博士論文

ネットワークの拡張に基づくNoC向きの耐故障ルーティング法に関する研究

(Study on Fault-tolerant Routing Methods for NoCs Based on the Extension of Networks)

西暦2021年3月

黒川　陽太

山口大学大学院創成科学研究科

概要

コンピュータにおける計算処理を担当するプロセッサ（CPU）は，高性能化を実現するために，複数のプロセッサコア（コア）を搭載し，並列処理を行う方式が一般的になりつつある．それらのコア間の新たな接続方式として，ネットワークオンチップ（Network-on-Chip: NoC）が注目されている．NoC では，各コアにルータを付与したノードをチップ内ネットワークで接続し，パケット転送を行うことでコア間の通信を行う．このような多数のコアからなる並列システムにおいては，コア間の通信性能がシステム全体の計算性能を左右する重要な要因になる．

NoC が実装される集積回路（VLSI）チップでは，製造時や稼働時における故障の発生は避けられない．故障ノードでパケットのルーティングが行われると，パケットに誤りや欠落が生じるため，一部のノードの故障がシステム全体としての故障へと発展する．このため，大規模な NoC においては，システムの高信頼化のため，故障ノードを回避しながらパケットルーティングを行う耐故障ルーティング法が必要不可欠となる．

これまでに，様々な耐故障ルーティング法が提案されてきた．従来の耐故障ルーティングの研究では，ノード間の通信遅延を抑制するために，いかに効率よく故障ノードを迂回できるか，または，ルータの回路のオーバヘッドを抑制するために，いかに少ない回路量でルーティングアルゴリズムを実装できるかの 2 つの問題が中心的な焦点であった．しかし，一般的には，通信遅延と回路量はトレードオフの関係にあり，この 2 つの問題を同時に解決し，大規模な NoC の実現に有効な手法は未だに確立されていない．

そこで，本論文では，通信遅延と回路量の両問題を同時に解決することを目的として，ネットワークの拡張に基づく新たな概念に基づく耐故障ルーティング法を提案する．従来の耐故障ルーティング法では，ほぼ全ての手法において，ルーティングアルゴリズムを改良することにのみ主眼が置かれているため，両問題を同時に解決することは容易ではなかった．本論文では，ルーティングが行われるネットワークに着目し，ネットワークの拡張とルーティングアルゴリズムの改良の両側面から問題の解決を図る．提案手法における原理は，ネットワークにスイッチとリンクを追加する拡張を行うことで，新たな通信経路を構築することである．それにより，目的ノードまでの経路の短縮による通信性能の向上とルーティングアルゴリズムの簡略化による回路量の削減を可能にする．

具体的には，本論文では，経路が一意に定まる決定的ルーティングと複数の経路から選択可能な適応的ルーティングの両方において，ネットワークの拡張に基づく耐故障ルーティング法を提案する．ネットワークの外周部を拡張することで，外周

部に物理的な迂回路を構築し，ネットワークの端の概念をなくして，迂回に要するルーティングルールを削減可能にする．加えて，正常ノード群に論理的な迂回路を形成して，遠回りになる迂回を回避可能にする．また，ネットワークの内部を拡張することで，故障ノードの通過という新たな概念を創出し，故障ノードの迂回をなるべく回避し，ルーティングルールを簡略化可能にする．本手法の有効性を示すために，通信性能と回路量の評価を行う．

性能評価により，以下の点を明らかにした．ネットワークの外周部を拡張した手法の通信遅延は，従来手法と比較して最大約 35% 削減可能である．さらに，全体の回路量は，従来手法と比較して約 69% 削減可能であり，拡張に必要な回路の面積オーバヘッドは，約 3% であることを明らかにした．また，ネットワークの内部を拡張した手法の通信遅延は，従来手法と比較して最大約 97% 削減可能である．さらに，全体の回路量は，従来手法と比較して約 64% 削減可能であることを明らかにした．以上より，本提案手法は，耐故障性を有して高い通信性能で少ない回路量の NoC を実現することに貢献することが可能であることを明らかにした．

目次

図目次

表目次

第 1 章

緒言

1.1　本研究の背景

1940 年代に John von Neumann によってノイマン型コンピュータが考案されて以来，コンピュータの計算性能に対する要求は，増加の一途を辿っている．コンピュータは，プロセッサ（CPU），メモリ，外部記憶などのいくつかの装置で構成されるが，中でも，プロセッサは，コンピュータの頭脳に相当し，計算性能に直結する重要な装置である．そのため，パイプラインやキャッシュ，仮想記憶などをはじめ，アーキテクチャに関する様々な工夫がなされてきた．その一方で，例えば，メディア処理や機械学習，ビッグデータ解析などのように，現代のプロセッサをもってしても，計算性能が不足している処理は数多く存在する．また，最先端の科学技術計算では，例えば，新型ウィルスの新薬開発 [1] や気象予測 [2] などのように，膨大な数のプロセッサで構成される並列コンピュータを用いても，計算性能が不足している処理も多い．それらの処理を高速化し，科学技術や産業の発展に貢献するために，より高性能なプロセッサが切望されている．

従来，プロセッサの高性能化は，アーキテクチャの工夫に加えて，トランジスタの微細化による動作周波数の向上により支えられてきた．スケーリング則として広く知られている通り，トランジスタを微細化してサイズを 1/2 にすると，チップ面積は $1/k^2$ になり，動作周波数は k 倍に向上する．実際，1972 年に登場した Intel 社のプロセッサ 4004 は，トランジスタ数が約 2,300 個，動作周波数が約 700KHz であったが，2004 年に登場した Pentium では，それぞれ，約 1 億 7,000 万個，3.4GHz に向上している．しかし，現在の集積回路（VLSI）製造技術では，チップの大規模化に伴う配線遅延が顕在化し，リーク電力による消費電力や発熱の問題も深刻化しており，動作周波数の向上による高性能化は限界を迎えている．そのため，現代のマルチコアやメニーコアプロセッサなどに見られるように，小面積で低動作周波数のプロセッサコア（コア）を複数個搭載し，VLSI チップ内で並列処理を行うことで高性能化を図る方式

が一般的になっている．

多数のコアからなる並列処理システムにおいては，コア間の通信性能がシステム全体の計算性能を左右する重要な要因になる．現在のマルチコアやメニーコアプロセッサでは，コア間の接続方式として，主に共通バス方式が採用されている．この方式では，各コアがバス線に直接接続され，バス線を経由したデータ伝送により，コア間の通信を行う．共通バス方式では，コア数を増加するとバスの延長に伴い配線遅延も増加し，さらには，同時に複数のコアがバスにデータを送出できないため通信性能が低く，大規模な並列処理システムには向いていない．

この問題を解決する新たな接続方式として，ネットワークオンチップ（Network-on-Chip: NoC）が注目されている．NoC では，各コアにルータを付与したノードをチップ内ネットワークで接続し，パケット転送を行うことでコア間の通信を行う．各ノードはいくつかの隣接ノードとのみ接続されており，パケットに埋め込まれた宛先ノードの情報を基にルーティング処理を行うことで，次の出力先ノードを決定し，パケットを転送する．NoC では，チップ内ネットワーク上でパケット転送を行うことで，現在のインターネットと同様に，複数のパケットの同時転送を可能にしている．このため，従来の共通バス方式と比較すると，隣接ノード間の配線遅延が少なく，通信帯域にも優れ，コア数に対するスケーラビリティが高いという利点がある．

NoC が実装される VLSI チップでは，製造時や稼働時に発生する故障に対処することが，最も基本的で必要不可欠な課題となる．例えば，チップの製造時には，半導体の材料であるシリコンの格子欠陥，微小なゴミや不純物，回路パターン作成時のゆがみなどにより，トランジスタの MOS（Metal-Oxide-Semiconductor）構造が正常に形成されなかったり，トランジスタ間の配線が断線/短絡したりする．近年では，製造ばらつきによるトランジスタの特性不良も問題視されている．また，チップの稼働時には，温度ストレスや静電気，経年劣化などにより，同様に，トランジスタおよびその配線に故障が発生する．故障ノードで並列計算やルーティングが行われると，計算結果やパケットに誤りや欠落が生じ，部分的な故障が並列処理システム全体としての故障へと発展する．このため，以下に示す様々なレベルで耐故障化手法が研究されている．

- 回路レベルの手法
  レジスタや演算器などの回路要素を単位として冗長化を行う手法であり，2 重化，3 重化，n 個の回路要素の中から k 個の正常な要素を使用する k-out-of-n などの手法がある [3]．
- システムレベルの手法
  システム全体として故障に対処する手法であり，NoC では，以下の 2 つの手法がある．
  - ネットワーク再構成
    ネットワークに冗長なスイッチやリンクを導入し，それらを利用することで故障ノードをシステムから排除し，論理的に正常なネットワークに構成し直す手法．予

備のノードを追加し，故障ノードを代替する冗長化アプローチ [4, 5] と，正常ノードのみを用いて可能な最大サイズのネットワークを再構成する縮退アプローチ [6, 7] に基づく手法がある．

- 耐故障ルーティング
  故障ノードを含むネットワークにおいて，パケットが故障ノードに侵入しないように，パケットのルーティングで故障ノードを回避する手法．後述するように，ネットワーク全体の情報を用いるルーティングテーブルに基づく手法と，隣接ノードの情報のみを用いる回路に基づく手法がある [8]．

- アプリケーションレベルの手法
  アプリケーションの実行時に故障に対処する方法であり，以下の 2 つの手法がある．
  - 耐故障マッピング
    故障を含むネットワークにおいて，タスクが故障ノードで実行されないように，タスクのマッピングで故障ノードを回避する手法．耐故障ルーティングの機能を前提とした手法とノードへのマッピングを再度行うリマッピングに基づく手法などがある [9]．
  - リカバリ
    主にシステムの稼働中に発生する故障や障害に対処するために，タスクが正常に終了しなかった場合，タスクを再実行する手法．タスクを最初から再実行する手法とチェックポイントとして保存されたデータを用いて，必要な箇所から再開する手法などがある [10]．

高性能なプロセッサを実現するには，多数のコアを搭載すると同時に，コア間の通信性能を高める必要がある．回路レベルの冗長化やシステムレベルの冗長アプローチに基づく再構成では，故障を含まない完全に正常な NoC を構成できるものの，そのために必要な冗長要素のコストが大きく，その反面，特定の数個の要素の故障により耐故障化に失敗し，故障要素を排除しきれなくなるという欠点がある．縮退アプローチに基づく再構成では，システムサイズの縮退が可能であるため耐故障化に失敗することはないが，事前に定められたトポロジのネットワークを構成するという制約上，大規模なシステムではサイズの縮退が大きくなり，多数の正常ノードが使用されないまま無駄になるという欠点もある．一方，耐故障ルーティングは，冗長な要素を必要とせずに，システムに存在する故障に対処可能な手法である．このため，他の手法の欠点を解決/補完し合うものであり，さらに，システムの通信性能に直結するものであるため，大規模な NoC の実現には欠かせない，最も重要な要素技術の 1 つである．

これまで，様々な耐故障ルーティング法が提案されてきた．第 2 章で詳述するとおり，それらの手法は，ルーティング法の実装方法の観点で分類すると，ルーティングテーブルを用いる手法と組み込み回路として実装可能な手法に分類される．また，ルーティング法のルーティン

グ能力の観点で分類すると，デッドロックと呼ばれるパケットの循環待ちを発生させることなく，任意のノード間で 100% のパケット到着率を保証できる完全な手法とそうでない不完全な手法に分類される．これらの耐故障ルーティングの研究では，以下の 2 つの問題が中心的な焦点であった．

1. ノード間の通信遅延を抑制するために，故障ノードをいかに効率よく迂回できるか
2. ルータのオーバヘッドを抑制するために，いかに少ない回路量で実装できるか

しかし，通信遅延と回路量は一般的にはトレードオフの関係にあり，上記の問題を同時に解決し，大規模な NoC の実現に有効な手法は未だに確立されていない．

1.2　本研究の目的

NoC において，高い通信性能を実現するためには，ネットワークトポロジの性能特性も大きく関係する．代表的なトポロジとして，リング型やメッシュ型，トーラス型などがある．中でも，メッシュ型は，直径や次数，2 分割幅などの性能特性のバランスがよく，隣接ノード間の距離が短く，VLSI への平面実装が容易であるため，NoC 向きのトポロジの 1 つとして注目されている．そこで，本論文では，2 次元メッシュ NoC を対象に，通信遅延と回路量の両問題を解決することを目的として，ネットワークの拡張に基づく耐故障ルーティング法を提案する．従来の耐故障ルーティング法では，ほぼ全ての手法において，ルーティングアルゴリズムを改良することに主眼が置かれていたため，両問題を同時に解決することは容易ではなかった．本論文では，ルーティングが行われるネットワークに着目し，ネットワークを拡張することで得られる機能を活用することにより，両問題を解決可能な新たな概念に基づく耐故障ルーティング法を提案する．

本論文では，まず，2 次元メッシュ NoC の外周部にスイッチとリンクを追加する拡張を行い，物理的/論理的な迂回経路の拡張に基づく耐故障ルーティング法を提案する．本提案手法の基本原理は，ネットワークの外周部に物理的な迂回路を構築することにより，ネットワークの端の概念をなくすることで迂回に要するルーティングルールを削減可能にすることであり，迂回路に隣接する正常ノード群に論理的な迂回路を形成することにより，遠回りになる迂回を回避可能にすることである．

次に，2 次元メッシュ NoC の内部にスイッチとリンクを追加する拡張を行い，故障ノードの通過が可能な耐故障ルーティング法を提案する．目的ノードまでの経路が固定される決定的ルーティングと状況に応じて複数の経路が選択可能な適応的ルーティングの両ルーティング法を対象に，故障ノードの通過に基づく手法を提案する．本提案手法の基本原理は，従来法には全く見られなかった故障ノードの通過という新たな概念により，故障ノードの迂回をなるべく

回避可能にし，ルーティングルールを簡略化可能にすることである．

1.3　本論文の構成

本論文では，決定的ルーティングと適応的ルーティングのそれぞれに対して耐故障ルーティング法を提案する．その中で，ネットワークの外周部の拡張に基づく決定的な耐故障ルーティング法とネットワークの内部の拡張に基づく決定的と適応的な耐故障ルーティング法を提案する．本論文は 6 章より構成されている．各章の関係を図 1.1 に示し，各章の概要を以下に述べる．

第 1 章 本研究の背景と目的について述べる．

第 2 章 NoC のルーティングにおける基礎的事項を述べる．NoC の構成とパケット転送，NoC のルーティングで満たすべき要件を説明する．さらに，従来提案されてきた耐故障ルーティング法を概観し，それらに共通する問題点を述べる．

第 3 章 迂回路の拡張に基づく決定的な耐故障ルーティング法を提案する．まず，提案手法で基本手法として用いる領域ベースの耐故障ルーティング法を詳しく説明し，その問題点を指摘する．次に，この問題を解決するために，迂回路の拡張を可能にする 2 次元メッシュ NoC の構成を述べる．本章では，拡張された物理的な迂回路と論理的な迂回路を利用する 2 つの手法を提案する．本手法の通信性能を評価するために，計算機シミュレーションにより，本手法と従来法のレイテンシを比較する．また，本手法の回路量を評価するために，ルータの回路を設計し，回路実装を通じて，本手法と従来法の回路量を比較し，ネットワークの拡張に伴う本手法の面積オーバヘッドを明らかにする．

第 4 章 故障ノードの通過に基づく決定的な耐故障ルーティング法を提案する．まず，故障ノードの通過を可能にする 2 次元メッシュ NoC の構成を述べる．次に，本提案手法で基本手法として用いる XY ルーティングについて述べる．これを基に，Y 軸方向の通過が可能な手法を提案し，さらに，本手法を発展させて，ネットワークを仮想化することで，X 軸と Y 軸方向の両方向の通過が可能な手法を提案する．レイテンシ，ノード使用率，回路量について従来法との比較評価を行うことで，本手法の有効性を示す．

第 5 章 故障ノードの通過に基づく適応的な耐故障ルーティング法を提案する．本章では，提案手法で基本手法として用いる適応的な West-Last ルーティングと East-Last ルーティングについて述べ，これらを併用した故障ノードの通過に基づく適応的な手法を提案する．本手法の通信性能の評価においては，従来の適応的な耐故障ルーティング法との比較評価に加え，4 章で提案した決定的な手法との比較評価も行う．さらに，本手法の適応性を制限した場合，並びに，特定のノードが込み合うホットスポットのトラヒックの場合に対する評価を行い，本手法の有効性を示すとともに，故障ノードの通過に基づく

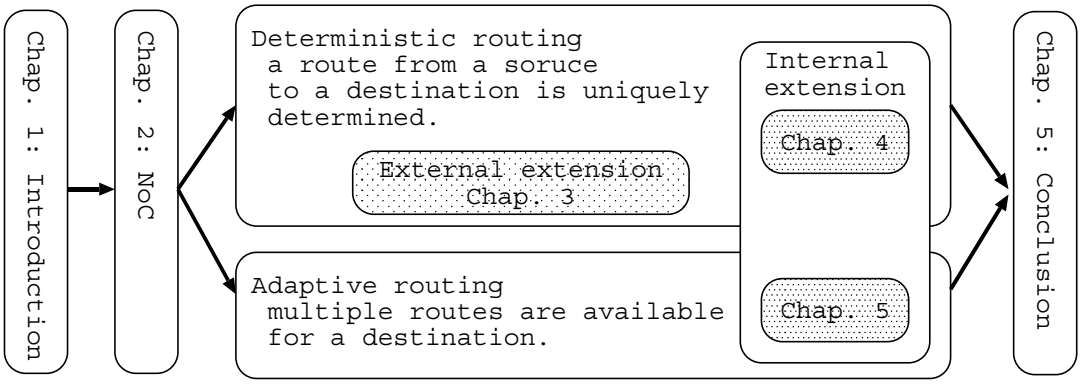


図 1.1: 本論文の構成図

適応的なルーティング法を設計する際の留意点を明らかにする．

第 6 章 ネットワークの拡張に基づく NoC 向きの耐故障ルーティング法について総括し，今後の課題を述べる．

第 2 章

NoC の構成と耐故障ルーティング法

2.1　まえがき

メニーコアプロセッサのコア間の接続方式として NoC が注目されている．NoC では，各コアにルータを付与したノードをネットワークに接続し，パケット転送を行うことでコア間のデータ転送を行う．各ノードは，いくつかの隣接ノードのみに接続されており，パケットの目的ノードの情報を基にルーティング処理を行うことで，次の出力先を決定し，パケットを転送する．NoC では，複数のパケットの同時転送を可能にしており，現在主流の接続方式である共通バス方式と比較して，隣接ノード間の配線遅延が少なく，通信帯域にも優れ，コア数に対するスケーラビリティが高いという利点がある．

NoC が実装される VLSI チップでは，製造時や稼働時に発生する故障に対処することが必要不可欠である．故障ノードでルーティングが行われると，パケットに誤りや欠落が生じて計算結果が誤り，部分的な故障がシステム全体としての故障へと発展する．このため，パケットが故障ノードに侵入しないように，ルーティングにより回避する耐故障ルーティング法を備える必要がある．

耐故障ルーティング法には，大きく分けて，ネットワーク全体の情報を用いるルーティングテーブルに基づく手法と隣接ノードの情報のみを用いる組み込み回路に基づく手法がある．これらの手法では，ノード間の通信遅延を抑制するために，故障ノードをいかに効率よく迂回できるかや，ルータの回路のオーバヘッドを抑制するために，いかに少ない回路量で実装できるかの 2 つの問題が中心的な焦点であった．しかし，一般的には，通信遅延と回路量はトレードオフの関係にあり，2 つの問題を同時に解決し，大規模な NoC の実現に有効な手法は未だに確立されていない．

本章では，上記の問題を検証するために，大規模な NoC のルーティングにおける基礎的事項を述べる．2.2 節では，本研究で対象とする NoC の構成を説明し，ネットワークトポロジや

パケット転送を行うためのルータの構成について述べる．2.3 節では，NoC におけるパケット転送について述べる．2.4 節では，NoC におけるルーティング法の要件を述べ，ルーティング法の一般的な分類を述べる．2.5 節では，従来提案されている耐故障ルーティング法について解説する．2.6 節では，本章のまとめを述べる．

2.2　NoC の構成

2.2.1　ノードの構成

図 2.1 に一般的な NoC の構成を示す．図 2.1 (a) に示すように，NoC では，コアがルータを介してネットワークで接続される．図 2.1 (b) に示すように，1 対のコアとルータをノードと呼ぶ．一般的に，NoC では，ノードが双方向リンク（2 本の単方向リンク）で接続されて，VLSI チップ上でノードのネットワークを形成する．

コアの役割は，プログラムの処理とルータへのデータ転送を行うことである．コアは，プログラムを処理するために，命令セットや拡張機能などにより様々な構成で実装される．また，ルータへデータ転送を行うために，送信するデータに目的地の情報を付与したパケットを作成し，コアからルータへパケットを送信する．

ルータの役割は，入力されたパケットを隣接ルータに転送することである．各ルータは独立して動作しており，それぞれが適切な隣接ルータにパケットを転送することで，最終的に，目的のコアまでパケットが転送される．

図 2.2 に NoC で用いられる一般的なルータを示す．以下にルータの構成要素を示す．

- 入力ユニット (Input unit)
- 出力ユニット (Output unit)
- ルーティング回路 (Route circuit)
- クロスバスイッチ (Crossbar Switch)
- スイッチアロケータ (Switch Allocator)
- 仮想チャネルアロケータ (VC Allocator)

入力ユニットは，隣接ルータまたはコアから受信したパケットをバッファに格納するものである．出力ユニットは，隣接ルータまたはコアへパケットを送信するものである．ルーティング回路は，ルーティングアルゴリズムが実装されており，そのアルゴリズムに基づいて，各入力パケットに対して次の移動先を決定するものである．クロスバスイッチは，入力ユニットと出力ユニットを接続するものである．スイッチアロケータは，クロスバスイッチの制御装置である．仮想チャネル（Virtual Channel: VC）アロケータは，各パケットが使用する VC を決定する制御装置である．ここで，VC とは，1 本の物理リンクを時分割で共有し，複数のチャネル

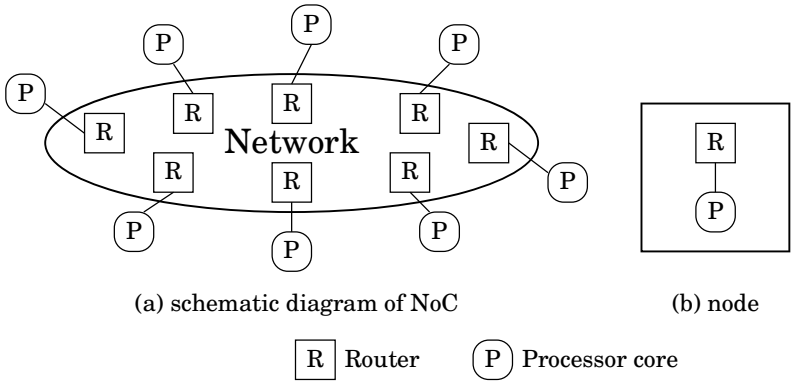


図 2.1: NoC の概略図

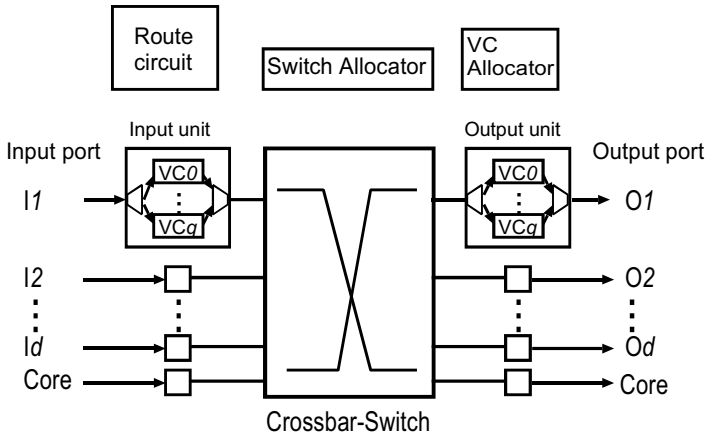


図 2.2: ルータの構成

としてみせかける技術のことである．VC により，ネットワークを仮想的に多重化したり，待ち状態のパケットを追い越したりすることが可能になる．

ルータの構成要素の数は，ネットワークトポロジと VC の数により異なる．ルータの入出力ユニットの個数は，次節で述べるネットワークトポロジの次数 d に依存し，$d+1$ 個に決定される．また，各入出力ユニット内のバッファ数 q は，使用する VC の個数と同数に決定される．

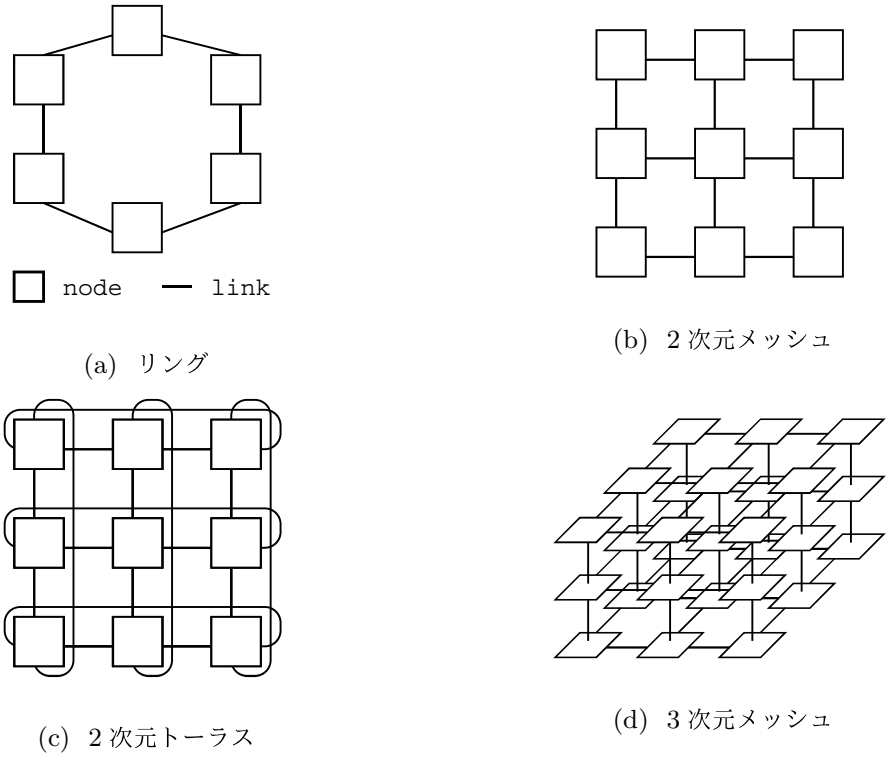
2.2.2　ネットワークトポロジ

NoC では，複数のノードを接続して様々なトポロジのネットワークが構成される．代表的なネットワークトポロジを図 2.3 に示し，表 2.1 に，それらの直径，2 分割幅，次数 [11] を示す．直径は，任意の 2 つのノード間の距離の最大値である．2 分割幅は，ネットワークを約半

分のサイズに 2 分割するために，切断しなければならないリンクの最小値である．次数は，各ノードに接続されているリンクの最大値である．直径は，目的ノードまでの最大ホップ数を意味するため，小さければホップ数が少ないことを示している．2 分割幅は，ネットワーク全体のスループットを意味するため，大きければ通信量が高いことを示している．次数は，各ルータの回路量を意味するため，小さければ少量の回路で実装可能であることを示している．これらの指標に基づいて，各ネットワークトポロジの利点や欠点を述べる．

- リング（図 2.3 (a)）
  ノードをリング状に接続したネットワーク．
  次数が低いため，ルータの回路面積を小さくすることが可能である．しかし，ネットワークがリング状で接続されているため，目的ノードまでの直径が長い．また，2 分割幅が小さいため，通信量が少ない．リングネットワークを用いた NoC の製品例として，Intel 社の Xeon Phi 5110P [12] がある．
- 2 次元メッシュ（図 2.3 (b)）
  ノードを 2 次元平面上で格子状（メッシュ状）に接続したネットワーク．
  直径がリングより短く，次数も 2 分割幅も大きいため，回路量が少なく，通信量が多い．2 次元メッシュネットワークを用いた NoC の製品例として，Intel 社の Xeon Knights Landing [13] がある．
- 2 次元トーラス（図 2.3 (c)）
  ノードを 2 次元メッシュ状に接続し，対応する端と端のノードを接続したネットワーク．
  ネットワークの端も接続されているため，2 次元メッシュよりも直径が短い．しかし，ネットワークの端と端を接続するリンクが長いため信号の伝送遅延も大きく，システム全体の動作周波数（クロック）が低下する．
- 3 次元メッシュ（図 2.3 (d)）
  2 次元メッシュ NoC を積層した 3 次元のメッシュネットワーク．
  上記いずれのトポロジよりも次数は高く，2 分割幅も大きいため，少ない通信遅延で大量の通信が可能である．しかし，次数が高いためルータの回路面積が大きい．

NoC 向きのネットワークトポロジは，ノード数のスケーラビリティが高く，平面実装が容易なものである．2 次元メッシュや 2 次元トーラスは，スケーラビリティも高く，平面実装も容易である．しかし，2 次元トーラスは，上記で述べたようにクロックが低下する問題がある．3 次元メッシュは，2 次元メッシュや 2 次元トーラスよりもスケーラビリティが高いが，次数が高いため，ルータの回路面積が増大する．さらに，3 次元に積層して実装するために，シリコンダイを貫通してリンクを配線しなければならなく，配線や位置合わせなどの実装上の問題点



