(コア長) 以内のルータへ接続する．
- 2D-TORUS-x-y: ルータ間のトポロジは Torus であり，他の条件は 2D-MESH-x-y と同様である．
- H-TREE-x-y: ルータ間のトポロジは H-Tree であり，他の条件は 2D-MESH-x-y と同様である．
- HYPERCUBE-x-y: ルータ間のトポロジは Hypercube であり，他の条件は 2D-MESH-x-y と同様である．

各トポロジのレイアウトについては、直感的な方法 (図 1) に基づいて行う。2D-MESH のルータ間リンク長は全て 1 コア長となり，その他のトポロジでは，コア長が 1 を上回るような長距離リンクが存在する．

コア間のフリット通信において，コアからローカルルータ，ローカルルータからコアへの転送遅延を 1[cycle], ルータを通過する遅延を 2[cycles], そしてルータ間の配線での転送遅延を 1 コア長あたり 1[cycle / コア長] とした．

先述の通り，各コアでのパケットフォワーディングは行わないものとした．

また，ルータ間トポロジが H-TREE の場合は，コアからのランダムリンクの接続先として，ハブとなっているルータは選ばないこととした．

4.1　ランダムコアリンク数

はじめに，各コアにおけるランダムリンク数 (x) を増加させた場合の平均 Zero-load 遅延の変化について解析する．

図 3-6 は 16, 64 コアの NoC トポロジにおける 1 コアあたりのランダムリンク数と最大・平均 Zero-load 遅延の関係を示している．ここで，ランダムリンクの接続先として許されるローカルコアからのコア距離 (y) を，16 コアでは 2, 64 コアでは 4 とした．また，各プロット値では，乱数を変えて 10 個のトポロジを生成し，それぞれの計測値の平均値と標準偏差をとっている．

これらの図より，最大 Zero-load 遅延についてはルータ間が H-TREE の場合を除く全ての場合で遅延を削減でき，平均 Zero-load 遅延については，全てのルータ間トポロジにおいて遅延削減の効果が得られることが分かった．特に 2DMESH の