(a) リング

(b) 2 次元メッシュ

(c) 2 次元トーラス

(d) 3 次元メッシュ

図 2.3: ネットワークトポロジ

表 2.1: ネットワークトポロジの性質

$N = 2^n$ ネットワークトポロジ（$n > 1$）	直径	2 分割幅	次数
リング	$N/2$	2	2
2 次元メッシュ	$2(\sqrt{N}$ - 1)	$\sqrt{N}$	4
2 次元トーラス	$\sqrt{N}$	$\sqrt{N}$	4
3 次元メッシュ	$3(\sqrt[3]{N}$ - 1)	$\sqrt[3]{N} \times \sqrt[3]{N}$	6

が多い．2 次元メッシュは，2 次元トーラスよりもクロックが高く，3 次元メッシュよりもルータを小さい回路面積で実装可能である．本論文では，NoC のルーティングの研究で広く採用されている 2 次元メッシュトポロジを対象とする．

2.2.3　2 次元メッシュ NoC

本論文で対象とする 2 次元メッシュ NoC は，同一構成のノードが行方向に m 行，列方向に n 列に配置された NoC である．このとき，各ノードには，自ノードの座標として行方向の座標 $X = \{0, 1, ..., m-1\}$ と，列方向の座標 $Y = \{0, 1, ..., n-1\}$ を与える．i 行 j 列のノードを (i, j) で表す（$i \in X, j \in Y$）．あるノード (i, j) に対し，

- $i^{'} = i$ かつ $j^{'} = j \pm 1$
- $i^{'} = i \pm 1$ かつ $j^{'} = j$

のいずれかを満たすノード $(i^{'}, j^{'})$ を隣接ノードと呼ぶ（ただし，$i' \in X, j' \in Y$ であるものとする）．また，自ノードより X 座標が大きくなる方向を東，小さくなる方向を西，Y 座標が大きくなる方向を北，小さくなる方向を南とする．

2.3　パケット転送

本節では，NoC のコア間でデータ通信を行うためのパケット転送について説明する．パケット転送とは，転送するデータに目的地のコアの情報を付与したパケットを作成し，各ルータでそのパケットを転送することである．

NoC では，一般的に，パケットはより小さいサイズのフリットと呼ばれる単位に分割される．フリットの種類を図 2.4 に示す．フリットには，目的地の座標などが格納されるヘッドフリット，データ部が分割されたボディフリット，パケットの終端を示すテイルフリットの 3 つの種類がある．ヘッドフリットが先行し，後続のフリットはヘッドフリットに追従する形で，各フリットはパイプライン式に転送される．

パケットのヘッドフリットがルータの入力ユニットに格納されると，以下の処理により，隣接ノードへ転送される．

1. ルーティング回路により出力先を決定
2. VC アロケータにより使用する VC を決定
3. スイッチアロケータによりクロスバスイッチを接続
4. 入力ユニットから出力ユニットへパケットを転送
5. 出力ユニットから隣接ノードへパケットを転送

上記の一連の処理により，競合が発生しなければ，各フリットは，VC を使用する場合は 5 サイクル，使用しない場合は 2. の処理が不要になり 4 サイクルで，隣接ノードへ転送される．

同一のルータに複数のパケットが存在する場合，上記の 2. ～ 5. の各処理において以下の競

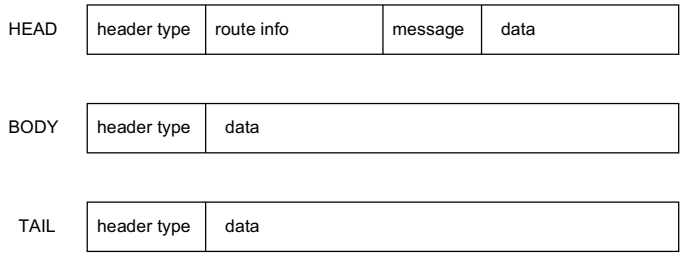


図 2.4: フリットの構成

合が発生する．

2’. 出力先の VC が使用されている場合
3’. 出力ユニットが既に他の入力ユニットと接続されている場合
4’. 出力ユニットのバッファに空きがない場合
5’. 隣接ノードの入力ユニットのバッファに空きがない場合
5”. 同じ出力ユニット内の複数の VC のバッファにパケットが存在する場合

これらの競合が発生した際，パケットは，各処理を完了するために待ち状態になる．2’. と 5”. は，VC を用いる場合に発生する競合である．5” については，同じ出力ユニット内の個別の VC のバッファに保存されているパケットは，隣接ノードの各 VC の入力バッファへ転送されるパケットであるため，物理リンクの使用権を獲得する際に競合が発生する．

一般的には，パケットの転送方式は，以下の 3 つに大別される．

- ストアアンドフォワード方式
  パケット全体を一旦バッファに受信してから，次のルータへ転送する．そのために，各ルータは，パケット全体を格納することのできるバッファを一個以上持つ必要がある．
- ワームホール方式 [14]
  パケットの構成要素であるフリットを受信したら，次のルータのバッファが空いている限り，即座に次のルータへ転送する．そのために，各ルータは，基本的に 1 フリット分を格納することのできるバッファを持つ．
- バーチャルカットスルー方式 [15]
  ワームホール方式と同様にフリットを次々に転送するものである．ただし，パケットのヘッドフリットが他のパケットによりブロックされた場合，ヘッドフリットが存在するルータのバッファにパケット全体を格納する．そのため，ストアアンドフォワード方式と同様に，各ルータは，パケット全体を格納することのできるバッファを持つ必要がある．

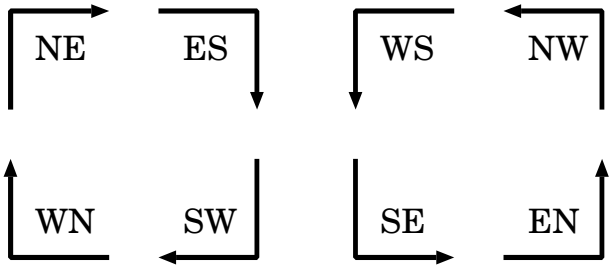
NoC では，少ない回路量で実装することが可能なワームホール方式が一般的に用いられる．

2.4　ルーティング法

2.4.1　ルーティング法の要件

本節では，NoC におけるルーティング法の要件について説明する．パケット転送において，出発地から目的地までの経路を選択するルーティングを行う場合，以下の要件を満たす必要がある．

- 耐故障性
  VLSI チップでは，製造時や稼働時に故障が発生する．そのため，パケットが故障ノードを通過すると，パケットが消失したり，パケットの一部が誤ったデータに書き換わる問題が発生する．それにより，一部の故障によりシステム全体が正常に動作しなくなる．NoC は，ネットワークを含むシステム全体がチップ上に実装されているため，故障ノードのみを取り除き，故障していない正常ノードに取り替えることは不可能である．そのため，故障の存在を前提として，故障ノードを避けながら，目的ノードまで確実にパケットを転送することを保証するような耐故障性を備える必要がある．
- デッドロックフリー性
  デッドロックとは，互いのパケットが占有する入出力ユニット内のバッファの解放を互いに待ち続け，パケットの転送が永久に停止する現象である．パケットは，図 2.5 に示す，8 つのターンが可能である．例えば，ES ターンは，あるルータにパケットが東方向（East）から入力されて，進行方向を変えて，南方向（South）に出力されることを指す．以降では，パケットの出発地を S，現在地を C，目的地を D で表す．
  図 2.6 に，デッドロックが発生する例を示す．この例では，出発地 S_i から目的地 D_i に向かうパケット i が 4 つ存在する．パケット 1 では，S_1 から D_1 に向かう途中で $(1,1)$ で SE ターンが発生する．このとき，$(1,1)$ のルータの東方向の入力ユニットがパケット 2 により占有されているため，このユニットが解放されるまで，パケット 1 は，それ以上先に進むことができない．同様に，パケット 2，3，4 も，図に示すように，進行方向の入力ユニットがパケット 3，4，1 に占有されているため，それ以上先に進むことができず，SE，EN，NW，WS ターンで構成される循環待ち（デッドロック）が発生する．これにより，これらのパケットの移動を待つ他のパケットにも待ちが発生し，ネットワーク全体でパケットの転送が永久に停止する状態に陥る．デッドロックを検知して再送する方法を用いる手法もあるが，検知のための回路がさらに必要であり，検知のために時間もかかる．そのため，デッドロックが発生しないようなルーティング法を設計



(a) clockwise direction　(b) counterclockwise direction

図 2.5: 可能な 8 個のターン

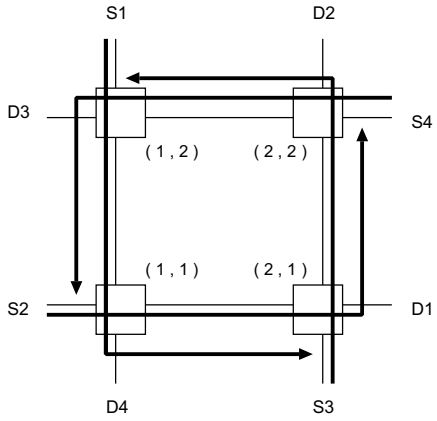


図 2.6: デッドロックの例

する必要がある．

- より高い通信性能
  並列プログラムを実行する際に，通信性能が低ければ各コアの遊休時間が増加する．このため，通信性能はなるべく高いほうが望ましい．
- 少ない回路量での実装
  VLSI チップに回路を実装する際に，回路面積が大きい箇所に故障が発生しやすい．このため，ルータの回路面積は小さいほうが望ましい．

上記の耐故障性やデッドロックフリー性を比較的容易に実現できる方法として，ルータ間の転送制御の 1 つとして，VC を利用したフロー制御 [16] がある．しかし，複数の VC を使用することにより，ルータの回路面積を増加させる原因となる．NoC では，故障の発生を抑えるため，少ない VC 数で実装されるルーティング法が望ましい．

- 制御方式

ルーティングの制御方式として，集中制御と分散制御がある．集中制御は，ネットワーク全体の経路表を用いて，出発地で目的地までの経路を決定する．分散制御は，各ルータで次の移動先を決定する．集中制御は，経路表のために多くの回路量が必要になるため，少ない回路量で実装可能な分散制御が望ましい．

2.4.2　ルーティング法の分類

ルーティング法は，ルーティングの観点から分類すると，決定的ルーティングと適応的ルーティングがある．決定的ルーティングは，出発地と目的地が決定すると一意に経路が決まる．そのため，経路が混雑している場合，その経路を避けることができない．一方，適応的ルーティングは，出発地と目的地が決定すると目的地までの複数経路の選択が可能になるため，ネットワークの状態により経路を変更することが可能である．例えば，ある経路が混雑している場合，混雑していない他の経路を選択することが可能である．

これらのルーティング法を実装する観点から分類すると，ルーティングテーブルを用いる手法と組み込み回路として実装される手法の 2 つに分類される．

- ルーティングテーブルを用いる手法
  ネットワーク内の故障情報や経路情報など様々な情報をテーブルとして各ルータが保持し，その情報を参照しながらを経路を選択する方法である．
- 組み込み回路として実装される手法
  少なくとも隣接ノードの経路情報や故障情報などを各ルータが保持し，それらの情報を参照しながら，組み込み回路により経路を選択する方法である．この方法では，システムの起動時などに，ルーティング回路に故障情報などが保存される．これらは静的な情報であるため動作時に更新されることはなく，それらの情報を入力とする組み込み回路によって経路を選択する．

いずれの手法も，2.3 節で述べたように 1 サイクルで出力先を決定できる必要がある．

ルーティングテーブルを用いる手法では，ルーティングテーブルの実装や情報の更新，出力先の決定のために多くの回路量が必要になる．さらに，ノード数が増加すると，より大きなルーティングテーブルが必要になり，ルータの回路面積は膨大になる．組み込み回路として実装される手法では，少なくとも隣接の情報だけでルーティングを行うことが可能であるため，ルーティングテーブルを用いる手法に比べてルータの回路面積を抑えることが可能である．

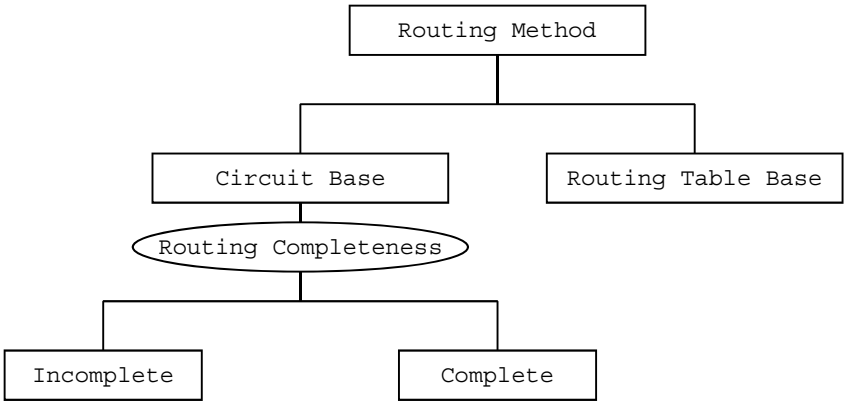


図 2.7: 耐故障ルーティング法の分類

2.5　従来の耐故障ルーティング法

本節では，従来提案されてきた耐故障ルーティング法を分類し，従来手法を概説する．図 2.7 に耐故障ルーティング法の分類を示す．図 2.7 に示すように，ルーティングテーブルを用いるルーティングテーブルベースの手法（Routing Table Base）とルータに組み込み回路として実装される回路ベースの手法（Circiut Base）に大別される．

前節で述べたように，ルーティングテーブルベースの手法では，システムのノード数に応じて回路量が膨大になる．この手法では，経路の決定処理の $\mathcal{O}$（オーダ）を削減する研究やルーティングテーブルの回路量を削減する研究が行われている．

回路ベースの手法では，ルーティングアルゴリズムが回路実装されるため，ルーティング法が重要になる．この手法では，主に，効率よく迂回するルールの研究が行われている．回路ベースの手法は，完全なルーティング法と不完全なルーティング法に分類される．

完全なルーティング法は，任意のノードから任意のノードまで，任意の位置に故障ノードが存在する場合においても，必ずパケットは目的ノードに到達することを保証できるルーティング法である．一方，不完全なルーティング法は，パケットが目的ノードに到達しない場合もあるため，プログラムの再実行やパケットの再送を行う必要があるルーティング法である．以上より，少ない回路量で効率の良い通信を行うことが可能な回路ベースの完全な耐故障ルーティング法が，高性能な NoC を実現するために適している．

以上の分類に基づき，本論文で対象とする 2 次元メッシュ NoC 向けに提案されたものを中心に，従来の手法を概説する．従来の耐故障ルーティング法について表 2.2 に示す．

テーブルベースの手法は，ルーティングテーブルで管理される情報を参照して，パケット

の出力先を決定する手法である．Hsin ら [17] は，パケットの通過回数などの情報を用い，アントコロニー法によりトラヒック分散を行う手法を提案している．この手法では，100% のパケット到達率は保証されていない．Liu ら [18] は，4 ホップ内のノードの故障情報とルータの入力バッファの情報を用いる耐故障ルーティング法を提案している．Zhao ら [19] は，ネットワーク内の全ノードの故障情報を用い，最短経路を計算して出力先を決定する手法を提案している．Mota ら [20] は，ネットワークテーブルを最適化して回路量を減らした耐故障ルーティング法を提案している．Xie ら [21] は，2 ホップ内のノードの故障情報を用い，南北方向だけに 2 つの VC を用いる耐故障ルーティング法を提案している．

回路ベースの手法は，故障領域の作成や隣接ノードの故障情報の設定により，それらの情報を参照してパケットの出力先を決定する手法である．以下に，不完全なルーティング法と完全なルーティング法を示す．

- 不完全なルーティング法
  Sinha ら [22] は，XY ルーティング法をベースとして，VC を切り替えながらルーティングを行う手法を提案している．Moscibroda ら [23] は，バッファレスの手法として，パケットを次々と転送し，目的地に到着しない場合は，再送を繰り返す手法を提案している．Janfaza ら [24] は，タイムアウトと再送を繰り返すことで適応的ルーティングを実現可能な手法を提案している．これらの手法は，ルーティングアルゴリズムだけでは，任意のノードから任意のノードへパケットをすべて到達させることができないため，タイムアウトや再送を行うものである．そのため，通信処理に時間を要する．
- 完全なルーティング法
  以下に，決定的ルーティングと適応的ルーティングの従来手法を示す．
  - 決定的ルーティング法
    Chalasani ら [25] は，非凸型の故障領域作成し，その周辺を迂回する手法を提案している．この手法では，故障領域の形状として，基本的な矩形に加えて，T 型や L 型，凹型などにも対応可能であり，1 リンクあたり 4 個の VC を必要とする．Boppana ら [26] は，矩形の故障領域を作成し，目的地の場所に応じて迂回方向を決定する手法を提案している．この手法では，1 リンクあたり，VC を 4 個使用する．Chen ら [27, 28]，Fukushima ら [29]，Fu ら [30] は，矩形の故障領域を作成し，VC を追加することなく迂回可能なルーティング法を提案している．これらの手法では，局所的な情報のみを用いて厳密なルールが定義されているため，コンパクトにルータに組み込み可能である．しかし，故障領域が大きくなると，迂回のために経路長が長くなる．
    2 次元メッシュを対象とした上記の手法とは異なり，トポロジ非依存の手法も研究されている．この手法は，様々なネットワークトポロジに対応した手法であるた

め，故障ノードが含まれるネットワークにおいても使用可能な手法である．Sancho ら [31] は，ネットワーク内に論理的なツリーを構築し，Up 方向と Down 方向の移動を行う手法を提案している．他にも，参考文献 [32] のサーベイ論文で紹介されている手法がある．これらの手法は，様々なネットワークトポロジに対応可能ではあるが，ネットワークトポロジの性能を最大限使用することができないため，トポロジ依存の手法よりも通信性能の向上は困難である．

– 適応的ルーティング
Shamaei ら [33] は，凸型の故障領域を作成し，1 つの VC では適応的ルーティングを行い，その他の VC では決められたルーティングを行う．Boppana ら [26] は，決定的ルーティングの耐故障ルーティング法 [26] を用いて，既存の適応的ルーティング法を耐故障化する方法論を提案している．Zhou ら [34] は，凸型の故障領域を作成し，VC を切り替えながら適応的にルーティングを行う手法を提案している．Wu ら [35] は，凸型の故障領域を作成し，故障領域周りの 2 列を確保することによりルーティングを行う手法を提案している．

2.4 節のルーティング法の要件で述べた耐故障性とデッドロックフリー性を満たす手法は数多く提案されている．VC を使用してネットワークを多重化する手法は，デッドロックフリー性を有するために，複雑なルーティングアルゴリズムが必要である．さらに，VC を使用しない手法では，ネットワークを仮想的に多重化しないため，VC を使用する手法よりも複雑なルーティングアルゴリズムが必要になる．また，上記で述べた数多くの手法は，故障ノードまたは故障領域を迂回する必要がある．そのため，故障ノードの迂回による通信性能の悪化とルーティングアルゴリズムの複雑さによる回路量の増大の 2 つの問題を有している．

表 2.2: 耐故障ルーティング法

著者	ルーティング	特徴	VC 数
ルーティングテーブルベースの手法			
H.K. Hsin [17]	アントコロニー法	パケット到達率 100% 以下	1
J. Liu [18]	4 ホップ先の経路情報	混雑回避が可能	1
H. Zhao [19]	最短経路を選択可能	探索時間が必須	1
R.G. Mota [20]	テーブルの最適化手法	少ない回路量	1
R. Xie [21]	2 ホップ先の経路情報	南北方向だけ 2 つの VC が必要	2
回路ベースの手法			
不完全なルーティング法			
D. Sinha [22]	次元順手法が基本手法	パケット到着率 100% 以下	複数
T. Moscibroda [23]	タイムアウトと再送	バッファが不必要	1
V. Janfaza [24]	タイムアウトと再送	パケット到着率 100% 以下	2
完全なルーティング法			
決定的ルーティング			
S. Chalasani [25]	非凸型の故障領域を迂回	故障の発生場所に制限	4
R. V. Boppana [26]	矩形の故障領域を迂回	矩形の故障領域の作成	4
K.H. Chen [27]	矩形の故障領域を迂回	矩形の故障領域の作成	1
R. Holsmark [28]	Chen[27] のルールを修正	矩形の故障領域の作成	1
Y. Fukushima [29]	矩形の故障領域を迂回	小さい矩形の故障領域	1
B. Fu [30]	矩形の故障領域を迂回	大きな故障領域が必要	1
J.C. Sancho [31]	Up*/Down*	トポロジ非依存	1
適応的ルーティング			
A. Shamaei [33]	矩形の故障領域を迂回	矩形の故障領域を作成	4
R. V. Boppana [26]	Boppana[26] の拡張手法	VC を追加して適応化	4 + 複数
J. Zhou [34]	凸型の故障領域を迂回	3 つの VC で実現可能	3
J. Wu [35]	Odd-Even[36] が基本手法	故障領域の作成に制限	1 / 複数

2.6　むすび

本章では，NoC の概要について述べ，本研究で対象とする NoC の構成やネットワークトポロジ，パケット転送について説明した．また，NoC のルーティングで満たすべき要件を説明した．さらに，従来の耐故障ルーティング法を概観し，それらに共通する問題点を述べた．

2.2 節では，NoC の構成を説明し，各ネットワークトポロジの利点やルータの構成について述べた．2.3 節では，NoC のデータ転送のパケット転送について述べた．2.4 節では，ルーティング法の要件や分類について述べた．そこで，ルーティング法が，耐故障性やデッドロックフリー性を有し，高い通信性能で少ない回路量で実装されることが望ましいことを示した．2.5 節では，NoC で用いられる従来の耐故障ルーティング法ついて解説した．

本章で述べたルーティング法の要件を満たすように，3 章，4 章，5 章で耐故障ルーティング法を提案する．

第 3 章

迂回路の拡張に基づく決定的な耐故障ルーティング法

3.1 まえがき

NoC を VLSI チップ上に実装し，コア間の正常なデータ転送を行う大規模な並列システムを実現するために，パケット転送で故障ノードを回避する耐故障ルーティング法が必要不可欠になる．耐故障ルーティング法は，2.5 節で述べたように，様々な手法が研究されている．その中でも，ルーティングテーブルを用いる手法や VC を用いる手法では，故障ノードを矛盾なく迂回するためのルーティングアルゴリズムの設計は比較的容易であり，数多く研究されている．しかし，これらの手法は，多くの回路を用いて実装されるため，ルータに故障が発生しやすくなる．

そこで，NoC 向きの手法として，ルーティングテーブルや VC を用いない手法である決定的な領域ベースの耐故障ルーティング法 [27, 28, 29] がある．この手法では，故障ノードを含む故障領域を作成し，その周辺を迂回するためのルーティングルールが厳格に決められている．そのため，ルーティングアルゴリズムが複雑であり，それが実装されるルーティング回路部に多くの回路量が必要になる．

本章では，Fukushima らの耐故障ルーティング法 [29] を基本手法とし，迂回路を拡張することによりルーティングアルゴリズムを簡略化する耐故障ルーティング法を提案する．まず，本手法は基本手法と同様の故障領域や定義などを用いるため，基本手法である領域ベースの耐故障ルーティング法の詳細について説明し，その問題点を述べる．次に，この問題を解決するために，迂回路の拡張を可能にする 2 次元メッシュ NoC のアーキテクチャを提案する．このアーキテクチャに基づき，迂回路を拡張する手法を提案する．本手法の有効性を明らかにするために，計算機シミュレーションにより通信性能の評価を行う．また，本手法の回路量を評価

するために，ルータの回路を設計し，回路実装を通じて，回路量の比較を行い，ネットワークの拡張に伴う本手法の面積オーバヘッドを明らかにする．

3.2 節では，本章で提案する手法の基本手法になる領域ベースの耐故障ルーティング法の詳細と問題点を述べる．3.3 節では，本章で提案する迂回路を拡張することが可能なアーキテクチャを述べる．3.4 節では，拡張された物理的な迂回路を利用する迂回路の拡張に基づく耐故障ルーティング法である E2DM-Message と，その手法のデッドロックフリー性を述べる．さらに，E2DM-Message をもとに，拡張された論理的な迂回路を利用する手法である E2DM-Extend-Message を提案する．3.5 節では，提案した 2 つの耐故障ルーティング法の通信性能の評価を述べる．3.6 節では，本研究で用いるルータを回路実装するために必要な各ユニットと迂回路の拡張に基づく耐故障ルーティング法に必要なユニットについて説明する．3.7 節では，実装した回路の回路量を評価し，提案手法の拡張部分における面積のオーバヘッドについて述べる．3.8 節では，本章のまとめを述べる．

3.2　基本手法

3.2.1　領域ベースの耐故障ルーティング法

本節では，故障領域を作成して矛盾なくパケット転送を行うことが可能な提案手法の基本手法となる領域ベースの耐故障ルーティング法 [29] について説明する．本手法では，故障ノードを含む故障領域を作成し，その周辺を迂回することで，パケットが故障ノードに侵入することを防ぐ．故障領域を矛盾なく迂回するために，パケットに格納されるメッセージの情報により迂回ルールが決められている．本論文では，本手法を Message と呼ぶ．以下で，Message のルーティングアルゴリズムについて述べる．

まず，故障ノードを含む故障領域が作成される手順について述べる．Message では，故障領域を作成するために，以下で定義する処理が新たな不使用ノードが生成されない定常状態になるまで，繰り返し行われる．

定義 3.1. ある正常ノード (i, j) に対し，その 4 近傍に故障ノードが 2 つ以上存在する場合，(i, j) を不使用ノードとする．ただし，$i-1 \leq i' \leq i+1$ かつ $i' \in X$ であり，$j-1 \leq j' \leq j+1$ かつ $j' \in Y$ である．

上記の定義 3.1 を不使用ノードが新たに生成されなくなるまで，繰り返すことで，矩形の故障領域が作成される．

作成された故障領域にパケットが侵入しないようにするために，故障領域の周辺に迂回路が作成される．作成される迂回路には，故障領域の場所ごとに，以下に示すように，種類が定められている．

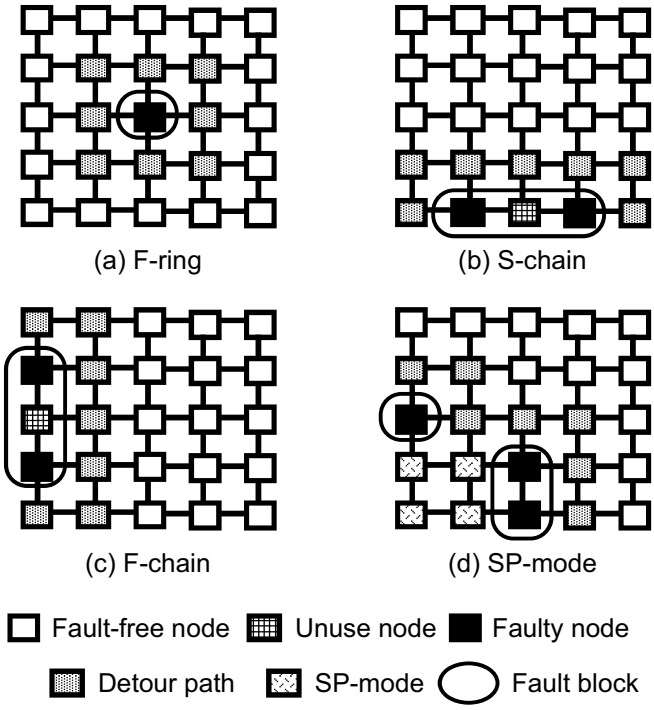


図 3.1: 迂回路の種類

- F-ring (Fault ring): ネットワークの内側にある故障領域からなる迂回路
- S-chain (South chain): ネットワークの南側に接している故障領域からなる迂回路
- F-chain (Fault chain): ネットワークの西側に接している故障領域からなる迂回路

ここで，ネットワークの南西の端の故障領域の場合，本論文では，S-chain として扱う．

迂回路の例を図 3.1 に示す．図 3.1 (a) ∼ (c) に示すように，故障領域の場所により，これらの迂回路が作成される．作成された迂回路には，矛盾なく故障領域を迂回するために，以下で定義する参照ノードと呼ばれるノードが設定される．

定義 3.2. 迂回路の北東のノードを参照ノード R として扱い，その座標を (R_x, R_y) で表す．

さらに，Message では，ある特定の故障パターンから SP-mode と呼ばれるエリアが作成される．SP-mode は，最も南西の S-chain と F-chain の迂回路が重なる場合に作成され，その迂回路に囲まれたすべてのノードで構成されるエリアである．図 3.1 (d) に示すように SP-mode が作成される．SP-mode においても，以下で定義する北東を示すノードが設定される．

定義 3.3. SP-mode の最北のノードの座標を SP_y，最東の座標を SP_x で表す．

Message では，これらの故障情報とパケットに格納されるメッセージにより移動方向が決定される．以下に，メッセージの種類を示す．

- RF: 西方向への移動
- CF-SN: 北方向への移動
- CF-NS: 南方向への移動
- RO: 東方向への移動

これらのメッセージには優先度が定義されており，RF が一番高く，RO が一番低い．また，優先度の低いメッセージから高いメッセージに変更することは不可能である．このメッセージを各ルータで出力先を決定する際に切り替えながら，目的ノードまでパケット転送が行われる．

図 3.2 に，アルゴリズムの簡略図を示す．例えば，F-ring に西方向へ移動するパケットが直面した場合，F-ring で RF の迂回を行う．このように，故障領域とメッセージごとに迂回ルールが決められている．図 3.3 に，Message のルーティング例を示す．図 3.3 (a) , (c) では，パケットの目的地が西にあるため，最初に，RF で移動を行い，その後，CF-SN で移動を行う．図 3.3（b）では，目的地が東にあるため，最初に，CF-NS で移動を行い，その後，RO で移動を行う．このように，本手法は，デッドロックフリーのためにルーティングテーブルや VC を用いる必要がない．

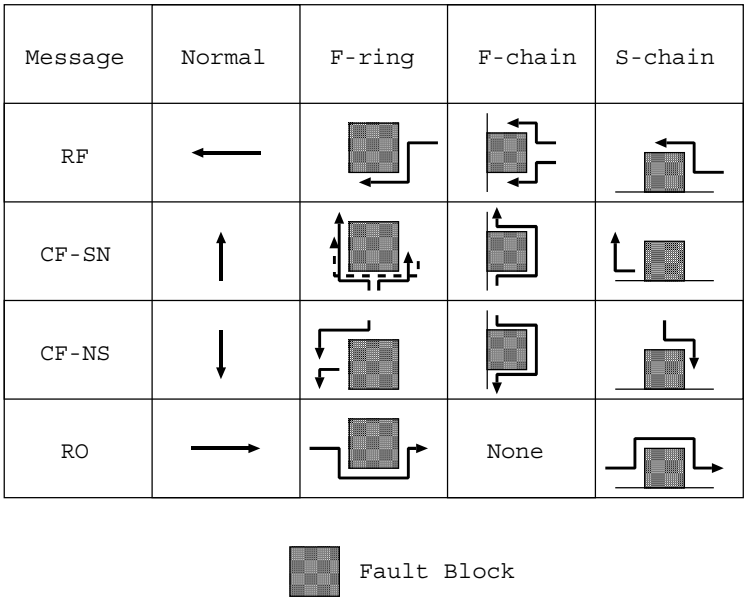


図 3.2: Message のルーティングアルゴリズムの概略図

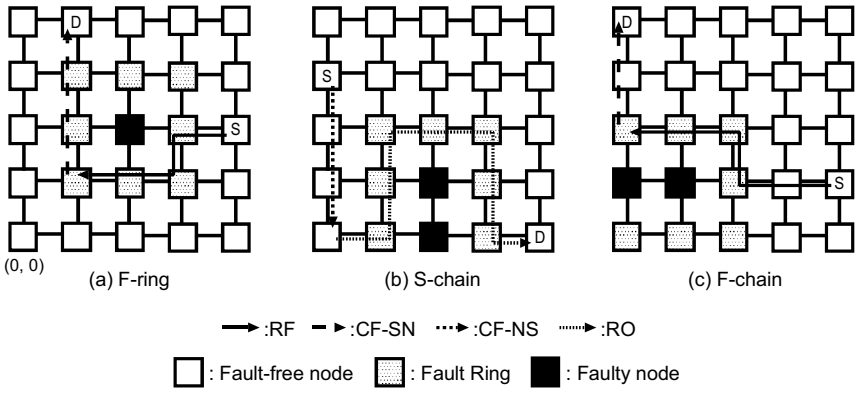


図 3.3: 領域ベースの耐故障ルーティング例

3.2.2　基本手法の問題点

前節で述べた Message の問題点について述べる．Message は，3.2.1 節で述べたように，迂回路ごとに複数のルーティングルールが必要である．そのため，ルーティングアルゴリズムが複雑であり，ルータをより少ない回路量で実装することは困難である．

Message では，F-ring と S-chain，F-chain の迂回路に加えて，さらに，SP-mode の迂回ルールが必要である．SP-mode の重要性を述べるために，図 3.4 に SP-mode の迂回例を示す．SP-mode がない場合，図 3.4（a）に示すようにデッドロックが発生する．図 3.4（a）では，パケット 1 と 3 により左下のターンが発生し，パケット 2 と 4 により左上のターンが発生する．各ターンが発生しただけではデッドロックは発生しないが，この 2 つのターンが同時に発生することにより無限大型（∞ 型）のデッドロックが発生する．そこで，この手法では，図 3.4（b）に示すように SP-mode を作成する．これにより，パケット 1 が最初に北へ移動するため，左下のターンが発生せず，デッドロックは発生しない．このように，デッドロックフリー性を有し，任意の故障ノードに対応することが可能な Message は，ルーティングルールが複雑である．そのため，図 2.2 に示すルータのルーティング回路に多くの回路が必要になり，ルータ全体の回路量も多くなる．

3.3　迂回路の拡張を可能にする 2 次元メッシュ NoC の構成

本節では，ネットワークの外周部を拡張し，従来の耐故障ルーティング法の複雑なルーティングアルゴリズムの簡略化を可能にするためのアーキテクチャを提案する．

ネットワークの外周にスイッチとリンクを配置した提案するアーキテクチャを図 3.5 に示す．このアーキテクチャは，以下で定義するネットワークの外周のノード間にスイッチとリン

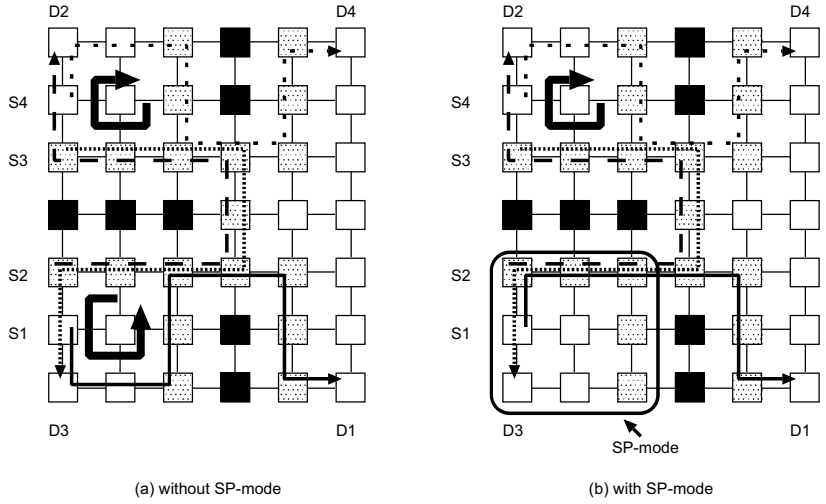


図 3.4: SP-mode の例

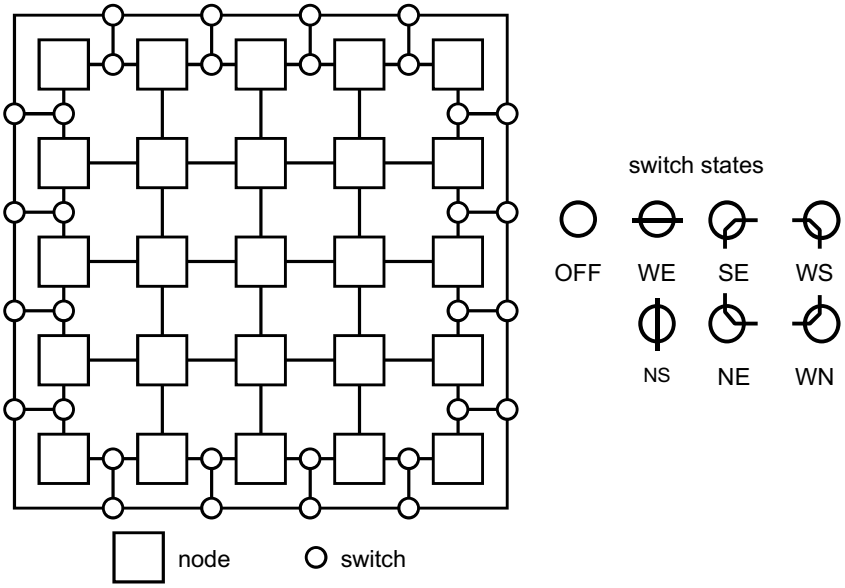
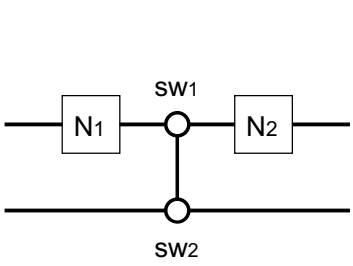


図 3.5: 迂回路の拡張に基づく 2 次元メッシュ NoC のアーキテクチャ

クを追加したものである．

定義 3.4. 外周のノード (i,j) は，ネットワークの端に接するノードである．ただし，$0 \leq i \leq m-1$ かつ $j = 0, n-1$ ，または，$0 \leq j \leq n-1$ かつ $i = 0, m-1$ であるものとする．

追加したスイッチは，様々なスイッチのステートを有しており，隣接の故障ノードを基にス



switch state

N_1	N_2	SW_1	SW_2
0	0	WE	OFF
0	1	WS	NE
1	0	SE	WN
1	1	OFF	WE

0 : Fault-free node
1 : Faulty node

図 3.6: 南端の場合のスイッチステート

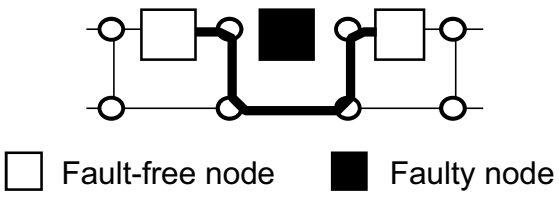


図 3.7: 南端の故障ノードに対するスイッチの設定例

テートが設定される．スイッチのステートを図 3.5 に示す．西と東の方向が接続されているステートを WE ステートと表し，他のステートも同様に表す．また，どの方向にも接続されていないステートを OFF ステートと表す．図 3.6 にネットワークの南端の場合のスイッチのステートを示し，ステートの切り替わる条件を説明する．例えば，ノード N1 が故障ノードでありノード 2 が正常ノードである場合，スイッチ SW1 は，SE ステートになり，スイッチ SW2 は，WN ステートになる．このように，南端以外でも図 3.6 と同様にスイッチを切り替える．上記の動作は，システムの起動時などに行われる．各スイッチは，隣接ノードが故障しているかどうかを確認し，故障している場合，自動で一意に切り替わる．

南端に故障が発生した場合の例を図 3.7 に示す．図に示すように，故障ノードの周りのスイッチを切り替えることで，故障ノードの南側の経路を使用することが可能になる．同様に，他のネットワークの端に故障ノードが存在する場合においても，拡張した経路を使用することが可能になる．そのため，ルーティングにおいて，ネットワークの端を考慮する必要がなくなる．

3.4　迂回路の拡張に基づく決定的な提案手法

3.4.1　物理的な迂回路の拡張に基づく決定的な耐故障ルーティング法: E2DM-Message

図 3.5 に示すアーキテクチャにより，ネットワークの端を考慮する必要がなくなるため，迂回ルールを簡略化することが可能になる．本節では，図 3.5 に示すアーキテクチャを対象に，従来の耐故障ルーティング法である Message を基にした耐故障ルーティング法を提案する．本論文では，本手法を E2DM-Message と呼ぶ．

Message では，故障領域の場所ごとに F-ring と S-chain，F-chain の迂回ルールが必要であったが，提案したアーキテクチャを用いることで，ネットワークの端の迂回ルールを考慮する必要がなくなる．そのため，S-chain の迂回ルールは，このアーキテクチャにより南側の経路を使用することが可能になるため，必要なくなる．同様に，故障ノードが西端に接する場合，西側の経路を使用することが可能になるため，F-chain の迂回ルールも必要なくなる．さらに，Message では，S-chain と F-chain が重なった場合に作成される SP-mode のエリアも，本手法では，必要なくなる．これにより，S-chain や F-chain，SP-mode の迂回ルールを簡略化することが可能である．以上から，3 つの迂回ルールを削除することが可能になるため，E2DM-Message では，F-ring の迂回ルールだけで，任意の場所の故障領域を迂回することが可能になる．

以下でアルゴリズムの記述に必要な変数を定義する．

定義 3.5. パケットの S，C，D の座標を，それぞれ，(S_x, S_y), (C_x, C_y), (D_x, D_y) で表す．

定義 3.6. あるノードの東西南北の隣接ノードが故障ノードか否かを表すフラグを，それぞれ，F_E, F_W, F_S, F_N とする．各フラグは，故障ノードであれば 1，そうでなければ 0 の値をとるものとする．

アルゴリズム 3.1 に，E2DM-Message のルーティングアルゴリズムを示す．このアルゴリズムは，Message の F-ring の迂回ルールに，拡張した経路を使用するルールを追加したものである．その追加した部分を太字で示す．また，メッセージを決定するアルゴリズムをアルゴリズム 3.2 に示す．Message_select() は，現在のメッセージを返す関数である．このメッセージの情報をアルゴリズム 3.1 の message の変数に格納し，message を基に次の出力先である Route が決められる．

図 3.8 に，提案するルーティングアルゴリズムの概略図を示す．本手法では，S-chain や F-chain の迂回ルールは必要ない．図 3.9 に，E2DM-Message によるルーティング例を示す．

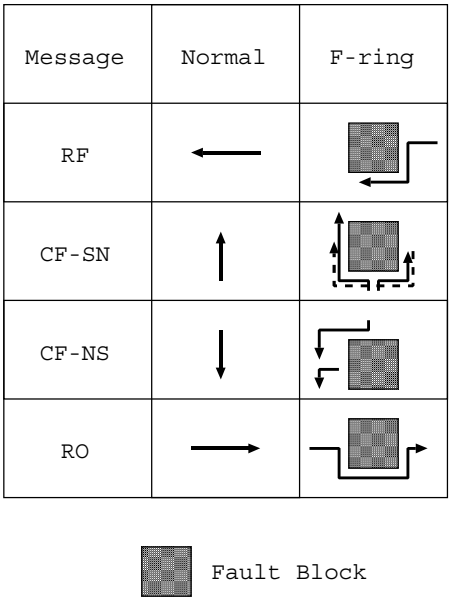


図 3.8: E2DM-Message のルーティングアルゴリズムの概略図

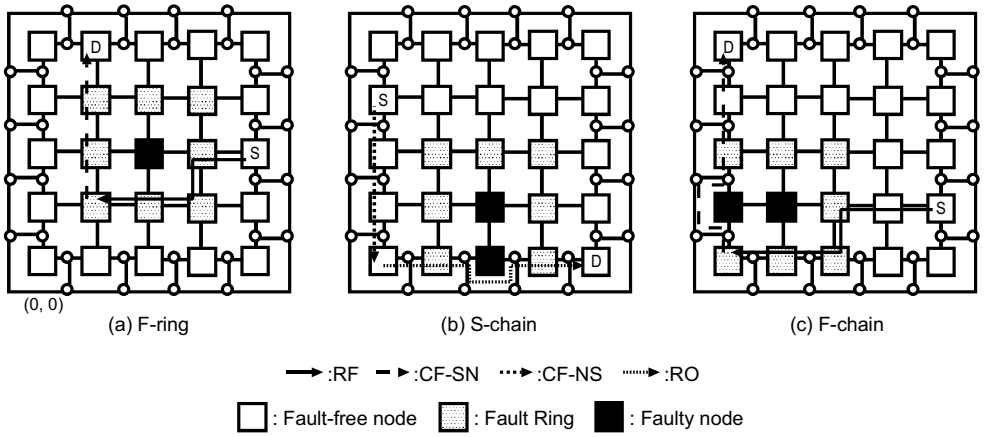


図 3.9: E2DM-Message の耐故障ルーティング例

図 3.9 (a) では，図 3.3 (a) と同様の経路を選択する．図 3.9 (b), (c) の場合，図 3.3 (b), (c) とは異なる経路を選択する．図 3.9 (b) では，S-chain の南側を通る経路が選択される．図 3.9 (c) では，F-chain の西側を通る経路が選択される．これらのように，どこに故障領域が接している場合においても F-ring の迂回ルールのみで，故障領域を迂回することが可能である．

アルゴリズム 3.1: E2DM F-ring Route

```
1 E2DM_Message_F−ring_Route(C, D)
2 if (C == D)
```

```
3           Route = Core
4   elsif (message == RF)
5           if (F_W == 1)
6                   Route = South
7           elsif (C_x < D_x && C_y == 0)
8                   Route = West
9           else
10                  Route = West
11          endif
12  elsif (message == CF-SN)
13          if (C_x < D_x && R_y > D_y)
14                  if (F_N == 1)
15                          Route = East
16                  elsif (F_E == 1 && C_y == 0)
17                          Route = East
18                  elsif (F_E == 1)
19                          Route = South
20                  else
21                          Route = North
22                  endif
23          else
24                  if (F_N == 1 && C_x == 0)
25                          Route = North
26                  elsif (F_N == 1)
27                          Route = West
28                  elsif (F_W == 1 && C_x == R_x && C_y == 0)
29                          Route = West
30                  elsif (F_W == 1)
31                          Route = South
32                  elsif (F_W == 0 && C_x == R_x)
33                          Route = West
34                  else
35                          Route = North
36                  endif
37          endif
38  elsif (message == CF-NS)
39          if (F_E == 1)
40                  Route = West
41          elsif (F_S == 1 && C_x == 0)
42                  Route = South
43          elsif (F_S == 1)
44                  Route = West
45          else
46                  Route = South
47          endif
48  else /* message == RO */
49          if (F_E == 1)
50                  Route = South
51          elsif (C_x == R_x && C_y < D_y)
```

```
52                  Route = North
53          elsif (C_y == 0)
54                  Route = East
55          else
56                  Route = East
57          endif
58  endif
```

アルゴリズム 3.2: メッセージの決定

```
1   Message_select(C, D)
2   if (C_x > D_x)
3           message = RF
4   elsif (C_y > D_y)
5           message = CF-SN
6   elsif (C_y < D_y)
7           message = CF-NS
8   else
9           message = RO
10  endif
```

3.4.2　E2DM-Message のデッドロックフリー性

提案した耐故障ルーティング法である E2DM-Message のデッドロックフリー性を示す．そのために，以下の定義を導入する．

定義 3.7. 2 つのターン T_1 と T_2，パケットの集合 $p_1, p_2, ..., p_l$ に対して，p_1 が T_1 を，p_l が T_2 を発生させ，p_i が p_{i+1} を待つ場合，T_1 と T_2 は連結しているという（$1 \leq i \leq l-1$）．

定理 3.1. E2DM-Message は，デッドロックフリーである．

証明. E2DM-Message において，図 2.5 に示したターンにより，時計回りと反時計回りのパケットの循環待ちが発生しないことを示す．E2DM-Message では，故障ノードがない場合は，メッセージに従い動作するため，ES, SW, NW, EN ターンは発生しない．故障ノードを迂回する際に，これらのターンが発生するが，その場合でも，パケットの循環待ちが発生しないことを示す．

時計回りの方向について考える．ここで，ES ターンと SW ターンが連結しないことを示す．ES ターンは，パケットが RO のメッセージで故障領域の西端に直面した場合（図 3.8 の RO メッセージ)，故障領域の迂回路の西側で発生する．ある故障領域の迂回路の西側において，南側に転送されなければならないパケットが存在すると仮定する．そのパケットは，図 3.8 の CF-NS メッセージの迂回ルールにおいて，一度，西側に転送されてから南側に転送される．それゆえに，ES ターンと SW ターンによる連結は発生しない．

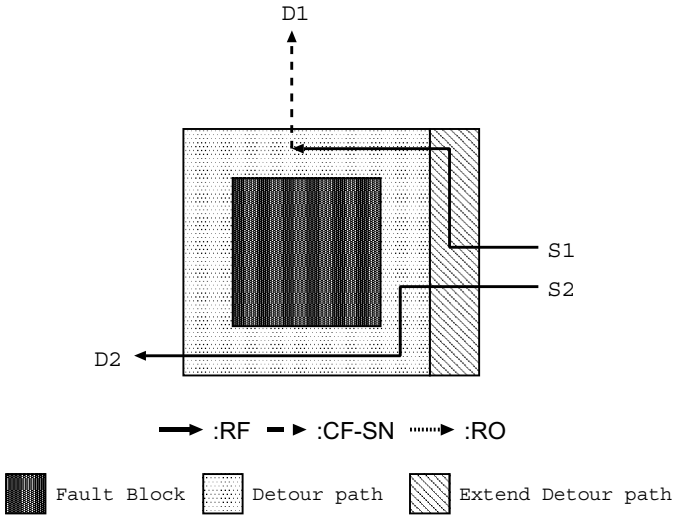


図 3.10: E2DM-Extend-Message ルーティング例

反時計回りの方向について考える．ここで，EN ターンと NW ターンが連結しないことを示す．EN ターンは，パケットが RO のメッセージで故障領域の参照ノード R_x と同じ列の場合（図 3.8 の RO メッセージ），故障領域の迂回路の南東で発生する．ある故障領域の迂回路の東側において，北側に転送されなければならないパケットが存在すると仮定する．そのパケットは，図 3.8 の CF-SN メッセージの点線の迂回ルールにおいて，参照ノード R を経由しない．なぜなら，基本手法として用いた Message において，参照ノードを経由する場合，デッドロックが発生することが明らかになっており，南から北方向への参照ノードの転送はルーティングアルゴリズムで禁止されている．それゆえに，E2DM-Message でも同様のルールを禁止しており，EN ターンと NW ターンの連結は発生しない．

以上から，E2DM-Message は，デッドロックフリーである． □

3.4.3　論理的な迂回路の拡張に基づく決定的な耐故障ルーティング法: E2DM-Extend-Message

目的ノードの場所により，遠回りの迂回により通信性能が悪化する可能性がある．本節では，3.4.1 節で述べた E2DM-Message に基づき，より最短経路を選択することが可能な論理的な迂回路を拡張した耐故障ルーティング法を提案する．本論文では，本手法を E2DM-Extend-Message と呼ぶ．

E2DM-Message では，図 3.9 に示すように，目的ノードが出発ノードよりも北側にあり，パケットが故障領域に直面した際に，南側に迂回する．そのため，経路長が長くなり，通信性能

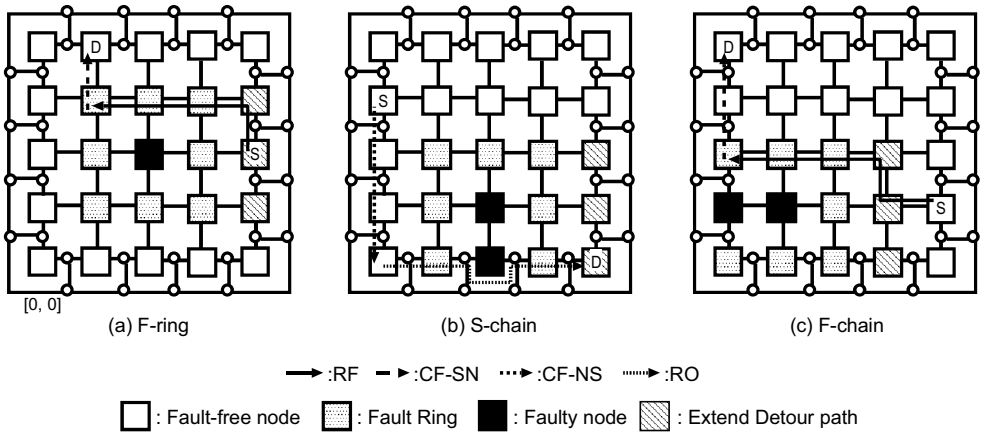


図 3.11: E2DM-Extend-Message の耐故障ルーティング例

が低下する．

この問題を解決するために，本手法では，迂回路に隣接する東側の列に新たな迂回路を拡張する．新たな迂回路の作成ルールを以下に定義する．

定義 3.8. 拡張する迂回路は，ある故障領域の参照ノード R の隣接ノード $(R_x + 1, j)$ の正常ノードに作成される．ただし，j は，その故障領域の迂回路の南東の Y 座標 FB_{SE_y} から参照ノードの Y 座標 R_y までである．すなわち，$FB_{SE_y} \leq j \leq R_y$ である．また，拡張する迂回路上に故障ノードが存在する場合，この迂回路を作成しない．

図 3.10 に拡張する迂回路とルーティングの例を示す．この図 3.10 のように，目的ノードが北西の場合，拡張した迂回路では，パケットを北方向に転送する．南西の場合，拡張した迂回路を使用せずこれまでのルール通り転送する．これにより，目的ノードが北西の場合において，経路長の短縮が可能である．

アルゴリズム 3.3 に，E2DM-Extend-Message のルーティングアルゴリズムを示す．E2DM_Extend_Message_partial_F-ring_Route() は，拡張した迂回路のみで用いられる関数である．また，これ以外のアルゴリズムは，E2DM-Message と同じアルゴリズム 3.1 を用いる．

図 3.11 に，E2DM-Extend-Message によるルーティング例を示す．図 3.11 (a), (c) では，いずれの場合においても，図 3.9 とは異なり，最短経路を選択することが可能である．

アルゴリズム 3.3: E2DM partial F-ring Route

```
1 E2DM_Extend_Message_partial_F−ring_Route(C, D)
2 if (C == D)
3         Route = Core
4 elsif (Message == RF)
5         if (R_y == C_y)
```

```
6                   Route = West
7           elsif (R_y ≤ D_y)
8                   Route = North
9           else
10                  Route = West
11          endif
12  endif
```

3.4.4　E2DM-Extend-Message のデッドロックフリー性

提案した耐故障ルーティング法である E2DM-Extend-Message のデッドロックフリー性を示す．

定理 3.2. E2DM-Extend-Message は，デッドロックフリーである．

証明. E2DM-Message はデッドロックフリーなルーティング法であり，E2DM-Extend-Message は，E2DM-Message を拡張した手法であるため，拡張された迂回路で発生するターンによる，パケットの循環待ちが発生しないことを示す．拡張された迂回路では，新たに NW ターンが発生する．その場合において，NW ターンが発生するパケットの連結によりパケットの循環待ちが発生しないことを示す．

NW ターンが関係するデッドロックである反時計回りの方向について考える．そこで，NW ターンと EN ターンが発生するパケットが連結しないことを示す．EN ターンは，パケットが CF-SN と RO のメッセージで故障領域の参照ノード R と同じ列の場合（図 3.8 の RO メッセージ），既存の迂回路の南東のみで発生する．拡張された迂回路上において，パケットがいずれのメッセージの場合でも，EN ターンは決して発生しない．それゆえに，拡張された迂回路により，EN ターンと NW ターンの連結は発生しない．

以上から，E2DM-Extend-Message は，デッドロックフリーである． □

3.5　通信性能の評価

3.5.1　評価方法

提案手法の通信性能を評価するために，C 言語でシミュレータを開発し，パケット転送を行ったときのレイテンシを測定した．レイテンシは，各パケットが出発地 S で生成されてから目的地 D に到着するまでにかかったサイクル数で定義される．本評価で対象とする Message，E2DM-Message，E2DM-Extend-Message は，VC を用いない手法である．そのため，2.3 節で説明したように，パケットは，ルータ内で競合が発生しなければ，4 サイクルで隣接ノード

表 3.1: シミュレーションのパラメータ

parameter	value
Network size ($N \times N$)	10×10, 20×20
Fault rate（f）	2, 4, 6, 8, 10 %
Input buffer depth	8 flits
Output buffer depth	1 flits
Packet length (L_p)	16, 32 flits
Total simulation length	50,000 cycles

へ転送される．故障ノードとパケットはランダムに発生させ，各測定において，1 ～ 5,000 サイクルまでは初期状態として，レイテンシの測定は行わないものとする．この試行を異なる 1,000 パターンの故障ノードの分布に対して実行し，平均レイテンシを算出した．また，公平な比較を行うために，パケットの生成パターンと故障ノードの発生パターンは，各手法で同一のパターンを適用する．シミュレーションパラメータを表 3.1 に示す．

各手法の通信性能を定量的に比較するために，手法 Ma の手法 Mb に対する最大レイテンシ削減率 $R(Ma, Mb)$ を次式で定義する．

$$R(Ma, Mb) = \max_{p} r_p(Ma, Mb).$$

ここで，$r_p(Ma, Mb)$ は，あるパケット生成率 p におけるレイテンシ削減率であり，次式で定義される．

$$r_p(Ma, Mb) = \frac{Lb - La}{Lb} \times 100.$$

ただし，La と Lb は，p における Ma と Mb のレイテンシである．

3.5.2　レイテンシ

本節では，様々なシステムを想定してレイテンシを比較する．そのために，以下の 3 つの条件でパラメータを設定して比較する．

条件 1: ネットワークサイズを 10×10（$N = 10$），パケット長を入力バッファの 2 倍の値（$L_p = 16$）に設定
条件 2: ネットワークが混雑しやすい場合での比較を行うために，$N = 10$，$L_p = 32$ に設定
条件 3: ネットワークが大きい場合での比較を行うために，$N = 20$，$L_p = 16$ に設定

以下に，上記の各条件で比較した結果を示す．

- 条件 1 での比較結果

  図 3.12 に，故障率 $f = 2\% \sim 10\%$ のシミュレーション結果を示す．グラフの横軸は，パケット生成率を表しており，縦軸は，平均レイテンシを表している．パケット生成率が高いほど，よりネットワークが混雑していることを表し，平均レイテンシが低いほど，より通信性能が高いことを表している．