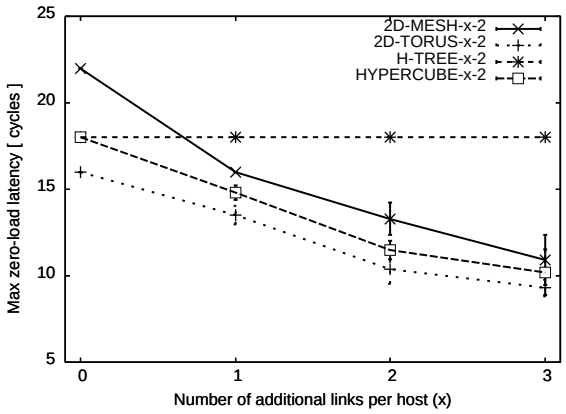


図 3　16 コア, $y = 2$ における各トポロジのランダムコアリンク数と最大 Zero-load 遅延

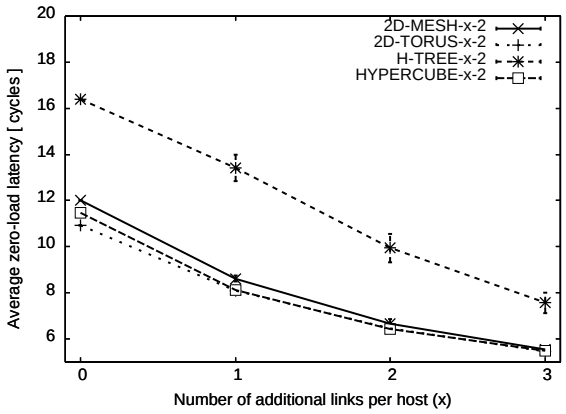


図 4　16 コア, $y = 2$ における各トポロジのランダムコアリンク数と平均 Zero-load 遅延

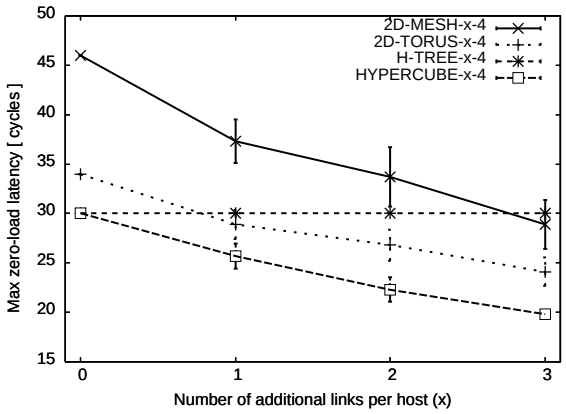


図 5　64 コア, $y = 4$ における各トポロジのランダムコアリンク数と最大 Zero-load 遅延

場合は，64 コアのトポロジにおいて 1 コアあたりランダムリンクを 3 本付加することにより，ランダムリンクを用いる前のトポロジと比べて最大，平均の Zero-load 遅延をそれぞれ 37%, 51%削減可能であることが分かった．

また，H-TREE が最大 Zero-load 遅延を減らすことができないのは，ランダムリンクの接続先であるルータが，HTREE の階層構造の末端に位置するため，リンク付加による遅延削減の効果が局所的となってしまうことが原因と考えられる．

4.2　ランダムリンクの接続半径

ここでは，各コアにおけるランダムリンクのコア距離での接続半径 (y) を増加させたときの遅延の変化について解析する．予測されうることとして，接続半径が増えるほど接続先の候補

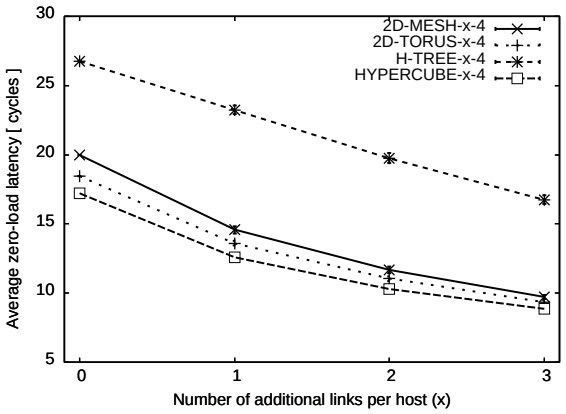


図 6　64 コア, $y = 4$ における各トポロジのランダムコアリンク数と平均 Zero-load 遅延

となるルータ数が増えるため，コア間の遅延が小さくなることが挙げられる．

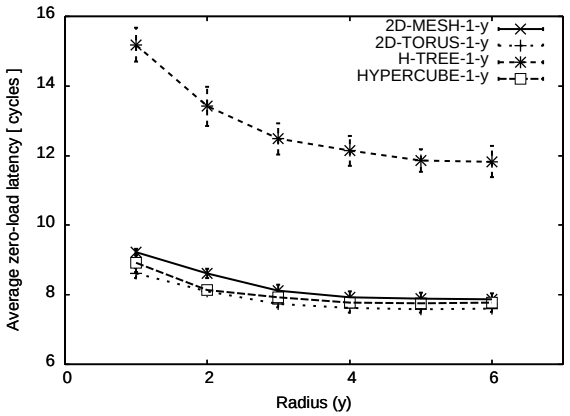


図 7　16 コア, $x = 1$ における各トポロジのランダムコアリンク数と平均 Zero-load 遅延

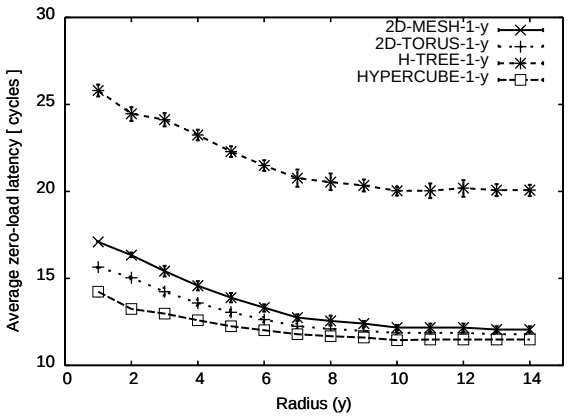


図 8　64 コア, $x = 1$ における各トポロジのランダムコアリンク数と平均 Zero-load 遅延

図 7, 8 は，16 コア及び 64 コアの NoC トポロジにおけるランダムリンクの接続可能半径と平均 Zero-load 遅延の関係を示している．4.1 章と同様に，各プロット値は乱数の異なる 10 個のトポロジについて，平均値と標準偏差を求めている．ここで，追加ランダムリンク数 (x) を 1 とした．4x4, 8x8 のコア配列における最長のコア間距離はそれぞれ 6, 14 であるため，4x4 のコア配列では $y = 6$ の時，8x8 のコア配列では $y = 14$ の時，完全にランダムなコアリンクの接続が実現されている．