  図 3.12 から，パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Message，E2DM-Message，E2DM-Extend-Message の順にレイテンシが低くなる．$f = 2\%$ のとき，Message と E2DM-Message の差はほとんどなく，E2DM-Extend-Message は，他の手法よりもレイテンシが低い．$f = 4\% \sim 8\%$ のとき，Message と E2DM-Message の差は開き始め，この場合においても，E2DM-Extend-Message は，他の手法よりもレイテンシが低い．$f = 10\%$ のとき，Message よりも E2DM-Message のレイテンシが低い．この場合，E2DM-Message と E2DM-Extend-Message のレイテンシの差は，他の故障率の場合よりも小さくなる．

  表 3.2 に，$f = 2\% \sim 10\%$ の場合の最大レイテンシ削減率 $R(Ma, Mb)$ を示す．表では，最大になったときのパケット生成率 p を括弧内に示している．Message と E2DM-Message を比較すると，E2DM-Message のほうが Message よりもレイテンシが低い．この理由を明らかにするために平均ホップ数を示す．$f = 2\%$ の場合，それぞれ 6.99，6.97 であり，$f = 10\%$ の場合，それぞれ 7.69，7.64 である．E2DM-Message では，ネットワークの端の故障領域をネットワークの新たに追加したリンクを使うことにより，故障を通過することが可能であるため，レイテンシが削減される．

  Message と E2DM-Extend-Message を比較すると，E2DM-Extend-Message は，E2DM-Message よりもさらに高い削減率である．この理由を明らかにするために E2DM-Extend-Message の平均ホップ数を示す．$f = 2\%$ の場合，6.93 であり，$f = 10\%$ の場合，7.61 である．この理由は，E2DM-Extend-Message では，物理的な迂回路と論理的な迂回路の追加により，平均ホップ数を削減することが可能であるため，レイテンシが削減される．

- 条件 2 での比較結果

  図 3.13 に，$f = 2\% \sim 10\%$ のシミュレーション結果を示す．条件 1 と同様に，パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Message，E2DM-Message，E2DM-Extend-Message の順にレイテンシが低くなる．パケット長を長くしたことにより，ネットワークが混雑しやすくなり，パケット生成率が低い場合でも，レイテンシが高くなる．

  表 3.3 に，$f = 2\% \sim 10\%$ の場合の最大レイテンシ削減率を示す．いずれの故障率においても，Message よりも E2DM-Message と E2DM-Extend-Message は，レイテンシ

表 3.2: 迂回路の拡張に基づく提案手法の最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

	f				
Ma, Mb	2%	4%	6%	8%	10%
E2DM-Message, Message	2%	6%	7%	14%	16%
	(0.40)	(0.35)	(0.40)	(0.30)	(0.30)
E2DM-Extend-Message, Message	35%	17%	15%	22%	20%
	(0.40)	(0.40)	(0.35)	(0.30)	(0.30)

表 3.3: 迂回路の拡張に基づく提案手法の最大レイテンシ削減率 ($N = 10$, $L_p = 32$)

	f				
Ma, Mb	2%	4%	6%	8%	10%
E2DM-Message, Message	1%	2%	6%	11%	11%
	(0.25)	(0.20)	(0.20)	(0.15)	(0.15)
E2DM-Extend-Message, Message	31%	17%	12%	18%	16%
	(0.20)	(0.20)	(0.20)	(0.15)	(0.15)

が低い．この理由は，条件 1 の評価結果でも述べたように，ホップ数が減少したからである．レイテンシ削減率が低下した理由は，前節のシミュレーションよりもパケット長が 2 倍になったことにより，ネットワークが混雑しやすくなったからである．以上のように，パケット長が変わった場合でも，レイテンシが削減できる結果が得られた．

- 条件 3 での比較結果

  図 3.14 に，$f = 2\% \sim 10\%$ シミュレーション結果を示す．パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Message，E2DM-Message，E2DM-Extend-Message の順にレイテンシが低くなる．

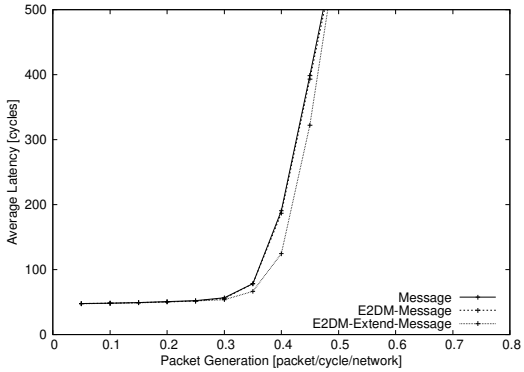
  表 3.4 に，$f = 2\% \sim 10\%$ の場合の最大レイテンシ削減率を示す．Message と E2DM-Message を比較すると，E2DM-Message は，$f = 2\%$ の場合，レイテンシ削減率は 3% であり，$f = 10\%$ の場合，8% である．Message と E2DM-Extend-Message を比較すると，E2DM-Extend-Message は，$f = 2\%$ の場合，レイテンシ削減率は 20% であり，$f = 10\%$ の場合，9% である．ネットワークサイズを変更した場合，故障率が高くなると故障ノード数が多くなり，故障領域が大きくなり，迂回経路長が長くなる．その場合でも，レイテンシが削減できる結果が得られた．

表 3.4: 迂回路の拡張に基づく提案手法の最大レイテンシ削減率 ($N = 20$, $L_p = 16$)

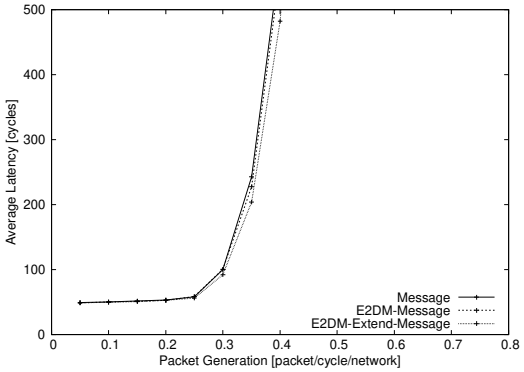
	f				
Ma, Mb	2%	4%	6%	8%	10%
E2DM-Message, Message	3%	8%	14%	13%	8%
	(0.65)	(0.60)	(0.40)	(0.40)	(0.35)
E2DM-Extend-Message, Message	20%	17%	23%	17%	9%
	(0.60)	(0.45)	(0.40)	(0.40)	(0.35)

以上の 3 つの比較から，E2DM-Message と E2DM-Extend-Message は，ネットワークが混雑しやすい場合やネットワークサイズが大きい場合でも，レイテンシを削減することが可能な手法であることを明らかにした．

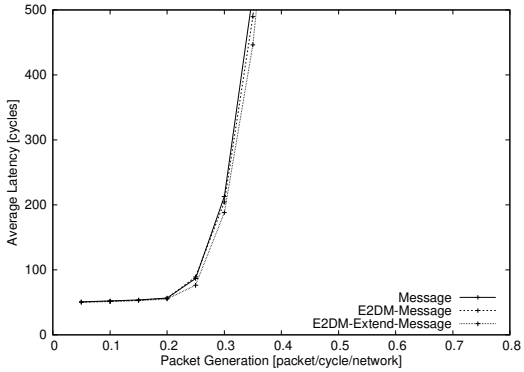
ここで，本評価で得られた値の信頼度について補足しておく．以上の各手法の平均レイテンシは，95% 信頼度の信頼区間に含まれていることを確認している．加えて，最大レイテンシ削減率が 15% 以上を達成する場合については，手法 Ma と手法 Mb の信頼区間に重なりはなく，Ma の上限と Mb の下限の差が開いていることも確認している．



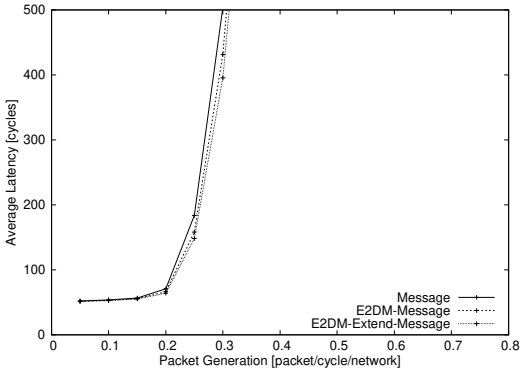
(a)　f=2%



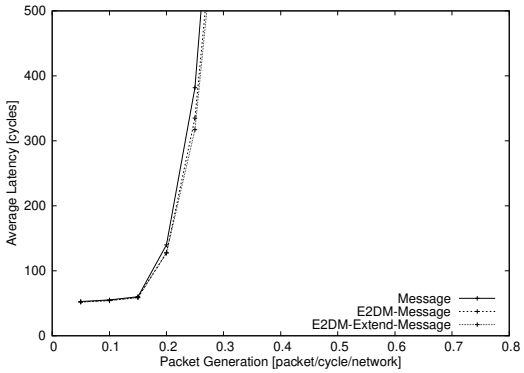
(b)　f=4%



(c)　f=6%

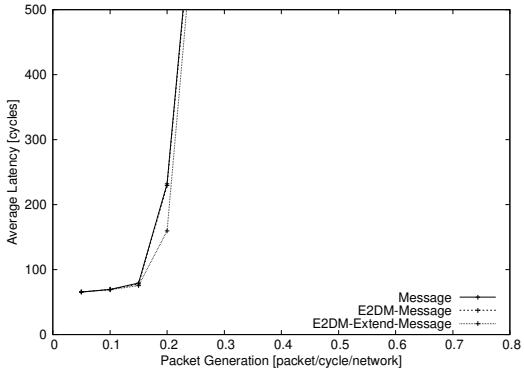


(d)　f=8%

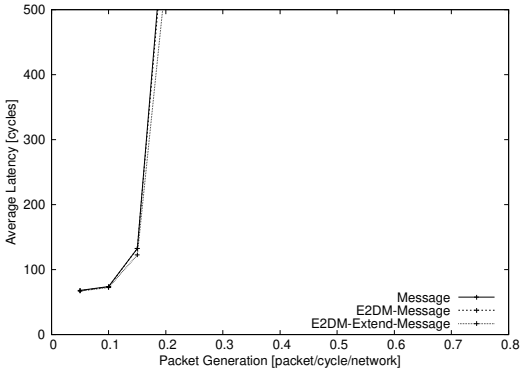


(e)　f=10%

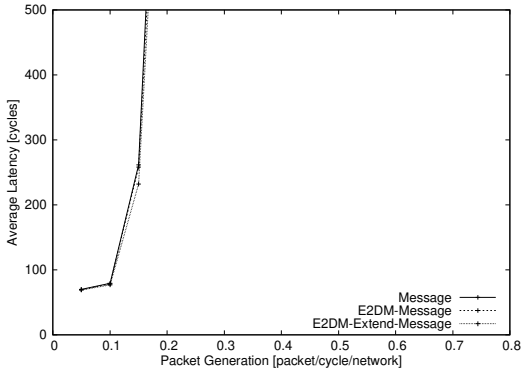
図 3.12: 迂回路の拡張に基づく決定的な耐故障ルーティング法の平均レイテンシ ($N = 10$, $L_p = 16$)



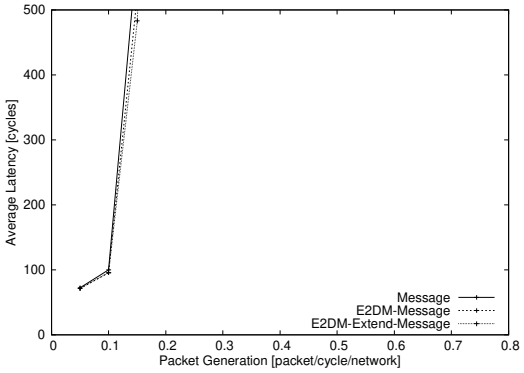
(a)　f=2%



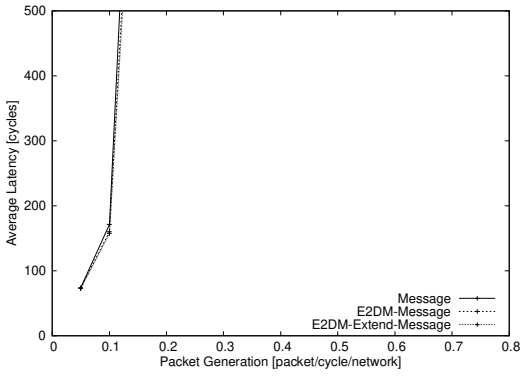
(b)　f=4%



(c)　f=6%

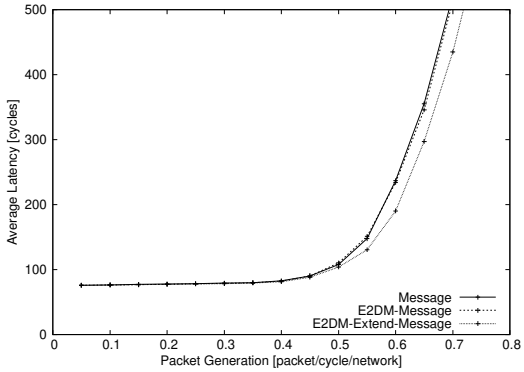


(d)　f=8%

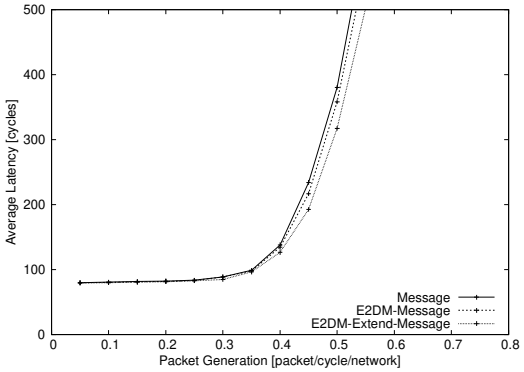


(e)　f=10%

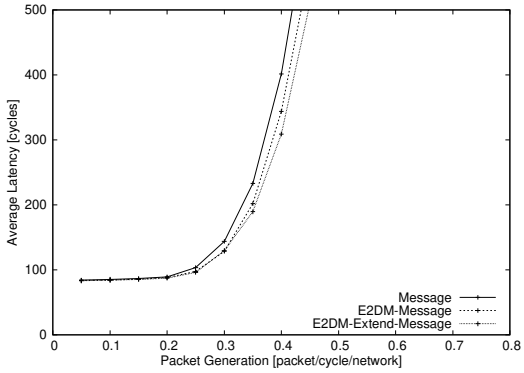
図 3.13: 迂回路の拡張に基づく決定的な耐故障ルーティング法の平均レイテンシ ($N = 10$, $L_p = 32$)



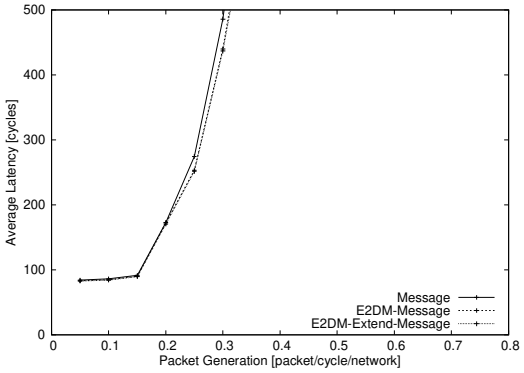
(a)　f=2%



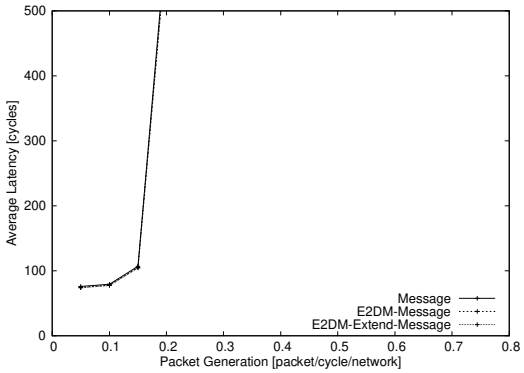
(b)　f=4%



(c)　f=6%



(d)　f=8%



(e)　f=10%

図 3.14: 迂回路の拡張に基づく決定的な耐故障ルーティング法の平均レイテンシ ($N = 20$, $L_p = 16$)

3.6　回路設計

3.6.1　ルータの設計

本節では，本章で提案した耐故障ルーティング法を回路実装するために，必要なルータの各ユニットについて説明する．実装するルータでは，16bit のフリットを想定し，リンク幅を 16bit とする．また，実装する耐故障ルーティング法は VC を使用しない手法であるため，4 サイクルで動作するルータとして実装する．以下の各節で，各ユニットの構成と動作を述べる．

3.6.2　入力ユニット

設計した入力ユニットの構成を図 3.15 に示す．入力ユニットは，8 フリットを格納するバッファ（buffer），フリットに対する処理を制御するためのステートマシン（State machine），ステートの情報を格納する 3bit のステートレジスタ（State），フリットの移動先が格納される 3bit のルートレジスタ（Route）で構成されている．

ステートマシンのステートとして，以下の 3 つの状態を持つ [8].

- I（Idle）ステート: フリットの到着を待つステート
- R（Route）ステート: フリットの次の移動先ルータを決定するステート
- A（Active）ステート: フリットをクロスバスイッチに送信するステート
  この A ステートには，サブステートとして SA と ST がある．SA は出力ユニットに移動するためのクロスバスイッチの割り当てを行うステートであり，ST はクロスバスイッチにフリットを送信するステートである．

これらのステートで処理を制御するのは，次々に送信されてくるフリットを，パイプライン式に処理するためである．

以下に，入力ユニットの動作を述べる．

- I ステート
  ルータは隣接ルータからヘッドフリットを受信すると，ヘッドフリットを入力ユニットのバッファに格納する．同時に，ルータ内のルーティング回路にヘッドフリットの情報を送信する．ヘッドフリットを入力バッファに格納すると，ステートを I から R に変更する．
- R ステート
  ルーティング回路から次の移動先である出力ポートの情報を受信し，ルートレジスタに格納する．その後に，ステートを R から A に変更する．

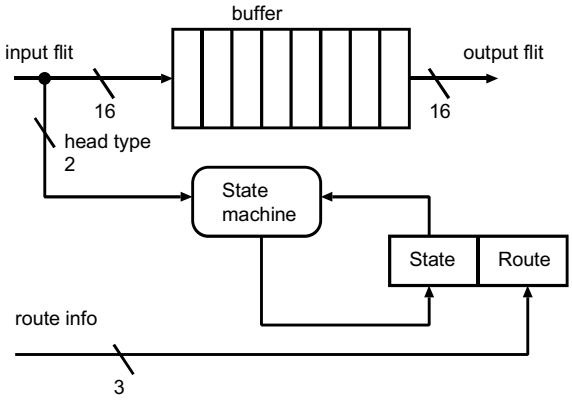


図 3.15: 入力ユニットの構成

- A ステート
  A ステートになると，サブステートが動作する．サブステートが SA となり，このとき，ルートレジスタ内の出力ポートの情報をスイッチアロケータに送信する．次に，サブステートが ST になると，入力バッファに格納されているフリットを出力ユニット内の出力バッファに送信する．

あるパケットのヘッドフリットが処理されている間に，後続のボディフリットも入力ユニットに受信され，それらのフリットはパイプラインで同様に処理される．

3.6.3　出力ユニット

設計した出力ユニットの構成を図 3.16 に示す．出力ユニットは，1 フリットを格納するためのバッファ（buffer），フリットに対する処理を制御するためのステートマシン（State machine），ステートの情報を格納する 3bit のステートレジスタ（State）で構成される．

ステートマシンのステートとして，以下の 2 つの状態を持つ．

- I ステート: フリットの到着を待つステート
- A ステート: フリットを隣接ルータへ送信するステート

以下に，出力ユニットの動作を述べる．

- I ステート
  入力ユニットから出力ユニットへ送信フラグ（入力フラグ）とフリットを受信する．フリットを出力バッファに格納し，送信フラグを出力ユニットのステートマシンに入力する．このとき，ステートを I から A に変更する．

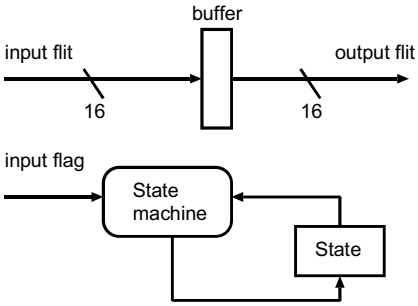


図 3.16: 出力ユニットの構成

- A ステート
  移動方向の隣接ルータの入力バッファに空きがある場合，フリットを送信する．空きがない場合，空きができるまでフリットを送信せず，待機させる．フリットの送信が完了した場合，ステートを A から I に変更する．

3.6.4　スイッチアロケータ

スイッチアロケータでは，クロスバスイッチの制御をおこなうため，ある 1 つの出力ポートに接続する入力ユニットが複数ある場合，いずれか 1 つを選択する必要がある．そのため，複数の入力ユニットから同じ出力ポートを要求する競合が発生した場合，スイッチアロケータで競合の解決も行い，複数の入力ユニットから 1 つの入力ユニットを決定する．設計するスイッチアロケータは，入力ユニットの決定方法に優先度付きのラウンドロビン方式を用いて，同じユニットが選ばれ続けないように順に決定する．

設計したスイッチアロケータの構成を図 3.17 に示す．スイッチアロケータは，ラウンドロビンで競合を解決するためのプライオリティエンコーダ（PE）群と前回選択されたポート情報を格納するレジスタ（history）で構成されている．図 3.17 は，東の出力ポートに対する回路例を示したものであり，出力ユニットの数だけ存在する．PE は，各入力ポートに優先度が付いている．例えば，PE1 の場合，東のポートが優先度が高いため 1 になっており，コアへのポートが優先度が低いため 5 になっている．

以下に，図 3.17 を例に，スイッチアロケータの動作を述べる．各入力ユニットから，接続要求として，出力ポートの情報が送信される．その情報が，東の出力ポートの場合，High の信号がフラグとして PE 群に入力される．このとき，競合が発生しなければ，PE からエンコードされた入力ポートの情報をクロスバスイッチに送信する．

競合が発生した場合には，同じポートが選ばれ続けないように複数の PE を切り替える回路

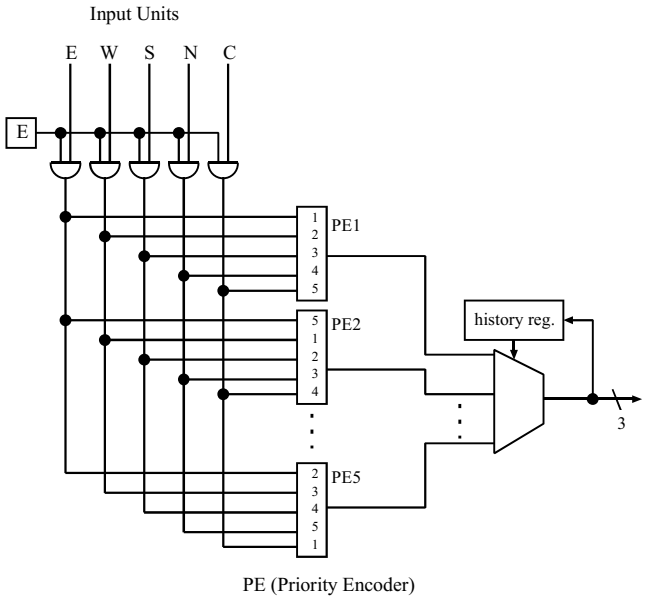


図 3.17: スイッチアロケータの構成例

構成とした．PE が複数個存在している理由は，PE が 1 つだけの場合，1 つのポートが優先され続けるため，複数用意して PE を切り替えていくことで選択される入力ポートが偏らないようにするためである．PE は，各入力ユニットに対応する数だけあり，任意の入力ユニットが一番優先されるように優先順位がつけられている．例えば，東の入力ユニットを優先する PE には，東，西，南，北，コアの順に優先順位が付いている．この PE では，この優先順位に従って一つの入力ポートを選択し，その入力ポートの情報をエンコードして出力する．南の入力ユニットを優先する PE では，南，北，コア，東，西の順に優先順位が付いている．history レジスタには，直前に選択された入力ユニットの情報が格納されている．例えば，現在，history レジスタにより東を優先する PE が選択されており，次の SA の処理で，西を優先する PE を選択するために，history レジスタに西を優先する PE の情報を格納する．以上のような，優先度付きのラウンドロビン方式により，各ポートに対してある程度公平に競合を解決する．

3.6.5　クロスバスイッチ

図 3.18 に設計したクロスバスイッチの構成を示す．クロスバスイッチは，スイッチアロケータから接続情報を受け取り，入力ユニットと出力ユニットのポートを接続する．厳密な意味では，クロスバスイッチは交点にスイッチを置き，スイッチを切り替えて接続を変更するものである．しかし，デジタル回路では，図 3.18 のように，マルチプレクサを用いて機能的に等価な

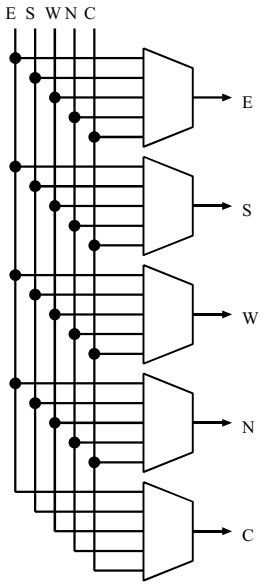


図 3.18: クロスバスイッチの構成

回路で実装するのが一般的である．

スイッチアロケータから接続する入力ポートと出力ポートの情報を受信すると，クロスバスイッチは，その情報に基づきスイッチを切り替える．

3.6.6　ルーティング回路

ルーティング回路は，入力ユニットから送信されてきたフリットを受け取り，その情報から次の移動先（出力ポート）を決定する．ルーティング回路では，入力ユニットからヘッドフリットを受け取り，ヘッドフリットの目的地のノードの座標とメッセージから出力ポートを決定する．

ルーティングアルゴリズムは，耐故障ルーティング法ごとに異なるため以下で述べる．

- Message
  実装したルーティング法は図 3.2 に示した通りである．故障領域に対する迂回路によりアルゴリズムが異なるため，適切な迂回路を選択する必要がある．そのために，北東，北西，南東，南西にある迂回路の情報を保存するための 2bit のレジスタが 4 つある．他に，隣接のルータ（東西南北）が故障かどうかのフラグを保存するための 1bit のレジスタが 4 つある．
- E2DM-Message

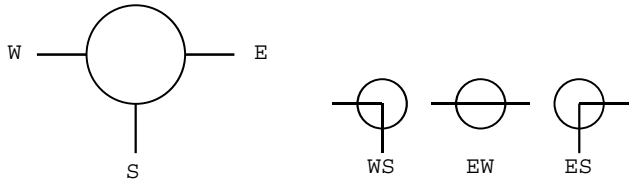


図 3.19: 拡張スイッチの構成

実装したルーティング法はアルゴリズム 3.1 に示した通りである．適切な迂回路を選択するための構成は，上記の Message で述べた構成と同じである．

3.6.7　拡張スイッチ

図 3.19 に E2DM-Message のための設計したスイッチの構成を示す．東（E），西（W），南（S）方向の 3 つの入出力ポートのうち，2 つのポートを接続する．図 3.19 に示すように，WS，EW，ES の 3 つの状態を取り得る．接続先を自動的に切り替えるため，スイッチの状態を内部のレジスタに保存する構成とした．

3.7　回路量の評価

3.7.1　回路量の比較

拡張した回路の回路量とルーティングアルゴリズムの回路量を明らかにするために，3.6 節で示したルータ回路と迂回路の拡張に基づく基本的な手法の E2DM-Message の回路設計を行い，動作確認および回路量の評価を行う．統合開発環境として Xilinx 社の Vivado 2013.4 を用いて，Verilog HDL で回路設計を行った．Vertex7 の FPGA デバイス xc7vx485tffg1761-2 を対象に論理合成を行った．

ルータ内の各モジュールの回路量を表 3.5 に示す．表 3.5 に示す値は，対象にした FPGA が消費した LUT（ルックアップテーブル）と register を示している．さらに，units は各ユニットの個数を示している．LUT とは，論理回路を作成する基本素子である．register は，入力や出力するデータを保存するためのものである．ルーティング回路の部分については，Message と E2DM-Message では，ルーティングルールが異なるため個別に示している．拡張スイッチの項目については，E2DM-Message で使用するモジュールであるため，Message では必要ない．Message で用いるルータは，5 つの入出力ユニットとルーティング回路，1 つのスイッチアロケータとクロスバスイッチで構成されている．Message のルータでは，ルーティング回路

表 3.5: 各モジュールの回路量

	LUT	register	# of units
Input unit	76	17	5
Output unit	4	17	5
Route circuit (Message)	229	53	5
Switch allocator	105	15	1
Crossbar-switch	144	0	1
Route circuit (E2DM-Message)	105	49	5
Extend switch	56	2	-

表 3.6: ネットワークサイズ 5×5 の場合の回路量

	LUT	register
Message	44,850	11,250
E2DM-Message	31,142	10,814

の回路量（LUT）が約 64% を占めている．そのため，ルーティング回路はルータ内で回路量が一番多く必要であることがわかる．E2DM-Message のルータでは，ルーティング回路の回路量が Message のルータと比べて約 46% 削減された．なぜなら，E2DM-Message は，Message に比べてルーティングルールが簡略化されたからである．

5×5 の Message と E2DM-Message の回路量を表 3.6 に示す．この結果から，E2DM-Message では，リンクとスイッチを追加したにもかかわらず，LUT 数とレジスタ数は，それぞれ Message の約 69%，約 96% となった．この理由は，ネットワークの外周部を拡張することにより故障領域の場所に関わらず，1 つのルーティングアルゴリズムだけで耐故障ルーティングを行うことを可能にしたからである．

3.7.2　面積オーバヘッド

前節で，NoC 全体の回路量を比較して E2DM-Message の回路量が Message よりも少ないことを示したが，拡張に要する回路面積のオーバヘッドが大きければ，その部分が故障しやすくなってしまう．拡張部が故障すると，ネットワークの外周部の切り替えが行えなくなる．そこで，本節では拡張に要する回路に対する面積オーバヘッドを評価する．そのために，Message を用いたサイズ $N \times N$ の 2 次元メッシュ NoC（2DM-NoC）と E2DM-Message を用いた迂回路の拡張に基づく 2 次元メッシュ NoC（E2DM-NoC）に対して，レイアウト面積

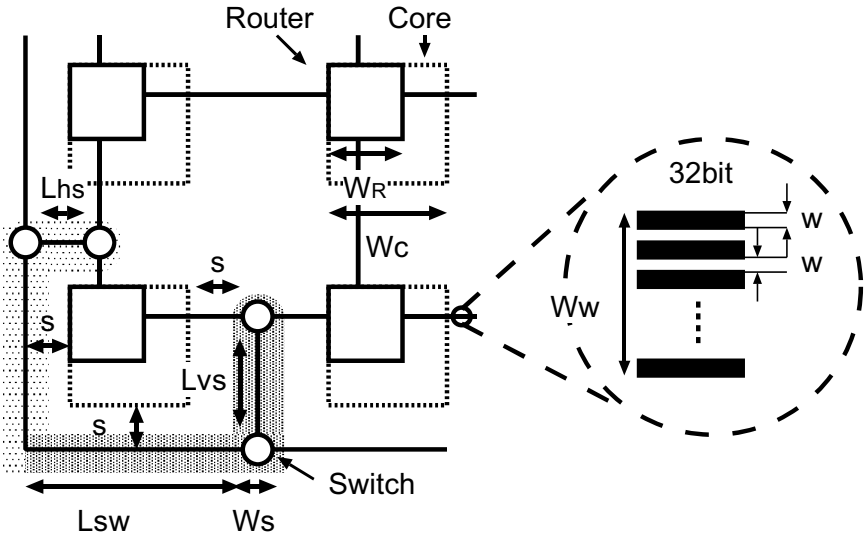


図 3.20: 各モジュールに対する長さと幅の定義

（回路面積）と面積オーバヘッドを推定する．

最初に，2DM-NoC のレイアウト面積 A_{2DM} を導出する．2DM-NoC の場合，拡張スイッチは不要である．図 3.20 に定めるように，コア幅を W_C，コア間のリンク長を s，リンク幅を W_W とすると，コアとリンクの個数から，A_{2DM} は次式で表わされる．

$$A_{\mathrm{2DM}} = N^2 W_C^2 + 2N(N-1)sW_w.$$

次に，E2DM-NoC のレイアウト面積 A_{E2DM} を導出する．外周と内部のスイッチ間のリンク長 L_{vs}（縦方向），L_{hs}（横方向）はそれぞれ次式で表わされる．

$$L_{\mathrm{vs}} = s + W_C - \frac{1}{2}(W_R + W_S),$$
$$L_{\mathrm{hs}} = s + \frac{1}{2}(W_R - W_S).$$

縦方向と横方向の長さは，図 3.20 に示すようにルータの位置によりそれぞれ異なる．縦方向のリンクの長さは，コアの長さが関係しており，横方向のリンクの長さは，コアの長さが関係していないため上記の式で表わされる．外周のスイッチ間のリンク長 L_{SW} は，次式のようになる．

$$L_{\mathrm{SW}} = 2s + W_C.$$

外周のスイッチ間のリンクの長さは，コアの長さとコア間のリンク長から上記の式で表わされる．E2DM-NoC の南側（図 3.20 に示される濃い点線）と西側（図 3.20 に示される薄い点線）

で，1 ルータあたりに追加するリンクとスイッチの面積をそれぞれ A_S と A_W すると，A_S，A_W は次式で表わされる．

$$A_S = W_w(L_{\mathrm{vs}} + L_{\mathrm{SW}}) + 2W_S^2,$$
$$A_W = W_w(L_{\mathrm{hs}} + L_{\mathrm{SW}}) + 2W_S^2.$$

A_{E2DM} は，2DM-NoC の面積 A'_{2DM}（コア間のリンク長を $2s + W_S$ として導出）と外周のスイッチとリンクの面積から，次式で表わされる．

$$A_{\mathrm{E2DM}} = A'_{\mathrm{2DM}} + 2(N-1)(A_S + A_W) + 4L_{\mathrm{SW}}.$$

評価のために，本報告では，w，s，W_C をそれぞれ次式の通り仮定する．

$$w = 28[nm],$$
$$s = 5W_w,$$
$$W_C = \alpha W_R(\alpha > 1).$$

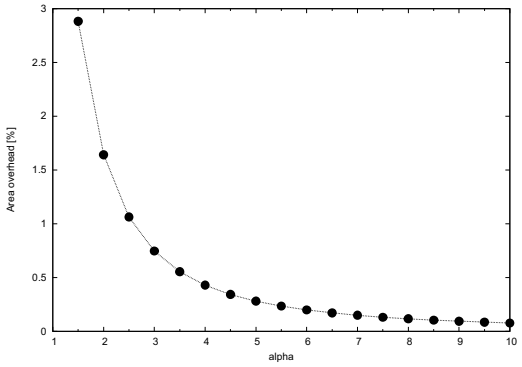
リンク幅 W_w は，16 ビットの双方向で伝送するリンクであるため，$W_w = (32+31)w$ となる．スイッチ幅 W_S は，表 3.5 より，ルータとスイッチの LUT 比が 1,174：56 であることから，$W_S = 0.22W_R$ とした．ここで，LUT 比のみを用いた理由は，比率の少ないレジスタ比を考慮しないことで，面積オーバヘッドの最悪値で比較するためである．

拡張によって生じる面積オーバヘッド A_{OH} を次式で定義する．

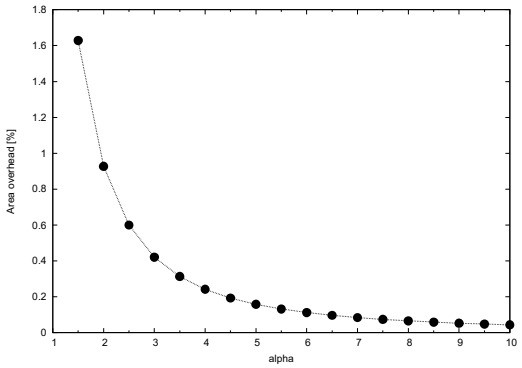
$$A_{\mathrm{OH}} = \left(\frac{A_{\mathrm{E2DM}}}{A_{\mathrm{2DM}}} - 1\right) \times 100\%.$$

図 3.21 (a) に，5×5 の E2DM-Message に対して，α を 1.5∼10 に変化させた場合の A_{OH} の値を示す．α はルータの線幅に対するコアの線幅の倍率を表すものである．コアのサイズが小さい場合（細粒度）は，低性能なコアであり，サイズが大きい場合（粗粒度）は，高性能なコアを想定することになる．この結果から，コアが細粒度から粗粒度のいずれの大きさの場合においても，A_{OH} は 3% 未満であり，拡張に必要なリンクやスイッチの面積オーバヘッドは少ないことがわかる．この結果より，E2DM-NoC の拡張部は，面積比で考えると故障しにくいことがわかる．

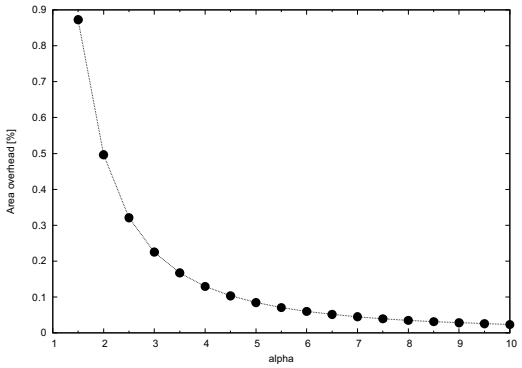
図 3.21 (b)，図 3.21 (c) に，それぞれ 10×10，20×20 の E2DM-NoC に対して，α を 1.5∼10 に変化させた場合の A_{OH} の値を示す．これらの結果から，コアが細粒度から粗粒度のいずれの大きさの場合においても，A_{OH} はそれぞれ 1.6，0.9% 未満であり，面積オーバヘッドは少ないことがわかる．この理由は，E2DM-NoC はネットワークの端にだけスイッチとリンクを追加しているため，ネットワークサイズが大きくなるにつれて，外周部の長さは長くなるが，ネットワーク内のノード数が多くなり相対的に面積オーバヘッドが小さくなるからである．



(a) ネットワークサイズ 5×5



(b) ネットワークサイズ 10×10



(c) ネットワークサイズ 20×20

図 3.21: 面積オーバヘッド

3.8　むすび

本章では，2 次元メッシュ NoC 向きの領域ベースの耐故障ルーティング法に対して，ルーティングアルゴリズムの各ルールに着目し，迂回ルールを簡略化するために，ネットワークの外周部を拡張するアーキテクチャを提案し，迂回路の拡張に基づく決定的な耐故障ルーティング法を提案した．具体的には，物理的な迂回路を利用する E2DM-Message と論理的な迂回路も利用する E2DM-Extend-Message の 2 つの手法を提案した．また，提案した 2 つの手法のデッドロックフリー性について述べた．さらに，本手法の有効性を明らかにするために，通信性能と回路量の評価を行った．

その結果, 故障率が低い場合は, 従来手法である Message と E2DM-Message のレイテンシは同程度あり，E2DM-Extend-Message のレイテンシは，2 つの手法よりも最大約 35% レイテンシが削減されることがわかった．また，故障率が高い場合は，Message よりも E2DM-Message や E2DM-Extend-Message のレイテンシが低く，E2DM-Message と E2DM-Extend-Message は同程度のレイテンシであることがわかった．さらに，拡張したアーキテクチャの回路量を明らかにするために，回路実装を行った．それにより，E2DM-Message を用いた 2 次元メッシュ NoC は，スイッチやリンクを追加したにも関わらず，Message を用いた 2 次元メッシュ NoC よりも約 69% 回路量が少ないことを明らかにした．また，追加したスイッチやリンクの面積オーバヘッドを明らかにするために，回路面積を推定した．その結果，面積オーバヘッドが最大でも約 3% であることを明らかにした．

第 4 章

故障ノードの通過に基づく決定的な耐故障ルーティング法

4.1 まえがき

3 章において，ネットワークの外周部にスイッチとリンクを追加した迂回路の拡張に基づく決定的な耐故障ルーティング法を提案し，少量の回路の追加で，大幅にルーティングアルゴリズムの回路量を削減可能であることを明らかにした．高性能なプロセッサを実現するためには，少ない回路量でルータを実装することに加えて，低い通信遅延の耐故障ルーティング法が求められる．3 章では，ルーティングアルゴリズムを簡略化することを主眼にしていたため，通信遅延は大幅に削減されなかった．

本章では，通信遅延の大幅な削減を目的とし，故障ノードの通過に基づく決定的な耐故障ルーティング法を提案する．そのために，まず，故障ノードの通過を可能にする 2 次元メッシュ NoC のアーキテクチャを提案する．このアーキテクチャ上で動作する XY ルーティングをベースにした耐故障ルーティング法（Passage-Y）を提案し，さらに，Passage-Y を発展させた手法（Passage-XY）を提案する．本手法の有効性を明らかにするために，通信性能と回路量の評価を行う．

4.2 節では，故障ノードを通過することが可能なアーキテクチャを提案する．4.3 節では，通過に基づく耐故障ルーティング法である Passage-Y を提案する．さらに，高い通信性能を実現するために Passage-Y を拡張した Passage-XY を提案する．4.4 節では，計算機シミュレーションにより，Passage-Y と Passage-XY の通信性能を評価する．4.5 節では，提案するアーキテクチャとルーティングアルゴリズムの回路量を明らかにする．4.6 節では，本章のまとめを述べる．

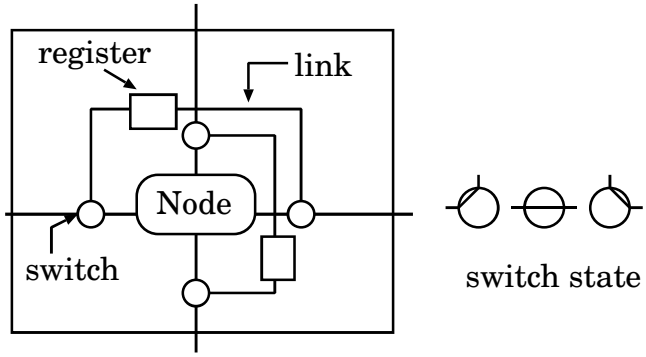


図 4.1: 通過可能な NoC のアーキテクチャ

4.2　故障ノードの通過を可能にする 2 次元メッシュ NoC の構成

本節では，ネットワークの内部を拡張し，故障ノードを通過することが可能なアーキテクチャを提案する．

そこで，ネットワークの内部にスイッチとリンクを配置したアーキテクチャを考える．提案するアーキテクチャを図 4.1 に示す．このアーキテクチャは，図 4.1 に示すように，各ノードの周辺にスイッチとリンク，レジスタを追加したものである．また，スイッチは，図に示す 3 つの状態を取り，あるノードが故障した場合，その故障情報から自動で一意に切り替わる．

図 4.2 に，故障ノードが発生した場合の例を示す．この場合，パケットが縦横方向に通過できるように，スイッチを切り替える．これにより，X 方向と Y 方向に通過することが可能になる．そのため，耐故障ルーティング法において，これまでの故障ノードを迂回する方法に加えて，故障ノードを通過することが可能になる．

4.3　故障ノードの通過に基づく決定的な提案手法

4.3.1　基本手法

4.2 節で説明した故障ノードの通過に基づくアーキテクチャにより，ルーティングルールとして故障ノードの迂回に加えて通過が可能になる．本節では，4.2 節で述べたアーキテクチャを対象にメッシュネットワークで一般的なルーティング法である XY ルーティングを基にした耐故障ルーティング法を提案する．本手法は，XY ルーティングを基本手法としているため，出発ノードと目的ノードが決まると経路が固定される決定的ルーティングである．

本手法のベースの手法である XY ルーティングのルーティングルールを説明する．XY ルー

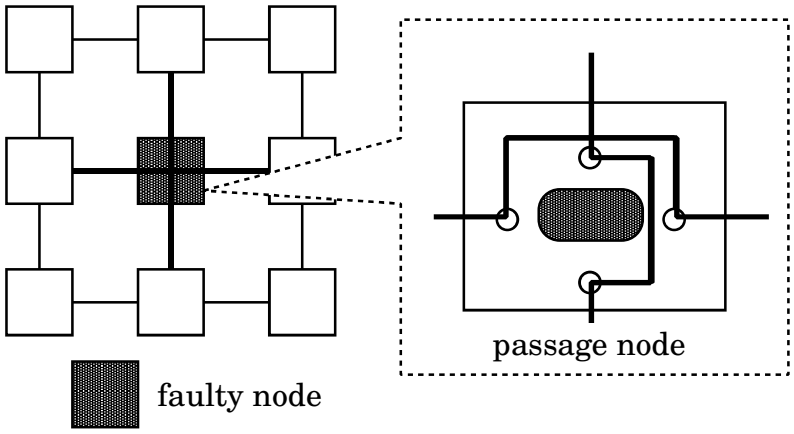


図 4.2: 故障ノードの通過のための再構成の例

ティングは，次元順ルーティングとも呼ばれており，ある次元の移動を行ったあとに次の次元の移動を行うものである．2 次元メッシュネットワークでは，まず，X 次元方向の移動を行う．その後，Y 次元の移動を行う．例えば，出発ノード $(0,0)$ から目的ノード $(4,5)$ に移動する場合，まず，X 座標を揃えるために $(4,0)$ まで X 方向の移動を行う．その後，Y 座標を揃えるために $(4,5)$ まで Y 方向の移動を行う．

4.3.2　Y 軸方向の通過に基づく耐故障ルーティング法: Passage-Y

通過に基づくアーキテクチャ上で XY ルーティングをベースにした耐故障ルーティングの通過と迂回のルールを説明する．以下で，本手法のルーティングアルゴリズムを述べる．

通過に基づくアーキテクチャに XY ルーティングを適用した場合について述べる．故障ノードが含まれるネットワークでのルーティング例を図 4.3 に示す．図 4.3 に示すように，故障ノードが (i,j) に存在する場合，パケットは，$(i+1,j)$ の出発地 S から (i,j') の目的地 D の X 座標を揃えるために，X 方向の移動が行われる（$j' \neq j$）．その際，パケットが故障ノードに直面すると，S から故障ノード (i,j) を通過し，$(i-1,j)$ の現在地 C に転送される．その後，C から D に転送するために，X 方向の移動が再度行われ，180 度ターンが発生する．それにより，D には永遠に到着しない．

この問題を防ぐために，XY ルーティングにルールの制限を追加する．そこで，本手法では，X 方向の通過を C と D の Y 座標が同じ場合のみ通過を許可し，それ以外の X 方向の通過を制限する．また，XY ルーティングにおいて，Y 方向の移動を行う場合，X 座標が揃っているため，Y 方向の通過は全て許可する．本論文では，本手法を Passage-Y と呼ぶ．

図 4.3 に示すような 180 度ターンを防ぐための X 方向の迂回ルールを述べる．図 4.4 に，X

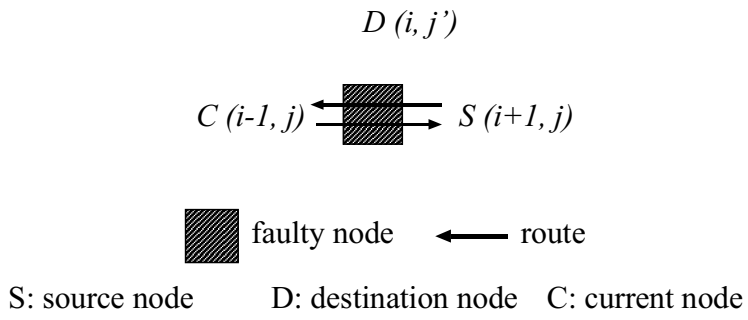


図 4.3: 180 度ターンの例

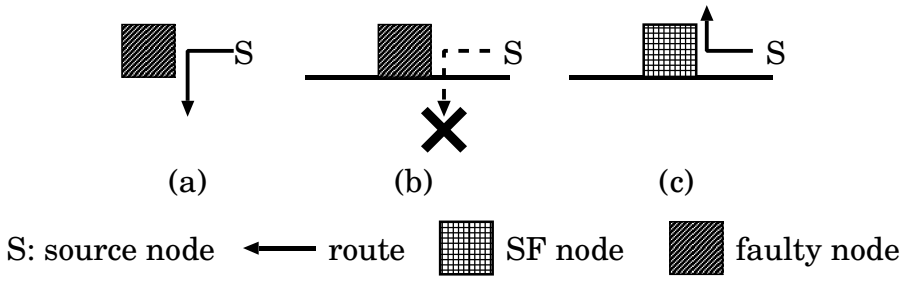


図 4.4: 制限をかけたルーティング例

方向の移動における故障ノードに対する迂回ルールを示す．図 4.4 (a) に示すように，X 方向の通過を制限するため，X 方向の移動の際に故障ノードに直面した場合，Passage-Y では，南方向に移動する．しかし，南方向の移動を許可すると，図 4.4 (b) に示すように，ネットワークの南端に故障ノードが存在する場合，パケットは，南方向に迂回できない．そこで，ネットワークの南端に接する故障ノードを以下で定義される SF ノードとして扱う．このノードにパケットが X 方向の移動の際に直面した場合，図 4.4 (c) に示すように，北方向に移動するようにする．

定義 4.1. 故障ノード (i, j) がネットワークの南端に接している場合 $(j = 0)$，(i, j) を SF ノードとする．

定義 4.2. ある SF ノード (i, j) の 8 近傍に存在する故障ノード (i', j') を SF ノードとする．ただし，$i - 1 \leq i' \leq i + 1$ かつ $i' \in X$ であり，$j - 1 \leq j' \leq j + 1$ かつ $j' \in Y$ である．

上記の定義 4.2 は新たな SF ノードが生成されなくなる定常状態になるまで，繰り返し適用される．

SF ノードを北方向に迂回するルールを追加した場合，図 4.5 に示すような無限大型（∞ 型）

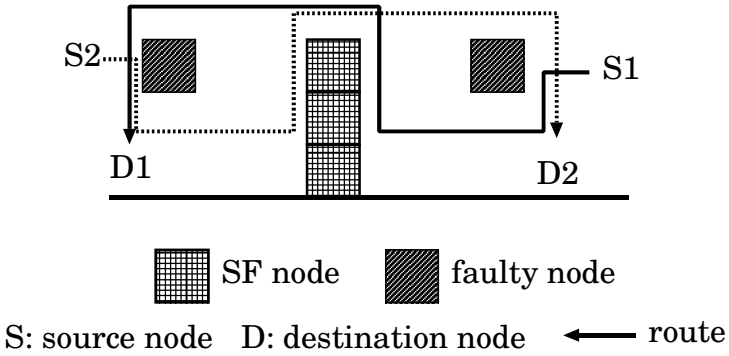


図 4.5: Passage-Y のデッドロック例

のデッドロックが発生する．図に示されるように，2 つのパケットはそれぞれ，故障ノードを南方向に迂回し，その後，SF ノードを北方向に迂回し，デッドロックが発生する．

このデッドロックを防ぐために，下記で定義する SF エリアを設定する．SF エリアを設定した後，SF エリア内の故障ノードを SF ノードに変更する．

定義 4.3. 定義 4.2 を繰り返して得られる最北の SF ノードの座標を (n_i, n_j) とする．このとき，$i \in X, 0 \leq j \leq n_j$ を満たす全てのノード (i, j) で構成されるエリアを SF エリアと呼び，このエリア内の全ての故障ノードを SF ノードとする．

定義 4.3 で新たに生成される SF ノードに対して，定義 4.2，4.3 を定常状態になるまで繰り返し適用する．SF エリアの作成例を図 4.6 に示す．図 4.6（a）では，まず，定義 4.1 により $(2, 0)$ の故障ノードを SF ノードに変更し，定義 4.2 により $(3, 1)$ の故障ノードも SF ノードに変更する．次に，定義 4.3 により $j = 1$ まで全ての X 座標のノードを SF エリアとして設定し，SF エリアに含まれる $(0, 1)$ の故障ノードを SF ノードに変更する．定義 4.2 により $(0, 2)$ の故障ノードを SF ノードに変更し，Y 座標が $j = 2$ までの全てのノードを SF エリアとして設定する．図 4.6（b）では，図 4.6（a）と同様に，定義 4.1 により $(2, 0)$ の故障ノードを SF ノードに変更し，定義 4.3 により Y 座標が $j = 0$ の全てのノードを SF エリアとして設定する．定義 4.2 により，$(2, 0)$ の SF ノードの 8 近傍に故障ノードが存在しないため，以降，SF エリアは形成しない．

このように，SF エリアを設定して故障ノードを SF ノードに変更することで，図 4.5 のデッドロックの問題が解決可能になる．SF エリアを導入した場合のルーティング例を図 4.7 に示す．図 4.7 に示すように，X 方向の移動を行うパケットは，全て北方向へ迂回する．これにより，図 4.5 のデッドロックは発生しなくなる．

以下で，ルーティングアルゴリズムの記述に必要な変数を定義する．

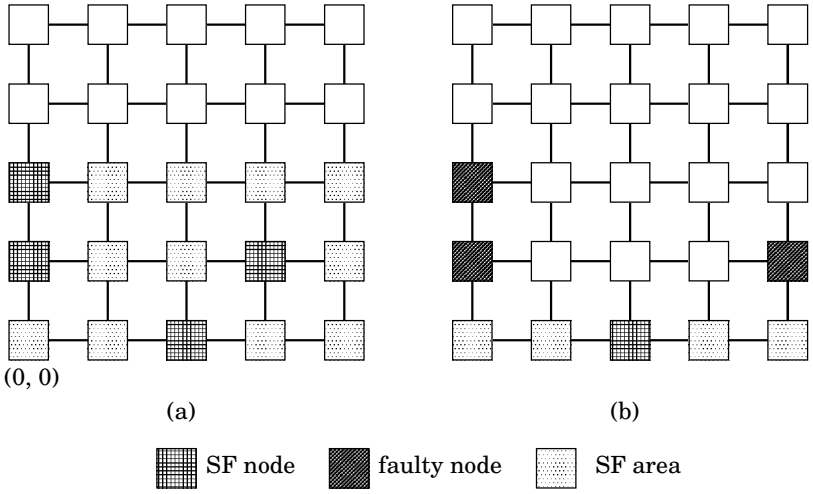


図 4.6: SF エリアの作成例

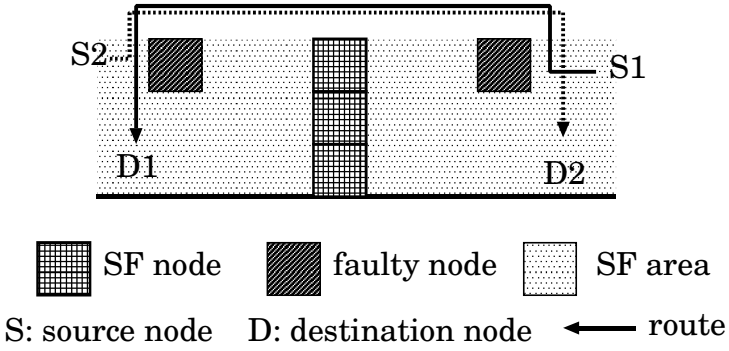


図 4.7: SF エリアを導入したルーティング例

定義 4.4. パケットの進行方向を Dir，次の移動先を $Next$ で表す．

定義 4.5. あるノードの東方向，西方向に隣接する故障ノードのフラグを F_E, F_W とする．各フラグは，隣接する東または西方向のノードが故障ノードであれば 1，そうでなければ 0 の値をとるものとする．

定義 4.6. あるノードの東西の隣接ノードが SF ノードか否かを表すフラグを，それぞれ，SF_E, SF_W とする．各フラグは，隣接する東または西方向のノードが SF ノードであれば 1，そうでなければ 0 の値をとるものとする．

アルゴリズム 4.1 に，Passage-Y のルーティングアルゴリズムを示す．アルゴリズム内の変

数は，3 章で定義した定義 3.5 と同様に，パケットの出発地 S，現在地 C，目的地 D の座標を，それぞれ，(S_x, S_y), (C_x, C_y), (D_x, D_y) で表す．このアルゴリズムでは，X 方向の移動ルールを X_Routing()，Y 方向の移動ルールを Y_Routing() で表している．Passage-Y は，ベースのルーティングアルゴリズムとして XY ルーティングを用いているため，最初，X_Routing() を行い，その後，Y_Routing() を行う．

図 4.8 に，Passage-Y のルーティングアルゴリズムの概略図を示す．図 4.8 に示すように，故障ノードに直面した場合，南方向に迂回し，SF ノードに直面した場合，北方向に迂回する．図 4.9 に，Passage-Y によるルーティング例を示す．パケット 1，2，3，4 は，故障ノードや SF ノードに直面すると，それぞれ，南方向や北方向に迂回して目的地まで転送される．例えば，パケット 1 と 3 は，故障に直面した際，南方向に迂回する．Passage-Y は，常に Y 方向の通過を許可しているため，両パケットとも故障ノードを通過する．また，パケット 2 と 4 は，SF ノードに直面した際，北方向に迂回して目的地まで転送される．以上のように，Passage-Y では，故障ノードの Y 方向の通過が可能であるため，迂回の回数を減らすことが可能である．加えて，従来の領域ベースの耐故障ルーティング法とは異なり，故障領域を作成する必要がないため，多くのノードを使用することが可能である．

アルゴリズム 4.1: Passage-Y Route

```
1  Passage−Y_Routing (C, D)
2  if (C_x == D_x && C_y == D_y)
3    consume the packet
4  elsif (C_x > D_x)
5    Next = X_Routing(C_x, D_x, WEST);
6  elsif (C_x < D_x)
7    Next = X_Routing(C_x, D_x, EAST);
8  elsif (C_x == D_x)
9    Next = Y_Routing(C_y, D_y);
10 endif
11
12 X_Routing (C_x, D_x, Dir)
13 if (C_y == D_y)
14      Next = Dir
15 elsif (F_Dir == 0)
16   if (SF_Dir == 1)
17      Next = NORTH;
18   else
19      Next = SOUTH;
20   endif
21 else
22   Next = Dir;
23 endif
24
25 Y_Routing (C_y, D_y)
26 if (D_y > C_y)
```

```
27    Next = NORTH;
28  else
29    Next = SOUTH;
30  endif
```

4.3.3　Passage-Y のデッドロックフリー性

Passage-Y のデッドロックフリー性を示す.

定理 4.1. ルーティングアルゴリズム Passage-Y は，デッドロックフリーである.

証明. Pasage-Y では，180 度ターンを許可していないため，図 2.5 に示したターンにより，時計回りと反時計回りのパケットの循環待ちが発生しないことを示す．Passage-Y では，故障ノードがない場合は，XY ルーティングと等価な動作をするため，NE, SW, NW, SE ターンは発生しない．故障ノードを迂回する際に，これらのターンが発生するが，その場合でも，パケットの循環待ちが発生しないことを示す.