これらの図において，半径の増加に伴い遅延は単調に減少し

ているが，接続可能半径の増加に伴い，遅延削減の効果は次第に穏やかになっている．例えば，ルータ間が Mesh の場合，ランダムコアリンクを使わない場合と比べた平均遅延削減の効果は次のとおりである．まず，16 コアの場合の削減率は，$y = 2$ で 28%，$y = 3$ で 32% であるのに対し，完全にランダムなコアリンクの接続が実現されている $y = 6$ の時，34% にとどまる．また，64 コアの場合，$y = 4$ で 27%，$y = 6$ で 33% であるのに対し，完全にランダム接続である $y = 14$ で 40% にとどまる．

長距離の配線はコスト・電力の面から不利であるため，接続半径をある程度小さくする設計が，性能対コストの面で有利であると結論づけられる．

5. ネットワークシミュレーション

本章では，ランダムコアリンクのネットワーク性能への影響を調べるため，遅延を評価した．

5.1 シミュレーション環境

複数ランダムコアリンクを用いたトポロジの性能評価のために、マルチコアプロセッサのフルシステムシミュレータである GEM5 [9] を使用した。GEM5 は、C++ 及び Python で実装されており、プロセッサの詳細なシミュレーションに加え、NoC を対象にしたネットワークシミュレーション機能も有している。ネットワークトポロジの生成部分は Python で記述されている。本研究では疑似乱数を生成することで、複数のコアリンクをランダムに生成した。

フリットがルータを通過する遅延は最低 2[cycles] とした．また，リンク遅延は，リンク長に関わらず 1[cycles] とした．各コアは独立してネットワークにフリットを注入するものとし，フリットサイズは 128 ビットとした．

スイッチング方式としてバーチャルカットスルーを用いることとし，全てのトポロジにおいて仮想チャネルは 4 本とした．また，ルータ間のトポロジには Mesh を採用し，ルータ間のルーティングには XY ルーティングを用いた．

ランダムコアリンクを用いたトポロジにおいては，経路として用いる送信コアおよび受信コアのリンクを選択する際は，最短経路になるようなリンクを選ぶこととした．また，先述のとおり，コア上でのルーティングは禁止した．

5.2 ネットワーク性能の評価

図 9-12 は，ランダムに宛先を選択する Uniform トラフィックと，合成トラフィックである Bit complement トラフィックを，コア数 16, 64 のネットワークに注入した場合のシミュレーション結果である．ここで，接続半径は $y = 2$(コア数 16), 4(コア数 64) とし，追加リンク数 $x = 0, 1, 3$ の 3 つについて評価を行った．縦軸はフリットが生成されてから宛先コアに到達するまでのコア間の平均遅延を, 横軸は各コアの送信フリットレートである Injection rate を示す．

これらの図から，2D-MESH-x-y はコアリンク数の増加に伴い，低負荷時の遅延が削減できる．特に，コア数が 64，$y = 4$ の場合，ランダムコアリンクを 3 本付加することにより，Uniform Traffic では 27%，Bit complement Traffic では 26%の遅延削減効果が得られた．

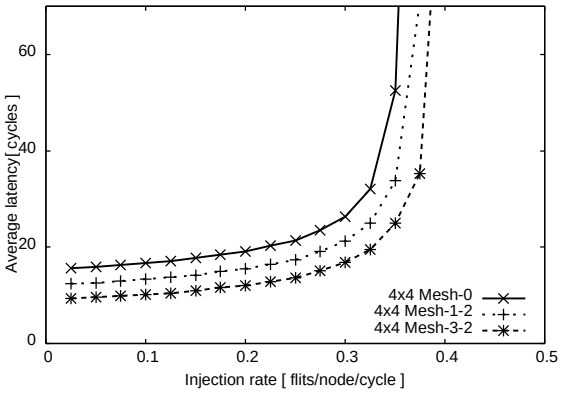


図 9　16 コアにおけるネットワーク性能 (16 コア，Uniform トラフィック)