時計回りの方向について考える．Passage-Y では，SF エリア内で SF ノードの迂回で NE ターンが，SF エリア外の故障ノードの迂回で SW ターンが発生する．しかし，2 つのターンは，同一エリア内では，決して発生しない．従って，時計回りのパケットの循環待ちは決して発生しない.

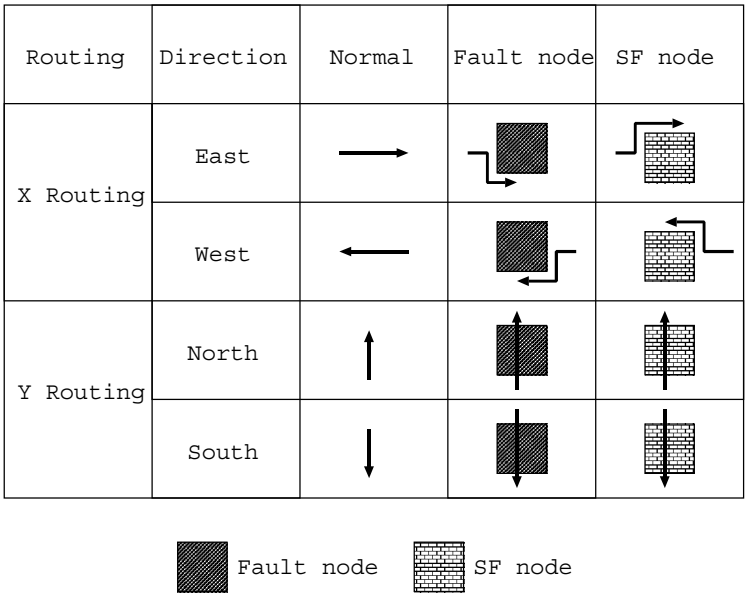


図 4.8: Passage-Y のルーティングアルゴリズムの概略図

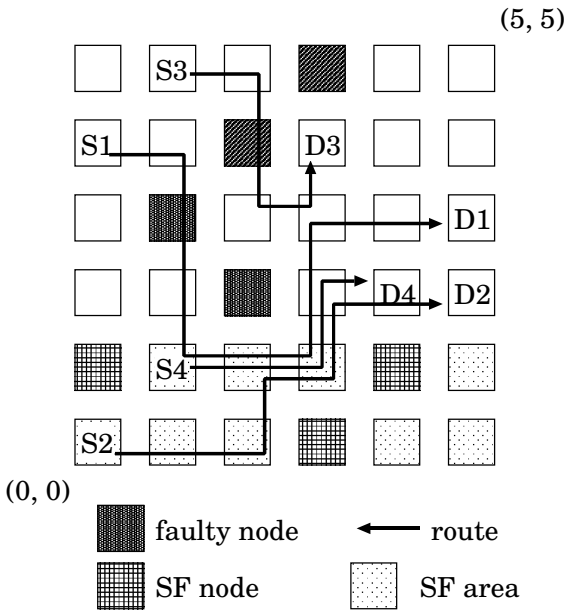


図 4.9: Passage-Y のルーティング例

反時計回りの方向についても，時計回りの方向と同じ方法で証明可能であるため，ここでは省略する．

以上から，Passage-Y は，デッドロックフリーである． □

4.3.4　X 軸と Y 軸方向の通過に基づく耐故障ルーティング法: Passage-XY

Passage-Y では，X 方向の通過は Y 座標が一致する場合のみでしか行われなく，Y 方向の通過は常に行われる．通過による経路長の削減効果を高めるために，Passage-Y に基づき，X 方向の通過をさらに可能にした耐故障ルーティング法を提案する．本論文では，本手法を Passage-XY と呼ぶ．

Passage-Y のルールに対して，単純に X 方向の通過を許可した場合について述べる．Passage-Y に X 方向の通過を許可した場合の例を図 4.10 (a) に示す．図 4.10 (a) に示すように，S_i から D_i に向かうパケット i の 4 つでデッドロックが発生する．パケット 1，パケット 3 は，それぞれ，SF ノードと故障ノードを X 方向に通過することにより，デッドロックが発生する．そこで，このデッドロックを回避するために，故障ノードのみに対して通過を許可する．その場合のルーティング例を図 4.10 (b) に示す．図 4.10 (b) においても，4 つのパケットが故障ノードを通過するため，デッドロックが発生する．このように，通過を許可するノードを制限する方法では，デッドロックの発生は避けられない．

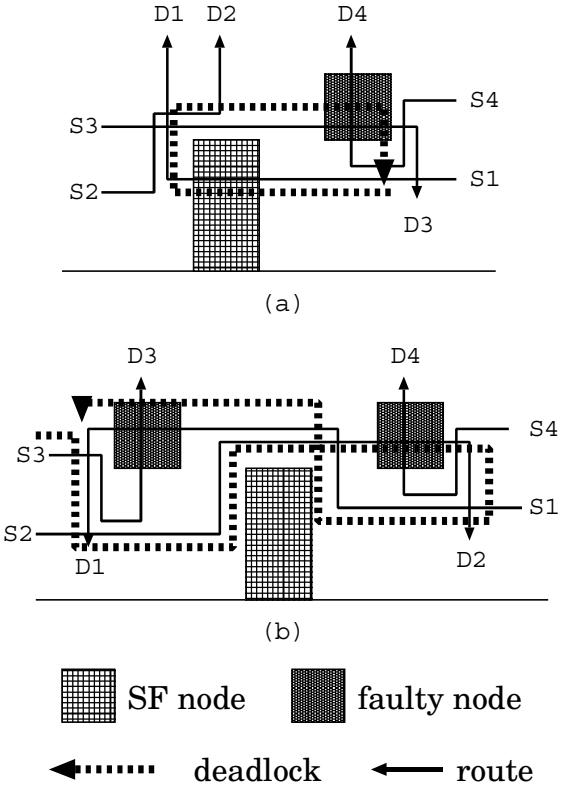


図 4.10: X 座標の通過を許可した場合のデッドロック

Passage-XY では，故障ノードと SF ノードの X 方向の通過を基本的に許可し，デッドロックを避けるために VC を 2 つ用いる．VC を用いてネットワークを仮想的に多重化することで，パケット転送を各ネットワークで行うことが可能になる．以下に，VC の使用ルールを示す．

- S において，
  - D が西にある場合: VC0 を使用する．
  - D が東にある場合: VC1 を使用する．
- VC0 から VC1，VC1 から VC0 への移動は禁止する．

X 方向の通過を許可しているため，故障ノードが連続して存在する場合，180 度ターンが発生する．図 4.11 に，故障ノードの通過する例を示す．図 4.11 に示すように，故障ノードの通過により，D の X 座標を超える．この場合，X 軸の座標を合わせるために，180 度ターンを繰り返し，永久に D に到達しない問題が発生する．これを防ぐために，Passage-XY では，D の X 座標を超えない場合のみ，X 方向の通過を許可する．

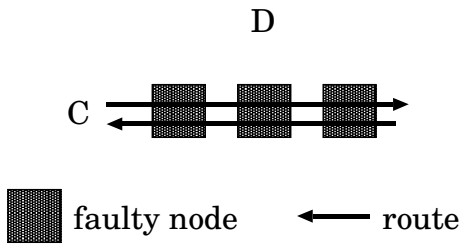


図 4.11: X 座標の通過を許可した場合の 180 度ターン

以下で，ルーティングアルゴリズムの記述に必要な変数を定義する．

定義 4.7. あるノード (i, j) の東方向，西方向に隣接する連続した故障ノードの数を F_E, F_W とする．

例えば，図 4.11 に示す C のノードの場合，$F_E = 3, F_W = 0$ となる．

アルゴリズム 4.2 に Passage-XY のルーティングアルゴリズムを示す．このアルゴリズムでは, Passage-Y から X 方向の通過を拡張したため, X 方向の移動ルールを Modify_X_Routing() とし，Y 方向の移動ルールを Y_Routing() で表している．また，Passage-XY は Passage-Y を拡張したものであるため，最初に，Modify_X_Routing() を行い，その後，Y_Routing() を行う．

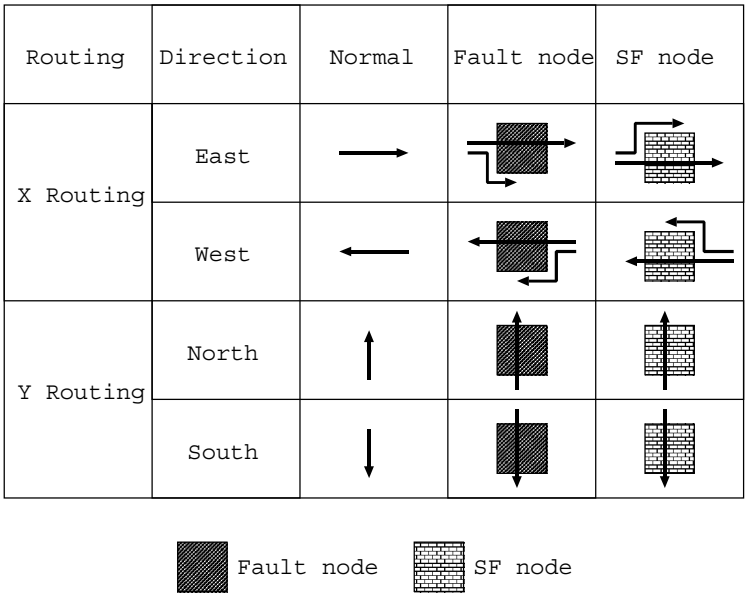


図 4.12: Passage-XY のルーティングアルゴリズムの概略図

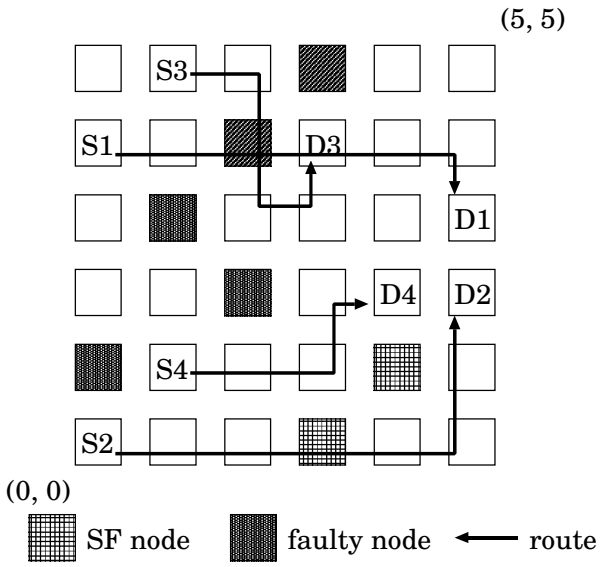


図 4.13: Passage-XY のルーティング例

図 4.12 に，Passage-XY のルーティングアルゴリズムの概略図を示す．図 4.12 に示すように，図 4.8 とは異なり，故障ノードと SF ノードの通過も可能である．図 4.13 に，Passage-XY によるルーティング例を示す．パケット 1，2 は，通過により C_x が D_x を超えないため，それぞれ，$(2,4)$ の故障ノード，$(0,3)$ の SF ノードを通過する．一方，パケット 3，4 は通過すると D を越えるため，通過することはできない．そのため，従来の Pasage-Y と同様に，それぞれ，$(3,5)$ の故障ノード南方向に，$(4,1)$ の SF ノードを北方向に迂回する．Passage-XY では，X 方向も通過することが可能なため，図 4.11 の例と比べて，ホップ数の少ないルーティングが可能である．また，VC を用いてネットワークごとに移動方向を決定するため，定義 4.3 以降の SF エリアを作成する必要がないことも補足しておく．

アルゴリズム 4.2: Passage-XY Route

```
1  Passage−XY_Routing (C, D)
2  if (C_x == D_x && C_y == D_y)
3     consume the packet
4  elsif (C_x > D_x)
5     Next = Modify_X_Routing(C_x, D_x, WEST);
6  elsif (C_x < D_x)
7     Next = Modify_X_Routing(C_x, D_x, EAST);
8  elsif (C_x == D_x)
9     Next = Y_Routing(C_y, D_y);
10 endif
11
12 Modify_X_Routing (C_x, D_x, Dir)
```

```
13 if (F_Dir > 0)
14   if ((C_x + F_Dir < D_x && Dir == WEST) ||
15       (C_x - F_Dir > D_x && Dir == EAST))
16     Next = Dir;
17   elsif (SF_Dir == 1)
18     Next = NORTH;
19   else
20     Next = SOUTH;
21   endif
22 else
23   Next = Dir;
24 endif
```

4.3.5　Passage-XY のデッドロックフリー性

Passage-XY のデッドロックフリー性を示す.

定理 4.2. ルーティングアルゴリズム Passage-XY は，デッドロックフリーである.

証明. Pasage-XY では，180 度ターンを許可していないため，図 2.5 に示したターンにより，時計回りと反時計回りのパケットの循環待ちが発生しないことを示す．Passage-XY では，故障ノードがない場合は，XY ルーティングと等価な動作をするため，NE, SW, NW, SE ターンは発生しない．故障ノードを迂回する際に，これらのターンが発生するが，その場合でも，パケットの循環待ちが発生しないことを示す.

時計回りの方向について考える．Passage-XY では，SF ノードの迂回で NE ターンが，SF ではない故障ノードの迂回で SW ターンが発生する．4.3.4 節で示した，VC の使用ルールから，VC0 のネットワークでは SW ターンが発生するが，NE ターンは決して発生しない．同様に，VC1 のネットワークでは NE ターンが発生するが，SW ターンは決して発生しない．VC0 と VC1 の間ではパケットの移動は禁止されているため，片方で発生したターンが，もう片方に影響することもない．従って，各 VC のネットワークで，時計回りのパケットの循環待ちは決して発生しない.

反時計回りの方向についても，時計回りの方向と同じ方法で証明可能であるため，ここでは省略する.

以上から，Passage-XY は，デッドロックフリーである. □

表 4.1: 性能比較で用いたルーティング法

手法	表記	使用 VC 数
トポロジ非依存の手法 [32]	Tree	1
Boppana らの手法 [26]	Fcube4	4
Fukushima らの手法 [29]	Position	1
Passage-Y	Passage-Y	1
Passage-XY	Passage-XY	2

4.4　通信性能の評価

4.4.1　評価方法

提案手法の通信性能を評価するために，C 言語でシミュレータを開発し，パケット転送を行ったときのレイテンシを測定した．性能比較で用いた手法を表 4.1 に示す．Tree は，故障を含むネットワーク上に論理ツリーネットワークを構成し，そのネットワーク上でルーティングを行う手法である．Fcube4 と Position は，矩形の故障領域を作成し，その周りを迂回する手法である．3.5.1 節の表 3.1 に示したパラメータと同様のパラメータでシミュレーションを行う．ただし，パケットは，ルータ内で競合が発生しない場合，2.3 節で述べたように，VC を用いない手法（表 4.1 で使用 VC 数が 1 の手法）では，4 サイクルで転送され，VC が複数の手法では，5 サイクルで隣接ノードへ転送されるものとする．

4.4.2　レイテンシ

本節では，3.5.2 節と同様に，様々な状況を想定した評価を行う．以下に，3.5.2 節と同様の各条件で比較した結果を示す．

- 条件 1 での比較結果
  図 4.14 に，故障率 $f = 2\% \sim 10\%$ の場合のシミュレーション結果を示す．パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Tree, Position, Fcube4, Passage-Y, Passage-XY の順にレイテンシが低くなる．Tree は，ツリーのルートノード付近で大きく混雑するため，最もレイテンシが高い．Position と Fcube4 は，矩形の故障領域を作成して迂回するため，その周辺で混雑が発生しやすくなる．D の位置に応じて迂回方向を変更できるため，Fcube4 のほうが

表 4.2: Passage-XY に対する最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

	f				
Mb	2%	4%	6%	8%	10%
Tree	99%	99%	99%	99%	99%
	(0.80)	(0.75)	(0.75)	(0.70)	(0.70)
Position	97%	98%	98%	98%	98%
	(0.95)	(0.85)	(0.80)	(0.80)	(0.75)
Fcube4	96%	97%	98%	98%	98%
	(1.00)	(0.95)	(0.90)	(0.90)	(0.75)
Passage-Y	93%	95%	96%	97%	97%
	(1.00)	(0.95)	(0.90)	(0.85)	(0.80)

レイテンシは低い．一方で，Passage-Y, Passage-XY は，故障領域は作成せずに，故障ノードの通過を許可するため，さらにレイテンシが低く，X と Y の両方向で通過を許可する Passage-XY が，最もレイテンシが低い．また，各手法とも，f が増加するにつれて，レイテンシが増加する．これは，故障ノード数が増加し，迂回により S から D までの経路長が増加するからである．

表 4.2 に，Ma を Passage-XY としたときの $f = 2\% \sim 10\%$ の最大レイテンシ削減率を示す．Passage-XY は，他の手法と比較して，レイテンシ削減率は 93% 以上であり，通過によるレイテンシの削減効果が非常に大きいことがわかる．

- 条件 2 での比較結果

  図 4.15 に，$f = 2\% \sim 10\%$ のシミュレーション結果を示す．条件 1 と同様に，パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Tree, Position, Fcube4, Passage-Y, Passage-XY の順にレイテンシが低くなる．パケット長を長くしたことにより，全体的にネットワークが混雑しやすくなり，パケット生成率が低い場合でも，各手法のレイテンシは高くなる．

  表 4.3 に，Ma を Passage-XY としたときの $f = 2\% \sim 10\%$ の最大レイテンシ削減率を示す．Passage-XY は，他の手法と比較して，レイテンシ削減率は 88% 以上であり，通過によるレイテンシの削減効果が非常に大きいことがわかる．

- 条件 3 での比較結果

  図 4.16 に，$f = 2\% \sim 10\%$ のシミュレーション結果を示す．条件 1 と同様に，パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くな

表 4.3: Passage-XY に対する最大レイテンシ削減率 ($N = 10$, $L_p = 32$)

	f				
Mb	2%	4%	6%	8%	10%
Tree	99%	99%	98%	98%	98%
	(0.35)	(0.35)	(0.30)	(0.30)	(0.30)
Position	96%	97%	97%	97%	98%
	(0.45)	(0.40)	(0.35)	(0.35)	(0.35)
Fcube4	93%	96%	97%	97%	98%
	(0.45)	(0.45)	(0.40)	(0.40)	(0.35)
Passage-Y	88%	93%	94%	95%	96%
	(0.45)	(0.45)	(0.40)	(0.40)	(0.40)

表 4.4: Passage-XY に対する最大レイテンシ削減率 ($N = 20$, $L_p = 16$)

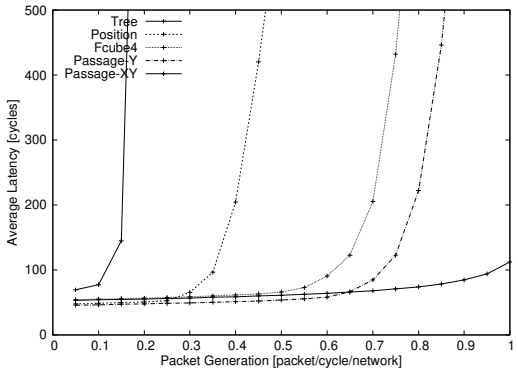
	f				
Mb	2%	4%	6%	8%	10%
Tree	98%	98%	98%	98%	98%
	(1.0)	(1.0)	(1.0)	(1.0)	(1.0)
Position	94%	95%	96%	96%	97%
	(1.0)	(1.0)	(1.0)	(1.0)	(1.0)
Fcube4	92%	96%	97%	95%	97%
	(1.5)	(1.5)	(1.5)	(1.0)	(1.0)
Passage-Y	93%	95%	96%	88%	92%
	(2.0)	(2.0)	(1.9)	(1.0)	(1.0)

ると，Tree, Position, Fcube4, Passage-Y, Passage-XY の順にレイテンシが低くなる．表 4.4 に，Ma を Passage-XY としたときの $f = 2\% \sim 10\%$ の最大レイテンシ削減率を示す．Passage-XY は，他の手法と比較して，レイテンシ削減率は 88% 以上であり，通過によるレイテンシの削減効果が非常に大きいことがわかる．

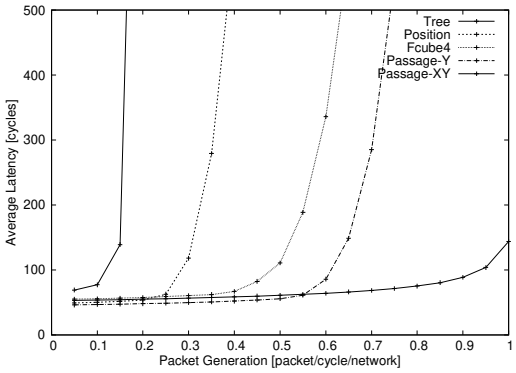
以上の 3 つの比較から，Passage-Y と Passage-XY は，ネットワークが混雑しやすい場合や大規模なネットワークの場合のどんな場合においても，レイテンシの削減が可能な手法である

ことを明らかにした．

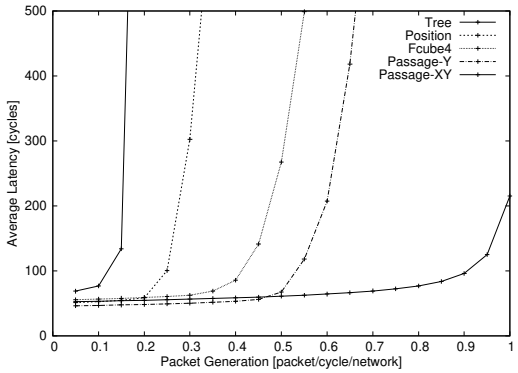
ここで，3.5.2 節のレイテンシの評価と同様に，各手法の平均レイテンシは，95% の信頼度の信頼区間に含まれていることを確認していることを補足する．



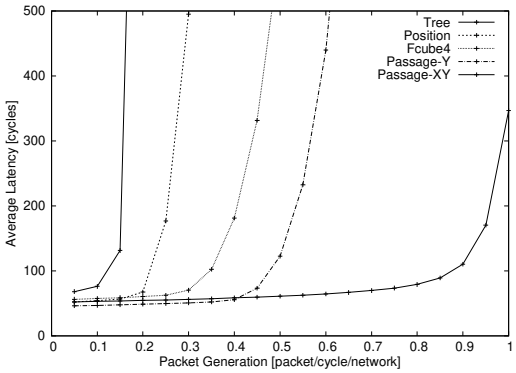
(a)　f=2%



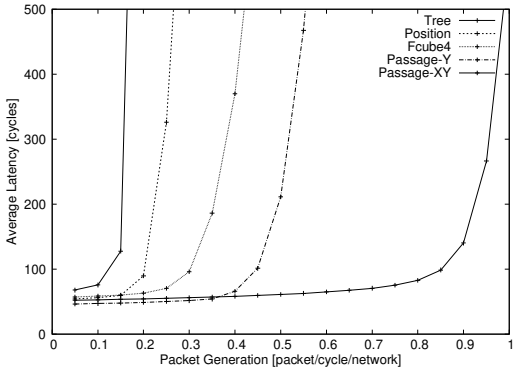
(b)　f=4%



(c)　f=6%

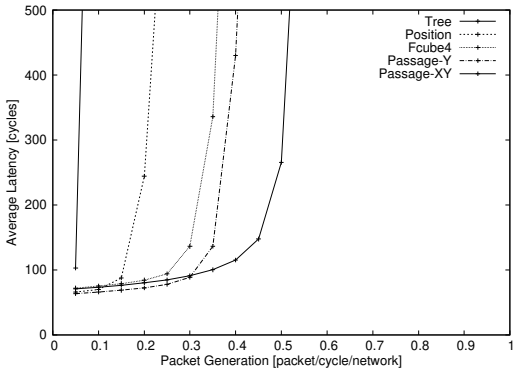


(d)　f=8%

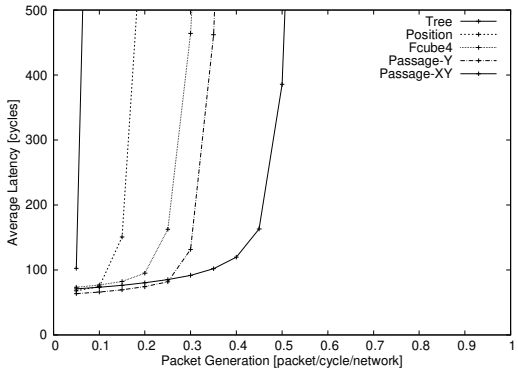


(e)　f=10%

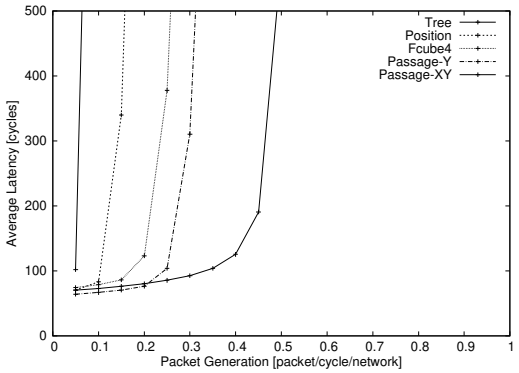
図 4.14: 故障ノードの通過に基づく決定的な提案手法の平均レイテンシ ($N = 10$, $L_p = 16$)



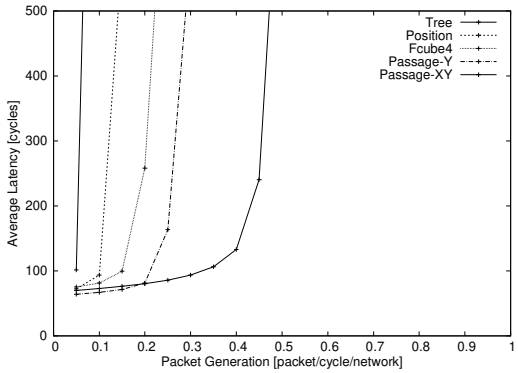
(a)　f=2%



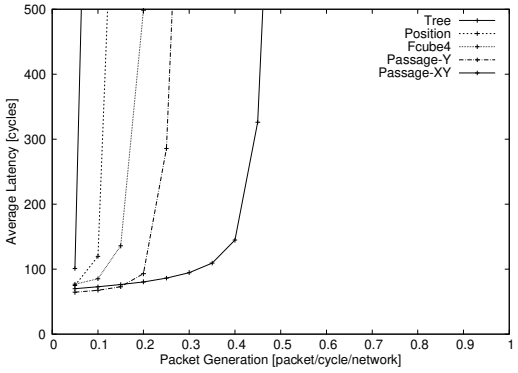
(b)　f=4%



(c)　f=6%

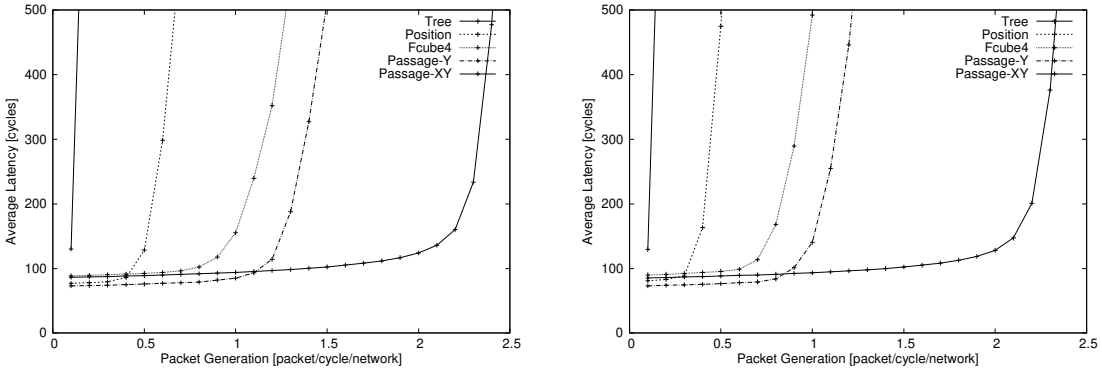


(d)　f=8%

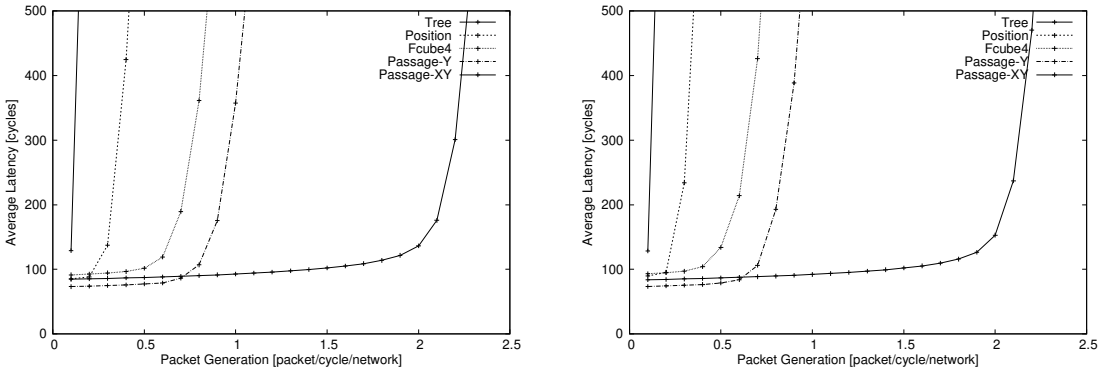


(e)　f=10%

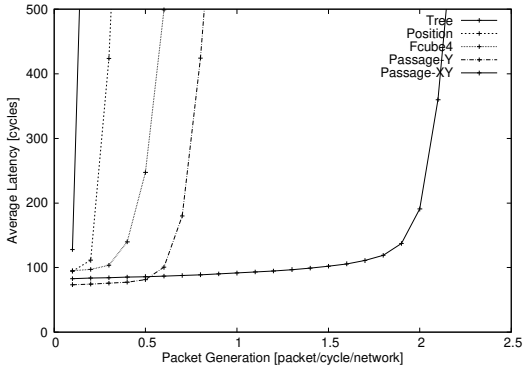
図 4.15: 故障ノードの通過に基づく決定的な提案手法の平均レイテンシ ($N = 10$, $L_p = 32$)