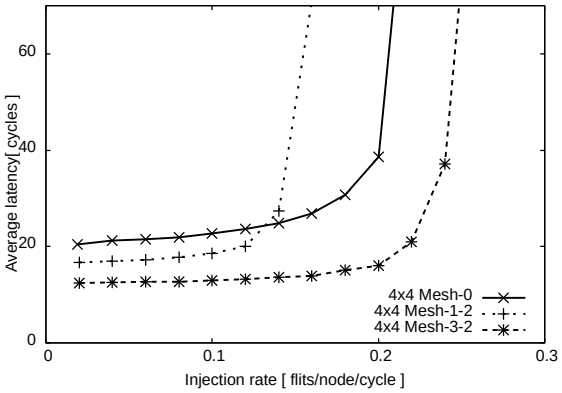


図 10　16 コアにおけるネットワーク性能 (16 コア，Bit complement トラフィック)

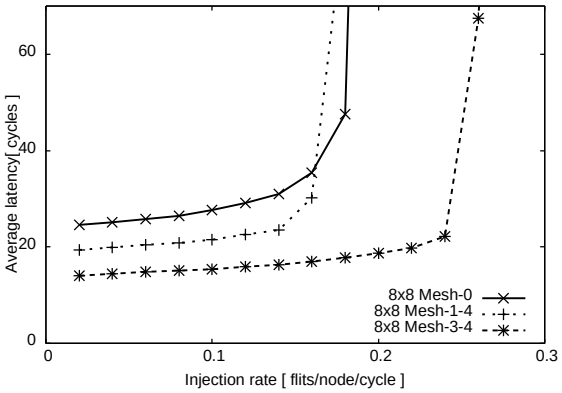


図 11　64 コアにおけるネットワーク性能 (64 コア，Uniform トラフィック)

しかし，図 10，図 12 に示すように，特に Bit complement Traffic において，ランダムリンクを付加することによりスループットが減少している．これは，今回の評価におけるルーティングが 1 つの最短経路に転送経路を限定するようなルーティングであるため，リンク付加によりルータ間トポロジにおけるロードバランシングが悪化していることが原因と考えられる．

このような問題を解決する手法として，複数の最短経路及び非最短経路を許すようなルーティングを実装し，ロードバランシングを図ることが考えられる．このようなルーティング手法の実装および評価は今後の課題である．

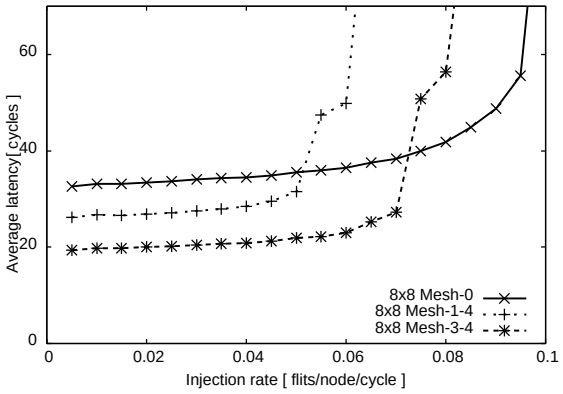


図 12　64 コアにおけるネットワーク性能 (64 コア，Bit complement トラフィック)

6. ルータ間ランダムトポロジとの性能・総配線長の比較

我々は従来の研究において，ルータ間のトポロジを最適化することよりも，コアから複数ルータへのリンク接続を増やすことによる遅延削減効果が大きいことを示している [2]．このことが NoC 向けトポロジにおいても同様であることを示すため，本章ではルータ間にランダムトポロジを適用した場合との遅延・総配線長の比較を行う．

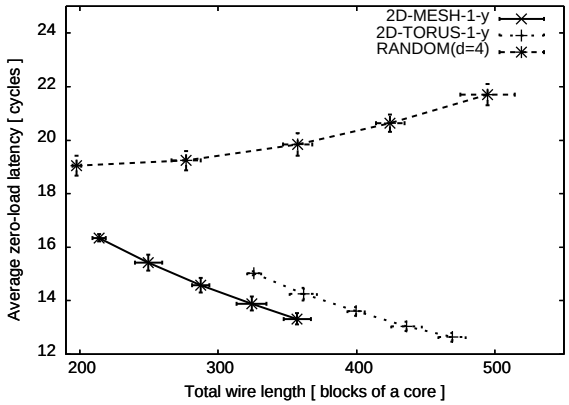


図 13　64 コアにおける各トポロジの総配線長と平均 Zero-load 遅延

図 13 は，ランダムコアリンクを用いたトポロジとルータ間ランダムトポロジの比較を示す．横軸はトポロジの総配線長をコア長で表したものであり，縦軸は平均 Zero-load 遅延を示す．