(a)　f=2%　　(b)　f=4%



(c)　f=6%　　(d)　f=8%



(e)　f=10%

図 4.16: 故障ノードの通過に基づく決定的な提案手法の平均レイテンシ ($N = 20$, $L_p = 16$)

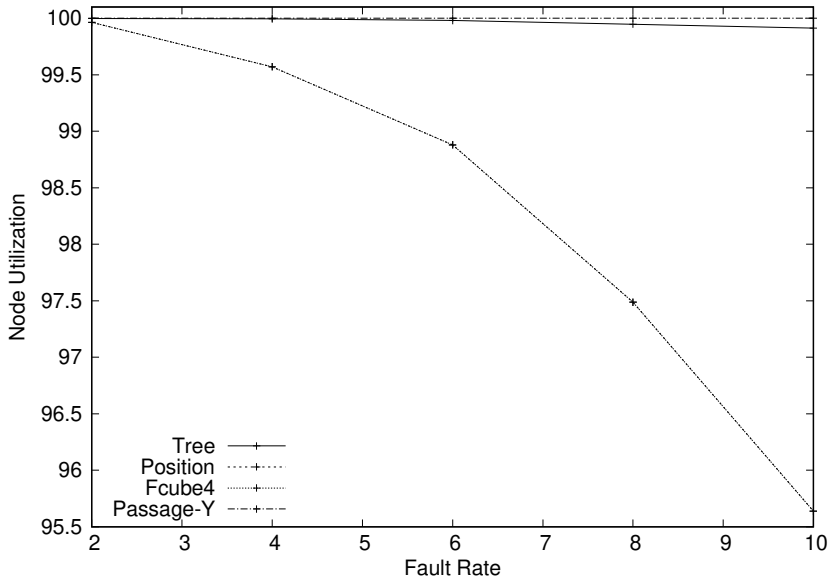


図 4.17: ノード使用率

4.4.3　ノード使用率

本節では，各手法における使用可能な正常ノード数を明らかにするため，ノード使用率を評価する．ノード使用率とは，全ノード数に対する正常ノード数の割合であり，次式で定義される．

$$\text{ノード使用率} = \frac{\text{全ノード数} - \text{故障ノード数} - \text{不使用ノード数}}{\text{全ノード数} - \text{故障ノード数}} \times 100.$$

図 4.17 に Passage-Y と従来手法のノード使用率を示す．$f = 2\%$ の場合，各従来手法のノード使用率は約 99% であり，$f = 10\%$ の場合，Position と Fcube4 のノード使用率は約 96% であり，Tree のノード使用率は約 99% である．Passage-Y は，いずれの故障率においてもノード使用率は 100% であり，このことは，ノードを無駄なく全て利用できるという望ましい特徴を有することを示している．

4.4.4　ベンチマークの実行時間

レイテンシの削減が並列プログラムの実行時間に与える影響を明らかにするために，実際のベンチマークを用いて，各手法でルーティングを行った場合の実行時間を評価する．本評価では，並列システムの評価で一般的に用いられている NAS パラレルベンチマーク [39] の整数ソート（IS）を用いる．これは，ランダムに生成される整数列に対して，バケットソートを並

列に行い，昇順にソートするプログラムである．IS では，整数列の長さをいくつかのクラスから選択できるようになっているが，本評価では，クラス W（整数列長=2^{20}）とする．また，並列数（並列に実行されるプロセス数）は 64 とする．

本研究で開発したシミュレータでは，プロセスが実行されるノード間の通信時間をシミュレートすることは可能であるが，ノードで実行される実際の計算時間はシミュレートすることはできない．そこで，以下に示す方法により，IS の実行時間を見積もることとする．

1. 計算時間の取得
   IS のソースコードにタイマ関数を挿入し，プロセス間通信以外の計算処理に要する計算時間を取得する．
2. プロセス間通信の情報の取得
   プロセス間通信が行われる箇所に通信先のプロセス番号や通信のデータ量などを出力する処理を新たに追加し，プロセス間の通信パターンを取得する．
3. ソースコードのコンパイルとプログラムの実行
4. 実行時間の見積もり
   - 計算時間 （task）
     取得した実際の計算時間（実時間）から，想定する CPU コアとルータの動作周波数を基に，シミュレータでの計算時間（所要サイクル数）を次式で算出する．

     $$\text{所要サイクル数} = \text{実時間} \times CPU \text{ コアの動作周波数} \times \frac{\text{ルータの動作周波数}}{CPU \text{ の動作周波数}}.$$

     本評価では，CPU コアの動作周波数を 2GHz，ルータの動作周波数を 200MHz として想定する．1Hz がシミュレータ上の 1 サイクルに相当する．
   - 通信時間 （comm）
     取得した実際の通信パターンを用いてシミュレータでレイテンシを計測する．
5. ベンチマークの実行時間の出力

上記の手順 1 から得られた実際の計算時間と手順 2 から得られたプロセス間通信の情報を表 4.5 に示す．task と comm は複数回出現するため，処理順に 0 から番号をつけている．各 comm は，プロセス間通信が 1 対 1 通信か集団通信かを表している．加えて，各 comm の括弧内には，ベンチマークで使用されている MPI 通信ライブラリ [37] の通信関数名を示す．

以上のデータを用いて，故障率を 2% または 10% とし，Position，Fcube4，Passage-Y，Passage-XY を用いた場合の IS の実行時間を見積もる．その際，各手法で，故障パターンとプロセスの割り当てパターンを全て統一した．後者については，使用可能ノード数が最も少ない Position を基準とし，ノード (0,0), (0,1), (0,2), ... の順に，正常ノードにのみ昇順にプロセス 0 ～ 63 を割り当てた．

表 4.5: IS ベンチマークの各処理

task 0	0.000109 [s]
comm 0	集団通信 (MPI_Allreduce)
task 1	0.000004 [s]
comm 1	集団通信 (MPI_Alltoall)
task 2	0 [s]
comm 2	集団通信 (MPI_Alltoallv)
task 3	0.000043 [s]

表 4.6: 故障率 2% の場合の各手法の合計サイクル

	Position	Fcube4	Passage-Y	Passage-XY
task 0	218,000	218,000	218,000	218,000
comm 0	574,408	480,475	454,335	426,958
task 1	8,000	8,000	8,000	8,000
comm 1	24,759	18,241	18,215	15,210
task 2	0	0	0	0
comm 2	156,379	136,770	123,608	116,286
task 3	86,000	86,000	86,000	86,000
合計	1,067,546	947,486	908,158	870,454

故障率 2% と 10% の測定結果をそれぞれ表 4.6, 4.7 に示す．故障率 2% の場合，Passage-XY の実行時間は，Position，Fcube4，Passage-Y と比較するとそれぞれ，約 18%，約 8%，約 4% 短縮される．また，通信にかかった総サイクル数のみで比較すると，Passage-XY は，それぞれ，約 26%，約 12%，約 6% 短縮される．故障率 10% の場合，Passage-XY の実行時間は，それぞれ，約 46%，約 24%，約 17% 短縮される．通信にかかった総サイクル数のみの比較では，Passage-XY は，それぞれ，約 57%，約 33%，約 24% 短縮される．

ここで，参考データとして，$f = 2\%$ で Passage-XY を用いた場合の comm0 のパケット生成率の時間推移を図 4.18 に示す．横軸は，サイクルを表しており，縦軸は，そのサイクル内でのパケット生成率を表している．このグラフは，10,000 サイクル毎に，その期間内に生成されたパケット数からパケット生成率を算出したものである．例えば，この図の 220,000 サイクル付近では，パケット生成率が約 0.72 である．これは，図 4.14 (a) におけるパケット生成率が

表 4.7: 故障率 10% の場合の各手法の合計サイクル

	Position	Fcube4	Passage-Y	Passage-XY
task 0	218,000	218,000	218,000	218,000
comm 0	961,169	622,125	538,684	416,448
task 1	8,000	8,000	8,000	8,000
comm 1	42,913	24,332	21,846	15,451
task 2	0	0	0	0
comm 2	266,062	171,024	157,919	112,744
task 3	86,000	86,000	86,000	86,000
合計	1,582,144	1,129,481	1,030,449	856,643

0.72 の場合に相当する．パケット生成率のピークがどのくらいの値になるかは，対象とするアプリケーションに依存する．このため，前節で示したように様々なパケット生成率で手法を比較することが重要である．

以上の結果から，ベンチマークの実行時間は，Passage-XY が最も短いことを明らかにした．また，通信時間においても，Passage-XY が 4.4.2 節の傾向と同様に，最も高い通信性能であることを明らかにした．

4.5　回路量の評価

追加した回路とルーティングアルゴリズムの回路の回路量を明らかにするために，故障ノードの通過に基づく決定的な耐故障ルーティング法の基本手法である Passage-Y の回路設計を行い，動作確認および回路量の評価を行う．統合開発環境として Xilinx 社の Vivado 2013.4 を用いて，Verilog HDL で回路設計を行った．Vertex7 の FPGA デバイス xc7vx485tffg1761-2 を対象に論理合成を行った．従来手法として，3.2 節で述べた NoC 向きの耐故障ルーティング法の Message を用いる．

Message を実装したルータは，1,865LUT を要し，本章で提案した Passage-Y は，664LUT を要した．Passage-Y のルータは，ルータ全体で Message のルータと比較して約 64% 減少する．ルーティング回路の LUT は，Message の場合，193LUT であり，Passage-Y の場合，18LUT であった．さらに，Passage-Y では，追加したスイッチは，27LUT であった．以上より，Passage-Y は，スイッチを追加したにもかかわらず，Message に比べて回路量が減少していることがわかる．この理由は，ルーティングアルゴリズムが Message よりもシンプルなも

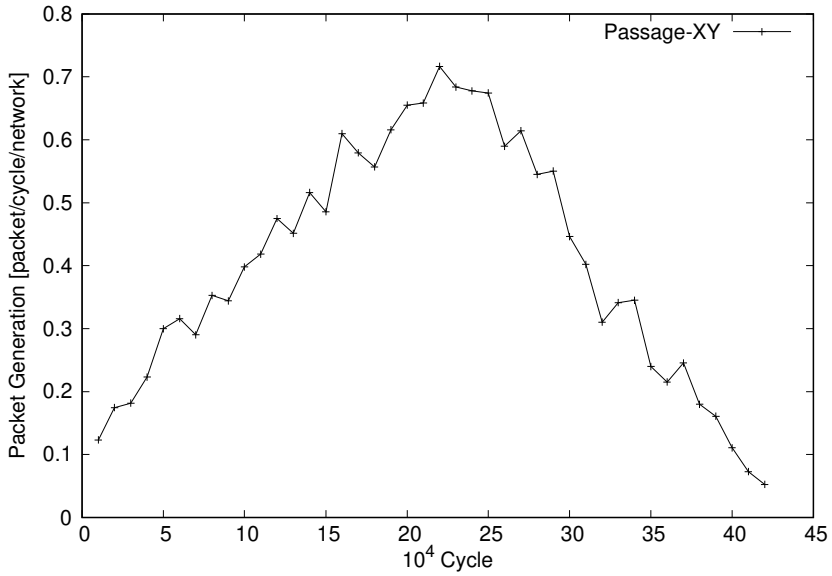


図 4.18: Passage-XY における時間推移によるパケット生成率

ので実装されているからである．

4.6　むすび

本章では，故障ノードの通過に基づく決定的な耐故障ルーティング法を提案した．そのために，故障ノードの通過を可能にする 2 次元メッシュ NoC のアーキテクチャを提案した．また，提案した 2 つの手法のデッドロックフリー性について説明した．さらに，本手法の有効性を明らかにするために，通信性能と回路量の評価を行った．

その結果，ネットワークサイズやパケット長，故障率がいずれの場合においても，提案手法の Passage-Y と Passage-XY は，どの手法よりもレイテンシが低いことがわかった．また，Passage-XY は，Passage-Y と比較しても最大約 97% レイテンシを削減できることがわかった．さらに，全てのノードにスイッチやリンクを追加したにも関わらず，従来手法よりも約 64% 回路量が少ないことを明らかにした．このことから，故障ノードの通過に基づく 2 次元メッシュ NoC は，非常に高い通信性能で少ない回路量の NoC を実現することが可能であることを明らかにした．

本手法は，目的ノードと出発ノードが決まると経路が固定される決定的ルーティングであるため，ネットワークが混雑した場合，混雑する経路を回避することが不可能である．そのため，混雑する経路を回避可能な適応的ルーティングでは，本手法よりもさらなる通信性能の向上が

期待される．

第 5 章

故障ノードの通過に基づく適応的な耐故障ルーティング法

5.1 まえがき

4 章では，ネットワークの拡張に基づく故障ノードを通過することが可能な 2 次元メッシュ NoC に対して，2 つの耐故障ルーティング法を提案し，それらの手法が従来のどの手法よりも高い通信性能であることを明らかにした．提案した 2 つの耐故障ルーティング法は，目的地と出発地が決定されると経路が固定される決定的ルーティングである．そのため，ネットワークが混雑した場合，混雑した経路を回避することが不可能であるため，通信遅延が高くなる．

複数の経路を選択可能である適応的ルーティングでは，一般的に，混雑の回避が優れていると考えられている．本章では，一般的な適応的ルーティングである West-Last ルーティング法と East-Last ルーティング法を併用した，通過に基づく適応的な耐故障ルーティング法を提案する．本手法の通信性能の評価においては，従来の適応的な手法と比較評価に加え，4 章で提案した決定的な手法との比較評価を行う．さらに，本手法の適応性を制限した場合，並びに，特定のノードが込み合うホットスポットのトラヒックの場合に対する評価を行い，本手法の有効性を示すとともに，故障ノードの通過に基づく適応的なルーティング法を設計する際の留意点を明らかにする．

5.2 節では，West-Last ルーティング法と East-Last ルーティング法を併用した耐故障ルーティング法である Passage-WLEL を提案する．5.3 節では，提案手法と従来手法の通信性能を比較する．さらに，提案手法と 4 章で提案した Passage-Y と Passage-XY の通信性能を比較する．5.4 節では，本章のまとめを述べる．

5.2　故障ノードの通過に基づく適応的な提案手法

5.2.1　基本手法

4 章で，故障ノードの通過に基づく 2 次元メッシュ NoC のアーキテクチャに対して提案した Passage-Y と Passage-XY は，従来の耐故障ルーティング法と比較して，通信性能が高いことを明らかにした．これらの手法は，決定的ルーティングであるため，出発ノードと目的ノードが決まると経路が固定される．そのため，ネットワークが混雑した場合，混雑を回避する経路を選択することができなく，通信遅延が高くなる．本節では，West-Last ルーティング法と East-Last ルーティング法を併用し，東方向と西方向に適応性を持ち最短経路を選択することが可能な通過に基づく耐故障ルーティング法を提案する．本論文では，本手法を Passage-WLEL と呼ぶ．

West-Last ルーティング法 [38] は，図 2.5 で示した WS ターンと WN ターンを禁止した手法であり，東側の転送で適応的にルーティングを行うことが可能である．また，East-Last ルーティング法は，ES ターンと EN ターンを禁止した手法であり，西側の転送で適応的にルーティングを行うことが可能である．

West-Last ルーティング法と East-Last ルーティング法は，それぞれ目的ノードの場所により，取り得る経路が異なる．図 5.1 に，それぞれの手法で取り得る経路の例を示す．West-Last ルーティング法の場合，図 5.1（a）に示すように，目的地が東方向の場合のみ，複数の経路を選択することが可能であり，西方向の場合，経路は一意に決定する．East-Last ルーティング法の場合も同様に，図 5.1（b）に示すように，目的地が西方向の場合のみ，複数の経路を選択することが可能であり，東方向の場合，経路は一意に決定する．そのため，本手法では，東西両方向とも適応的に経路を選択するために，2 つのルーティング法を併用する．

このような適応的ルーティング法を 2 つ併用する方法は，従来法 [40, 41] においても同様のアプローチが取られており，一般的なものである．

5.2.2　通過に基づく適応的な耐故障ルーティング法: Passage-WLEL

故障ノードの通過に基づくアーキテクチャ上で，West-Last ルーティングと East-Last ルーティングを併用したルーティングアルゴリズムを説明する．以下で，本手法のルーティングアルゴリズムを述べる．

通過に基づくアーキテクチャに，これらのルーティングを適用した場合について述べる．両手法を単純に併用した場合，デッドロックが発生する問題がある．なぜなら，各手法の禁止ターンが発生するからである．そのため，1 つのネットワーク上では，2 つの手法を併用して，

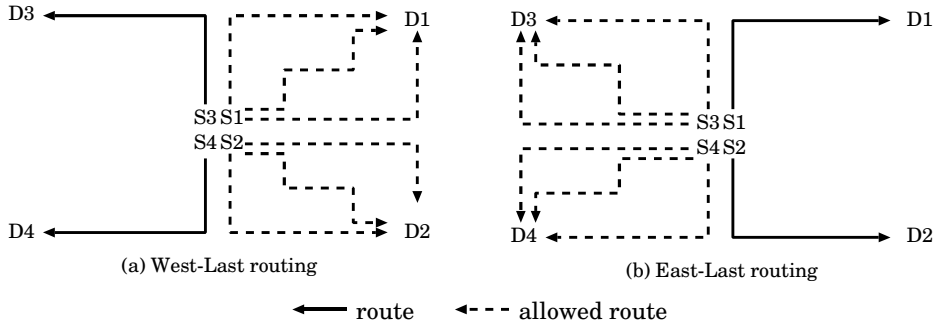


図 5.1: West-Last ルーティング法と East-Last ルーティング法の適応性

かつ，デッドロックを防ぐことは不可能である．

Passage-WLEL では, VC を 2 つ用いてこのデッドロックに対処する．そのため, West-Last ルーティング法と East-Last ルーティング法を異なるチャネル上で動作させる．VC の使用ルールを以下に示す．

- 出発地において，
  - 目的地が東にある場合: VC0 を使用し，West-Last ルーティング法を用いる．
  - 目的地が西にある場合: VC1 を使用し，East-Last ルーティング法を用いる．
- VC0 から VC1 または VC1 から VC0 への移動は禁止する．

上記のように，VC ごとにルーティング法を分けることにより，禁止ターンを別なネットワークに分離できるため，前述したデッドロックは発生しなくなる．

Passage-WLEL は，通過に基づくアーキテクチャ上で動作するため，故障ノードに直面した場合，故障ノードの X 方向と Y 方向の通過を許可するものである．Y 方向について，West-Last ルーティング法と East-Last ルーティング法ともに，北方向と南方向の通過を許可する．X 方向については，West-Last ルーティング法では東方向，East-Last ルーティング法では西方向の通過を許可する．

故障ノードを通過することにより，目的地までの経路が遠回りになる場合がある．最短経路で目的地の経路を選択できない例を図 5.2 に示す．故障ノードが連続して存在する場合，図 5.2 (a) に示すように，故障ノードを通過して遠回りになる場合や 180 度ターンが発生する場合など，最短経路を選択することができない場合がある．また，180 度ターンが発生するとデッドロックが発生しやすくなる．一方で，図 5.2（b）に示すように，目的地のノードの隣接ノードが全て故障ノードである場合，目的地に近づく経路だけを選択すると，到着できない場合がある．

本手法では，決定的ルーティングである Passage-Y や Passae-XY とは異なり，適応的な

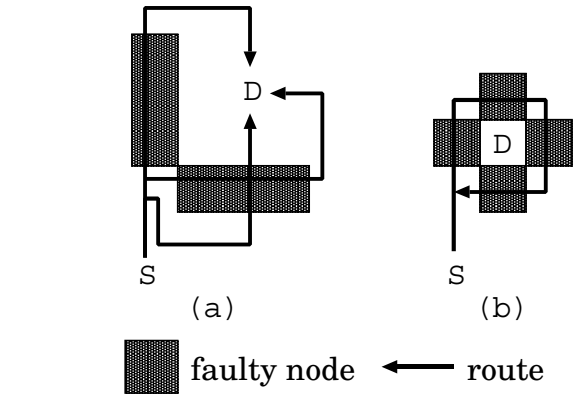


図 5.2: 最短経路を選択できない例

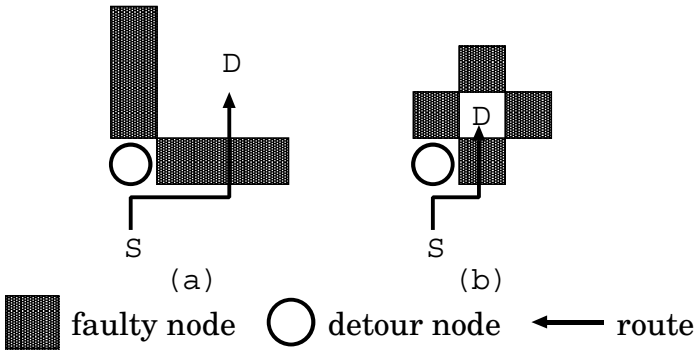


図 5.3: 最短経路を選択する例

ルーティングを行うため，いずれの場合でも，目的地に到着することは可能である．しかし，遠回りによるレイテンシの増加や 180 度ターンによるデッドロックの発生を抑制するために，目的地の X と Y 座標の両方を超えない場合のみ，通過を許可する．それ以外の場合は，以下で定義される迂回ノードを設定し，迂回ノードに侵入する前に最短経路を選択する．

定義 5.1. あるノード (i, j) において，X 方向の隣接ノードが 1 つ以上故障または迂回ノードであり，かつ Y 方向の隣接ノードが 1 つ以上故障または迂回ノードである場合，(i, j) を迂回ノードとする．

定義 5.1 は，迂回ノードが新たに生成されなくなるまで，繰り返し適用される．

図 5.3 に，迂回ノードを設定した例を示す．進行方向に迂回ノードがある場合，そのノードには侵入せずに，目的地に近づく方向に迂回する．これにより，目的地まで最短経路を選択することが可能になる．

本手法では，4 章で定義した定義 4.6 と定義 4.7 の情報に加え，以下の情報を用いる．

定義 5.2. あるノードの南方向，北方向に隣接する連続した故障ノードの数を F_S, F_N とする．

これにより，故障ノードの通過により，パケットが目的地を越えてしまうことを防ぐ．

アルゴリズム 5.1 に，本手法のルーティングアルゴリズムを示す．アルゴリズム内の変数は，3 章で定義した定義 3.5 と同様に，パケットの出発地 S，現在地 C，目的地 D の座標を，それぞれ，(S_x, S_y), (C_x, C_y), (D_x, D_y) で表す．このアルゴリズムは，パケットの目的地が西の場合，East-Last ルーティング法を基にしたアルゴリズムを使用し，パケットの目的地が東の場合，West-Last ルーティング法を基にしたアルゴリズムを使用するものである．

図 5.4 に，Passage-WLEL のルーティングアルゴリズムの概略図を示す．図 5.4 に示すように，故障ノードの場合でも SF ノードの場合でも，適応的にルーティングを行うことが可能である．図 5.5 に，ルーティング例を示す．パケット 1，2，4 は通過により，C_x が D_x，C_y が D_y を超えないように，故障ノードと SF ノードを通過し，複数の経路を選択することが可能である．一方，パケット 3 は東に移動すると最短経路を選択することができなくなるため，南方向に移動する一意の経路が決定される．Passage-WLEL では，X と Y 方向を通過することが可能であり，最短経路を選択することが可能であるため，Passage-Y と Passage-XY のルーティング例を示した図 4.9，図 4.13 の例と比べて，ホップ数の少ないルーティングが可能である．

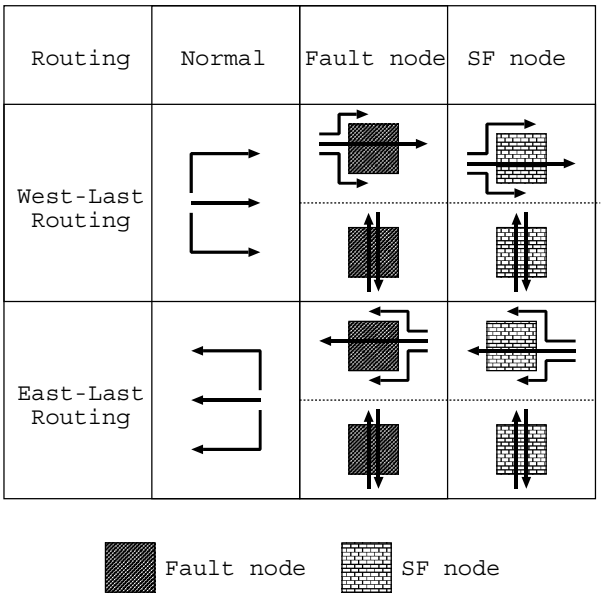


図 5.4: Passage-WLEL のルーティングアルゴリズムの概略図

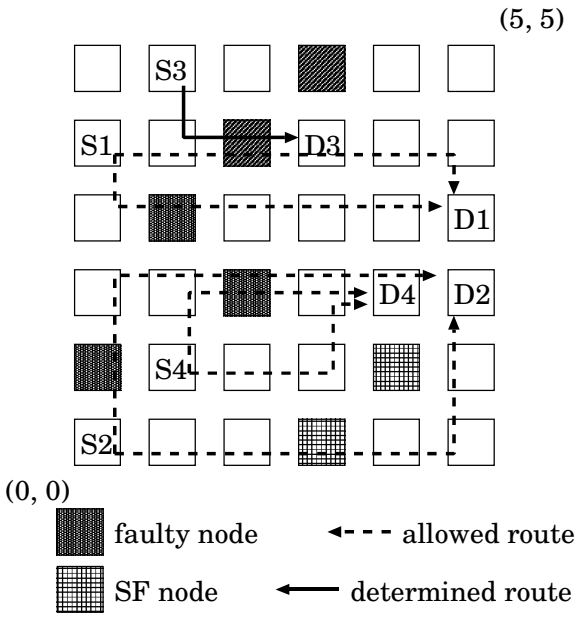


図 5.5: Passage-WLEL のルーティング例

アルゴリズム 5.1: Passage-WLEL のアルゴリズム

```
1  Passage−WLEL_Routing (C, D)
2  if (C_x == D_x && C_y == D_y)
3    consume the packet
4  elsif (C_x > D_x)
5    if (C_y > D_y)
6      Next = WLEL_Routing(C, D, WEST, SOUTH);
7    else
8      Next = WLEL_Routing(C, D, WEST, NORTH);
9    endif
10 elsif (C_x < D_x)
11   if (C_y > D_y)
12     Next = WLEL_Routing(C, D, EAST, SOUTH);
13   else
14     Next = WLEL_Routing(C, D, EAST, NORTH);
15   endif
16 elsif (C_x == D_x)
17   Next= Y_Routing(C_y, D_y);
18 endif
19
20 WLEL_Routing(C, D, Dir_x, Dir_y)
21 J_WEST = 1 if C_x − F_{Dir_x} > D_x, or 0 otherwise
22 J_EAST = 1 if C_x + F_{Dir_x} < D_x, or 0 otherwise
23 J_SOUTH = 1 if C_y − F_{Dir_y} > D_y, or 0 otherwise
24 J_NORTH = 1 if C_y + F_{Dir_y} < D_y, or 0 otherwise
```

```
25 if (J_{Dir_x} == 1 & J_{Dir_y} == 1)
26   Next = either Modify_X_Routing(C_x, D_x, Dir_x) or
27            Y_Routing(C_y, D_y);
28 elsif (J_{Dir_x} == 0 & J_{Dir_y} == 1)
29   Next = Y_Routing(C_y, D_y);
30 else
31   Next = Modify_X_Routing(C_x, D_x, Dir_x);
32 endif
```

5.2.3　Passage-WLEL のデッドロックフリー性

Passage-WLEL のデッドロックフリー性を示す．

定理 5.1. ルーティングアルゴリズム Passage-WLEL は，デッドロックフリーである．

証明. Pasage-WLEL では，180 度ターンを許可していないため，図 2.5 に示したターンにより，時計回りと反時計回りのパケットの循環待ちが発生しないことを示す．Passage-WLEL では，故障ノードがない場合は，それぞれ West-Last ルーティング，East-Last ルーティングと等価な動作をする．そのため，West-Last ルーティングでは，WS，WN ターンが，East-Last ルーティングでは，ES，EN ターンが発生しない．故障ノードを迂回する際に発生するこれらのターンにおいて，パケットの循環待ちが発生しないことを示す．

時計回りの方向について考える．Passage-WLEL では，SF ノードの迂回で WN ターンが，SF ではない故障ノードの迂回で ES ターンが発生する．5.2 節で示した，VC の使用ルールから，VC0 のネットワークでは ES ターンが発生するが，WN ターンは決して発生しない．同様に，VC1 のネットワークでは WN ターンが発生するが，ES ターンは決して発生しない．VC0 と VC1 の間ではパケットの移動は禁止されているため，片方で発生したターンが，もう片方に影響することもない．従って，各 VC のネットワークで，時計回りのパケットの循環待ちは決して発生しない．