ランダムコアリンクを用いたトポロジにおいて，コア間トポロジとして 2D-MESH, 2D-TORUS の 2 種類を挙げた．これらは追加リンク数 (x) は 1 であり，接続半径 (y) を 2-6 の間で変化させている．一方，ルータ間ランダムトポロジにおいて，ルータ間の接続に使われる各ルータのポート数 (d) を 4 とした．コア数は 64 でそろえた．

図 13 より，同程度の総配線長においてランダムコアリンクを用いたトポロジの方が平均 Zero-load 遅延が大幅に小さいことが分かる．この結果は．ルータ間の経路で遅延を小さくするよりも，コア・ルータ間での経路で遅延を小さくする工夫の方がより有用であるという我々の従来の研究成果によって裏付けされている．

7. まとめと今後の展望

本研究では，単一コアと複数ルータとの間でランダムなショートカットリンクを接続し，チップ上のネットワークにおける遅延の削減を目指した．

ランダムコアリンクの付加により，コア間のホップ数や Zero-load 遅延が劇的に改善する．この効果は，スイッチ間トポロジにおいてランダムにリンクを付加するよりも効率的であることも分かった．

さらに，フリットレベルシミュレーションにより，ランダムコアリンクを用いたトポロジは，従来のトポロジに比べ，低負荷時の平均遅延を最大 27%減少させた．

今後の課題としては，ランダム性を生かしてスループット向上を実現するルーティング手法の開発及び，3 次元積層のネットワークオンチップにおいて、垂直方向への (異なるチップの) ルータに対して、ランダムコアリンクを追加した場合の通信遅延の評価，さらに，end-to-end のパケットの転送エネルギーを削減するための提案手法の改良などが挙げられる．

謝 辞　本研究の一部は科学研究費 (若手 22700061)、および国立情報学研究所公募型共同研究 (一般研究公募型) の助成を受けたものである。

文　献

[1] W. J. Dally and B. Towles: “Route Packets, Not Wires: On-Chip Interconnection Networks”, Proceedings of the Design Automation Conference (DAC’01), pp. 684–689 (2001).

[2] 河野, 藤原, 松谷, 天野, 鯉渕：“ホストから複数リンクを用いた低遅延ネットワークトポロジ”, 電子情報通信学会技術研究報告 CPSY2012-77 (2013).

[3] H. Matsutani, M. Koibuchi, Y. Yamada, D. F. Hsu and H. Amano: “Fat H-Tree: A Cost-Efficient Tree-Based On-Chip Network”, IEEE Transactions on Parallel and Distributed Systems, **20**, 8, pp. 1126–1141 (2009).

[4] H. Matsutani, M. Koibuchi, H. Amano and T. Yoshinaga: “Prediction Router: Yet Another Low Latency On-Chip Router Architecture”, Proceedings of the International Symposium on High-Performance Computer Architecture (HPCA’09) , pp. 367–378 (2009).

[5] D.Burger, S.W.Keckler, K. McKinley, M.Dahlin, L. John, C.Lin, C. Moore, J.Burrill, R. McDonald, W.Yoder and the TRIPS Team: “Scaling to the End of Silicon with EDGE Architectures”, IEEE Computer, **37**, 7, pp. 44–55 (2004).

[6] P. Gratz, C. Kim, K. Sankaralingam, H. Hanson, P. Shivakumar, S. W. Keckler and D. Burger: “On-Chip Interconnection Networks of the TRIPS Chip”, IEEE Micro, **27**, pp. 41–50 (2007).

[7] Ü. Y. Ogras and R. Marculescu: “”it’s a small world after all”: Noc performance optimization via long-range link insertion”, IEEE Trans. VLSI Syst., **14**, 7, pp. 693–706 (2006).

[8] 飯尾, 平木：“三次元トポロジ NoC の比較評価”, 先進的計算基盤システムシンポジウム SACSIS (2012).

[9] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill and D. A. Wood: “The gem5 Simulator”, ACM SIGARCH Computer Architecture News, **39**, 2, pp. 1–7 (2011).