反時計回りの方向について考える．SF ノードの迂回で WS ターンが，SF ノードではない故障ノードの迂回で EN ターンが発生する．5.2 節で示した，VC の使用ルールから，VC0 のネットワークでは EN ターンが発生するが，WS ターンは決して発生しない．同様に，VC1 のネットワークでは WS ターンが発生するが，EN ターンは決して発生しない．VC0 と VC1 の間ではパケットの移動は禁止されているため，片方で発生したターンが，もう片方に影響することもない．従って，各 VC のネットワークで，反時計回りのパケットの循環待ちも決して発生しない．

以上から，Passage-WLEL は，デッドロックフリーである． □

表 5.1: Passage-WLEL に対する最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

	f				
Mb	2%	4%	6%	8%	10%
Adapt Fcube	84%	94%	96%	97%	98%
	(0.85)	(0.80)	(0.80)	(0.80)	(0.75)

5.3　通信性能の評価

5.3.1　評価方法

提案手法の通信性能を評価するために，C 言語でシミュレータを開発し，パケット転送を行ったときのレイテンシを測定した．3.5.1 節の表 3.1 に示したパラメータと同様のパラメータでシミュレーションを行う．ここで，従来手法として，Adapt Fcube [26] と比較する．Adapt Fcube は，West-Last と East-Last ルーティング法を文献 [26] の手法で耐故障化したものである．また，Adapt Fcube，Passage-WLEL ともに，適応的に経路を選択するルーティングとして，よりバッファが空いている方向を選択するものとしている．

5.3.2　適応的な手法のレイテンシの比較評価

本節では，3.5.2 節と同様に，様々な状況を想定した評価を行う．以下に，3.5.2 節と同様の各条件で比較した結果を示す．

- 条件 1 での比較結果
  図 5.6 に，故障率 f= 2% ∼ 10% に対するシミュレーション結果を示す．いずれの故障率においても，パケット生成率が低い場合，レイテンシは同程度であり，パケット生成率が高くなるにつれて，Adapt Fcube よりも Passage-WLEL のレイテンシが低くなる．
  表 5.1 に，Ma を Passage-WLEL としたときの f = 2% ∼ 10% の最大レイテンシ削減率を示す．Passage-WLEL の Adapt Fcube に対する最大レイテンシ削減率は，$f = 2\%$ のとき，84%（$p = 0.85$）であり，$f = 10\%$ のとき，98%（$p = 0.75$）であった．適応的ルーティングにおいても，迂回を基本とする Adapt Fcube に対して，X と Y 方向の通過を許可する Passage-WLEL は，レイテンシの削減効果が非常に高いことがわかった．

表 5.2: Passage-WLEL に対する最大レイテンシ削減率 ($N = 10$, $L_p = 32$)

	f				
Mb	2%	4%	6%	8%	10%
Adapt Fcube	72%	89%	94%	95%	96%
	(0.35)	(0.35)	(0.35)	(0.35)	(0.35)

表 5.3: Passage-WLEL に対する最大レイテンシ削減率 ($N = 20$, $L_p = 16$)

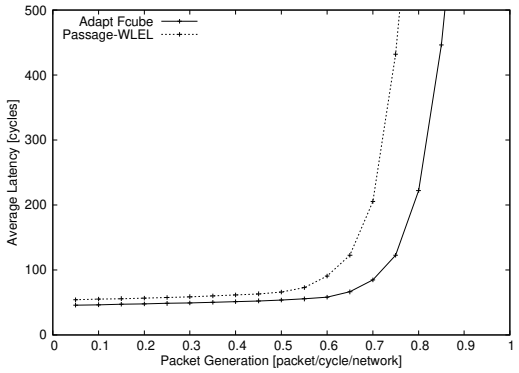
	f				
Mb	2%	4%	6%	8%	10%
Adapt Fcube	86%	92%	94%	88%	93%
	(1.5)	(1.5)	(1.4)	(1.0)	(1.0)

- 条件 2 での比較結果
  図 5.7 に，$f = 2\% \sim 10\%$ のシミュレーション結果を示す．条件 1 と同様に，パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Passage-WLEL のレイテンシが Adapt Fcube よりも低くなる．パケット長を長くしたことにより，全体的にネットワークが混雑しやすくなり，パケット生成率が低い場合でも，各手法のレイテンシは高くなる．
  表 5.2 に，Ma を Passage-WLEL としたときの $f = 2\% \sim 10\%$ の最大レイテンシ削減率を示す．Passage-WLEL は，他の手法と比較して，レイテンシ削減率は 72% 以上であり，通過によるレイテンシの削減効果が非常に大きいことがわかった．
- 条件 3 での比較結果
  図 5.8 に，$f = 2\% \sim 10\%$ のシミュレーション結果を示す．条件 1 と同様に，パケット生成率が低い場合は，各手法のレイテンシは同程度であるが，パケット生成率が高くなると，Passage-WLEL のレイテンシが低くなる．
  表 5.3 に，Ma を Passage-WLEL としたときの $f = 2\% \sim 10\%$ の最大レイテンシ削減率を示す．Passage-WLEL は，Adapt Fcube と比較して，レイテンシ削減率が 86% 以上である．これにより，通過によるレイテンシの削減効果が非常に大きいことがわかった．

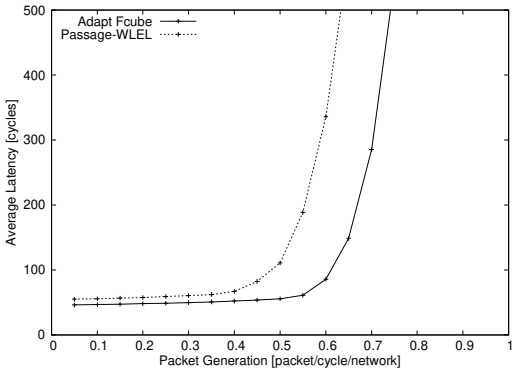
以上の 3 つの比較から，Passage-WLEL は，ネットワークが混雑しやすい場合や大規模な

ネットワークの場合のいずれの場合においても，レイテンシの削減が可能な手法であることを明らかにした．

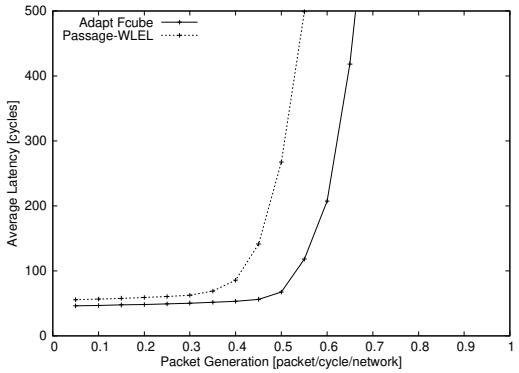
ここで，3.5.2 節のレイテンシの評価と同様に，各手法の平均レイテンシは，95% の信頼度の信頼区間に含まれていることを確認していることを補足する．



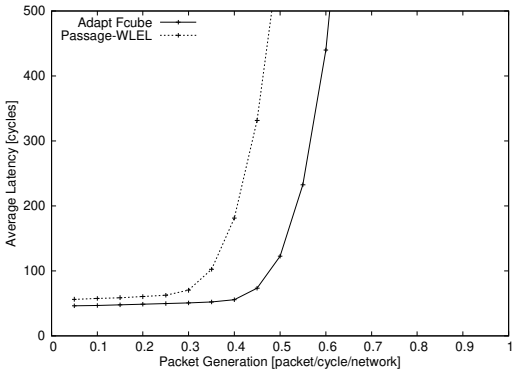
(a)　f=2%



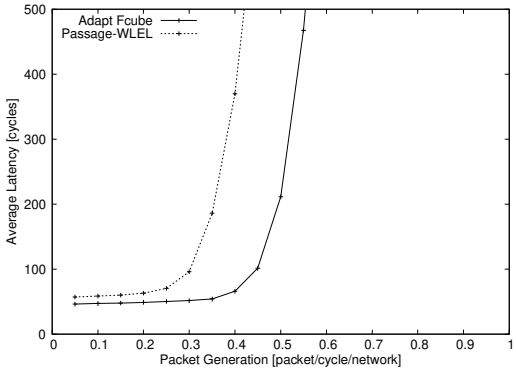
(b)　f=4%



(c)　f=6%

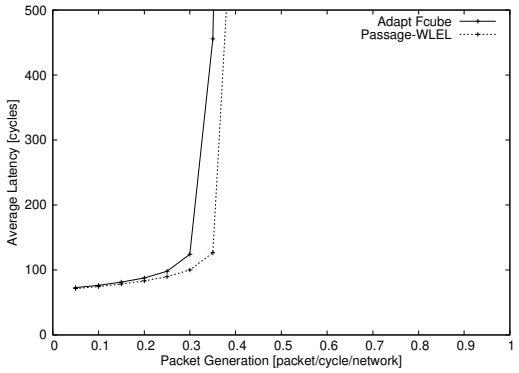


(d)　f=8%

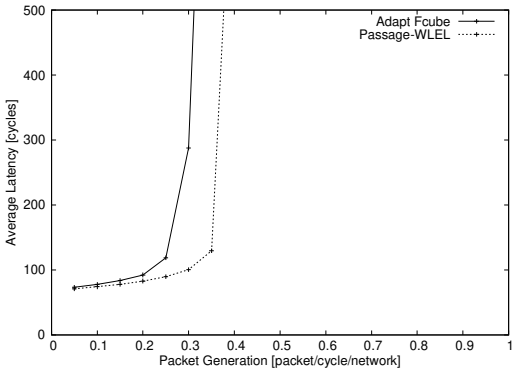


(e)　f=10%

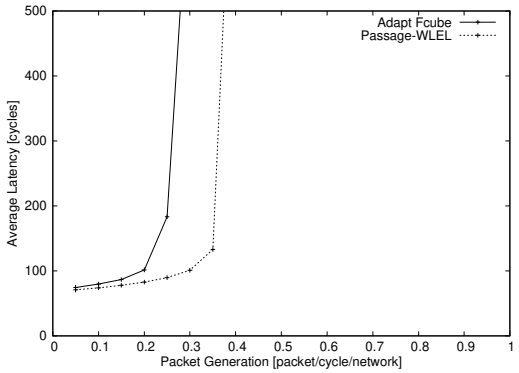
図 5.6: 通過に基づく適応的ルーティング法の平均レイテンシ ($N = 10$, $L_p = 16$)



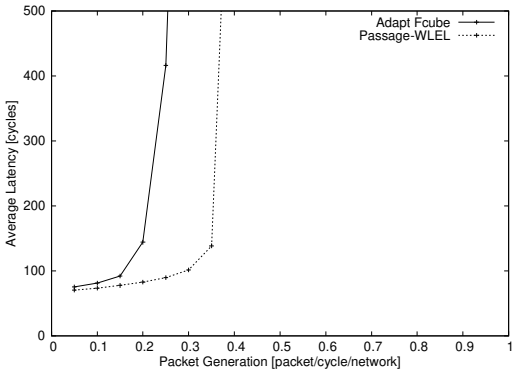
(a)　f=2%



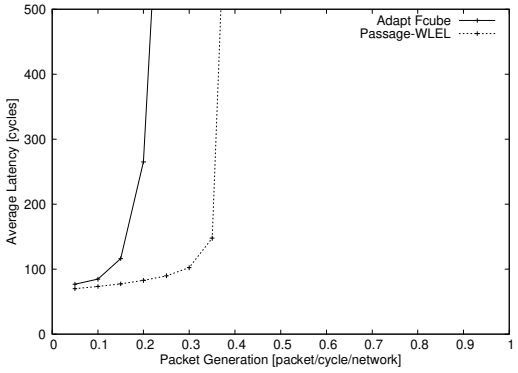
(b)　f=4%



(c)　f=6%

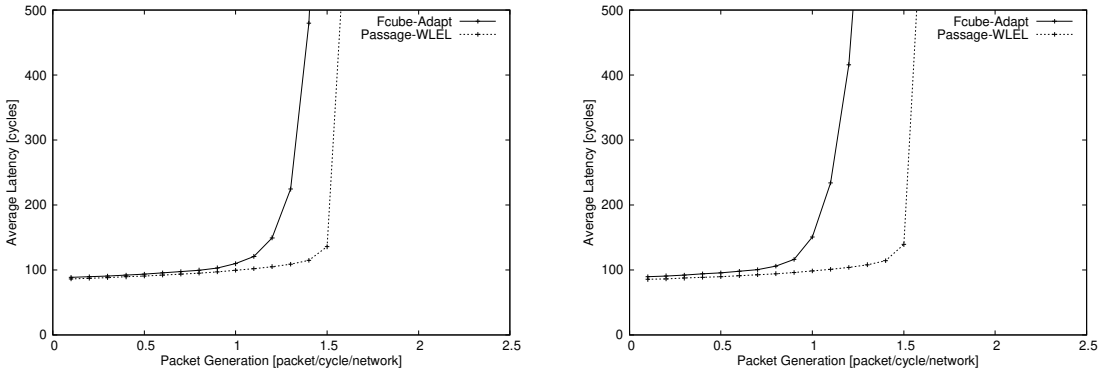


(d)　f=8%

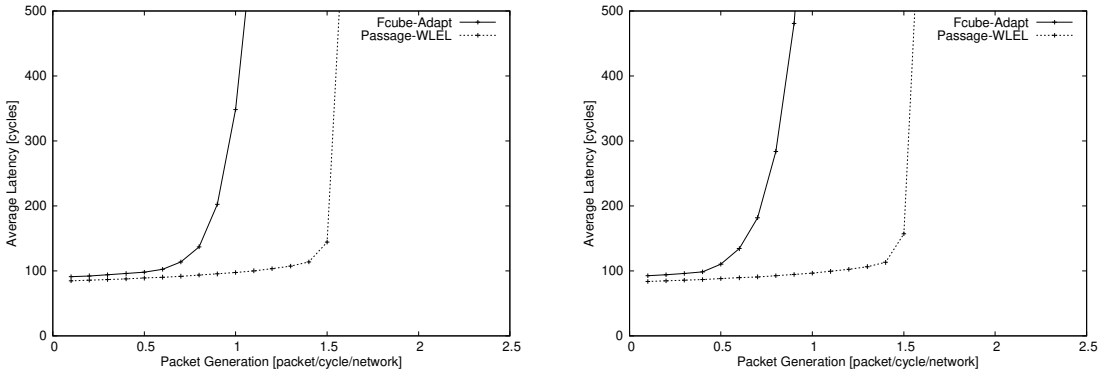


(e)　f=10%

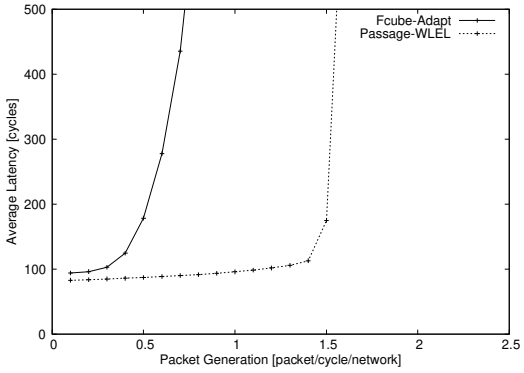
図 5.7: 通過に基づく適応的ルーティング法の平均レイテンシ ($N = 10$, $L_p = 32$)



(a)　f=2%　　(b)　f=4%



(c)　f=6%　　(d)　f=8%



(e)　f=10%

図 5.8: 通過に基づく適応的ルーティング法の平均レイテンシ ($N = 20$, $L_p = 16$)

表 5.4: 提案手法の Passage-WLEL に対する最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

	f				
Mb	2%	4%	6%	8%	10%
Passage-Y	61%	90%	94%	96%	97%
	(0.85)	(0.80)	(0.80)	(0.80)	(0.75)
Passage-XY	-1476%	-1253%	-1077%	-830%	-707%
	(1.00)	(0.95)	(0.95)	(0.90)	(0.90)

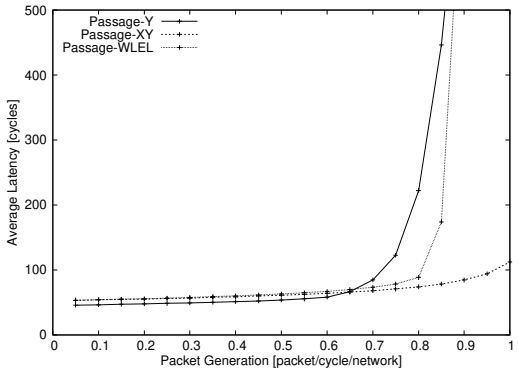
5.3.3　通過に基づく決定的と適法的な手法のレイテンシの比較評価

本節では，4 章と本章で提案した故障ノードの通過に基づく手法における決定的な手法と適応的な手法のレイテンシを比較する．図 5.9 に，$N = 10$，$L_p = 16$，各故障率 f に対するシミュレーション結果を示す．

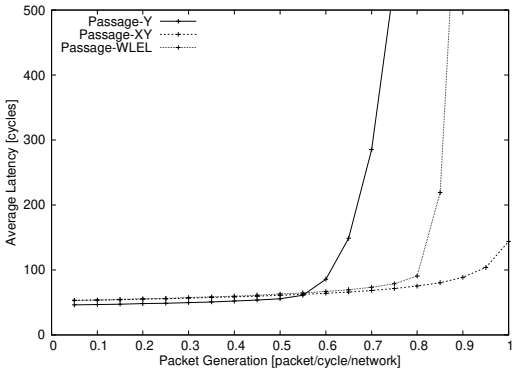
各故障率において，パケット生成率が低い場合，レイテンシは同程度であり，パケット生成率が高くなるにつれて，Passage-Y，Passage-WLEL，Passage-XY の順にレイテンシが低くなる．また，f=2 $\sim$ 10% のいずれの場合においても Passage-XY のレイテンシが一番低いことがわかる．

表 5.4 に，Ma を Passage-WLEL としたときの $f = 2\% \sim 10\%$ の場合の最大レイテンシ削減率を示す．Passage-XY と Passage-WLEL のレイテンシを比較すると，Passage-XY の方がいずれの場合においても，Passage-WLEL よりも低いレイテンシであることがわかる．

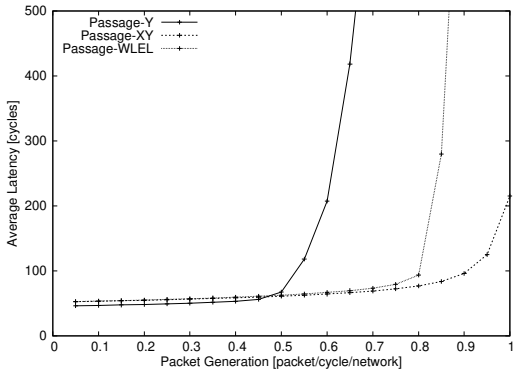
本節での評価により，適応的ルーティングである Passage-WLEL よりも決定的ルーティングである Passage-XY の方がレイテンシが低いことがわかった．一般的に，決定的ルーティングよりも適応的ルーティングの方が性能が高いと考えられている．そこで，次節以降で，本節での結果を検証するために適応的ルーティングのルーティングルールに制限を設けて，本評価の結果の理由を明らかにする．



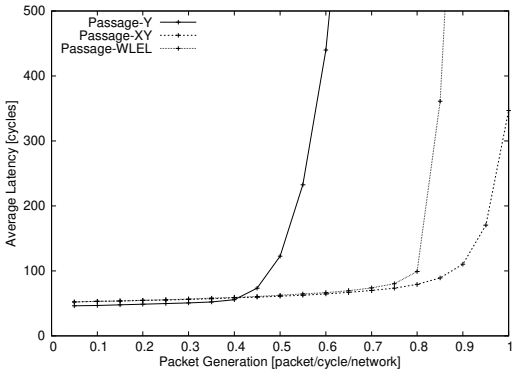
(a)　f=2%



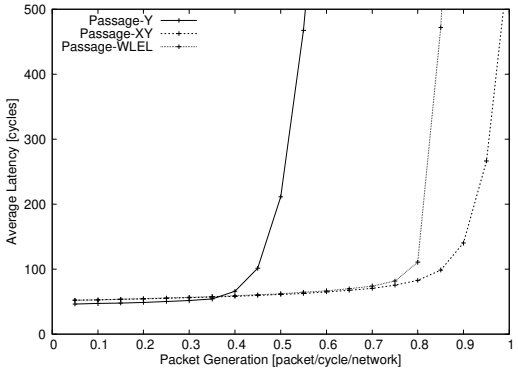
(b)　f=4%



(c)　f=6%



(d)　f=8%



(e)　f=10%

図 5.9: 通過に基づく 3 つの提案手法の平均レイテンシ ($N = 10$, $L_p = 16$)

5.3.4　ターン数を制限した適応的な手法のレイテンシの比較評価

本節では，決定的ルーティングよりも適応的ルーティングの方が高いレイテンシである理由を明らかにする．そのために，Passage-Y と Passage-XY，Passage-WLEL を比較する．

Passage-WLEL では，適応性を減らした手法のレイテンシを計測した．適応性を減らすために，ターン回数の制限を行う．ここで，ターン回数の制限とは，適応的ルーティングにおいて，選択できる経路を制限するために，パケットが目的ノードの到達までの可能なターン数を制限するものである．そのため，ターン回数の少ない場合，選択可能な経路は少なく，多い場合，選択可能な経路は多くなる．図 5.10 にターン回数を制限した際のルーティングの例を示す．目的地 D までの経路において，ターン回数の制限を 1 回にした場合（Turn: 1），出発地 S からの経路を選択したあとは，一意の経路を選択する．ターン回数の制限を 3 回にした場合（Turn: 3），最初の経路を選択した後，3 回だけ経路変更を行う．

図 5.11 に，Passage-Y と Passage-XY，ターン制限を行った Passage-WLEL の結果を示す．パラメータを $N = 10$，$L_p = 16$ とした．グラフの Turn: i は，Passage-WLEL でターン回数を i 回に制限した手法である．ただし，i 回ターンした後で故障ノードを迂回せざるを得なくなると目的地 D に到着できなくなるため，故障ノードを迂回するためのターンは制限回数には含まないものとする．

この結果から，許容ターン数 i を小さくするほどレイテンシが低くなり，決定的ルーティングである Passage-XY に近づくことがわかる．表 5.5 に，Ma を Turn: 1 としたときの最大レイテンシ削減率を示す．Turn: 1 は，Turn: 9 と比較して，最大レイテンシ削減率は約 97% である．すなわち，適応的な経路選択の効果が全く出ていないことがわかる．これは，以下の 2 点が理由として考えられる．

1. Passage-WLEL と Turn: i では，よりバッファが空いている方向を選択するため，混雑が発生しておらず，ターンしなくてもよい箇所でもターンすることがある．これにより，他のパケットの移動を妨げる機会が多くなる．
2. 故障ノード付近では複数のパケットの迂回が重なると混雑が発生するが，これを避けるように経路が選択されるため，故障ノードを通過してホップ数を削減する機会が少なくなる．

適応的ルーティングに制限を設けてレイテンシを比較した結果，Turn: 1 ～ 9 の中では，Turn: 1 が最もレイテンシが低いことを明らかにした．その場合においても，Passage-XY のレイテンシが低いことも明らかにした．次節では，本評価でレイテンシが低い Turn: 1 を用いて，パケットの生成パターンを変更した場合でも同様の結果が得られるかを検証する．

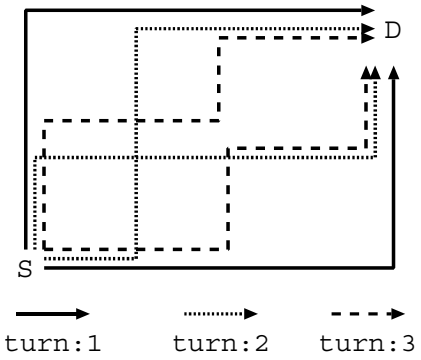
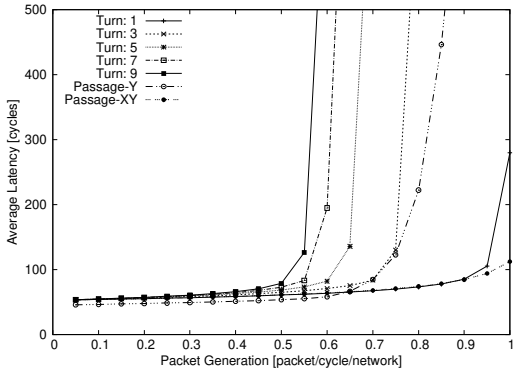


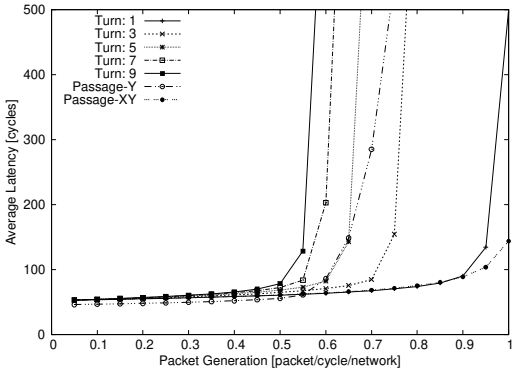
図 5.10: ターン制限を行った場合のルーティング例

表 5.5: パケット生成パターンがランダムの場合の Turn: 1 に対する最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

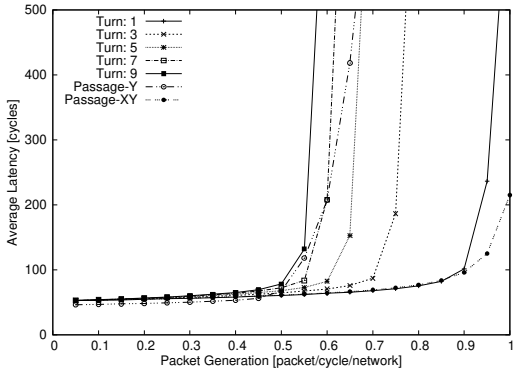
	f				
Mb	2%	4%	6%	8%	10%
Turn: 9	98%	98%	98%	97%	97%
	(0.90)	(0.90)	(0.85)	(0.85)	(0.80)
Passage-XY	-149%	-248%	-244%	-186%	-161%
	(1.00)	(1.00)	(1.00)	(1.00)	(0.95)



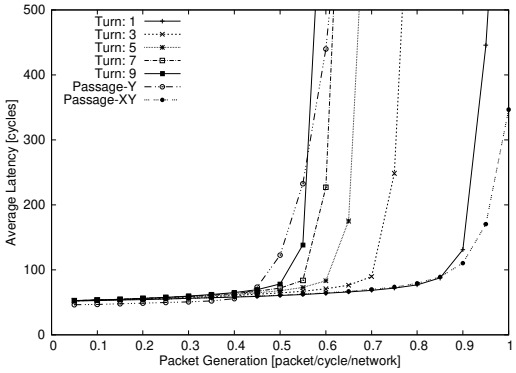
(a)　f=2%



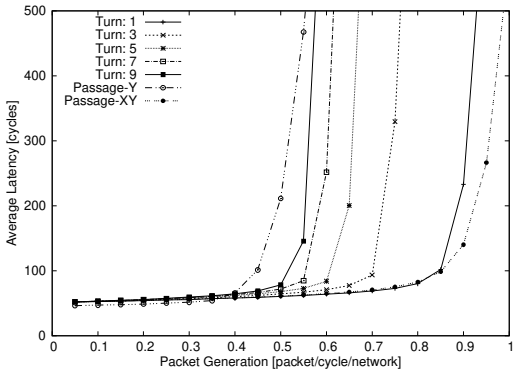
(b)　f=4%



(c)　f=6%



(d)　f=8%



(e)　f=10%

図 5.11: 適応的ルーティング法のターン制限を行なった場合の平均レイテンシ ($N = 10$, $L_p = 16$)

5.3.5　ホットスポットの場合の通過に基づく手法のレイテンシの比較評価

前節では，適応的ルーティングに制限を設けた手法が設けていない手法よりも低いレイテンシであることを明らかにした．本節では，パケットの生成パターンをホットスポットに変更し，Passage-XY と Passage-WLEL，Passage-WLEL にターン回数を 1 回に制限した手法（Turn: 1）でレイテンシを比較する．ホットスポットとは，任意の座標を決定し，パケットの目的地を決定する際，ある確率でその座標を目的地するパケット生成のパターンである．本評価では，ホットスポットの座標になる確率を 5% に設定し，ホットスポットを 1 つにした場合と 2 つにした場合でレイテンシを比較する．

図 5.12 に，ホットスポットを 1 つにした場合の結果を示す．Passage-XY，Passage-WLEL，Turn: 1 の順にレイテンシが低くなり，その差は f が大きくなるほど少なくなる．これは，Passage-XY では，そもそも経路が固定されるため混雑を避けることはできないが，Passage-WLEL では，よりバッファが空いている方向を選択できるためである．表 5.6 に，Ma を Turn: 1 としたときの最大レイテンシ削減率を示す．Turn: 1 は，他の手法と比較して，レイテンシ削減率は 26% 以上である．Turn: 1 が最もよいのは，混雑を回避できることに加えて，ターンが制限されることにより，5.3.4 節の 2. とは逆に，故障ノードを通過する機会が増えるためであると考えられる．

図 5.13 に，ホットスポットを 2 つにした場合の結果を示す．Passage-WLEL が他の手法に比べてレイテンシが高い．Turn: 1 は，レイテンシが高くなり始める点のパケット生成率で Passage-XY のレイテンシと逆転する．ただし，両手法ともネットワークが混雑しており，ネットワークが飽和しているいるため，本評価では，飽和する前の混雑していない点で比較する．ホットスポットが 2 つの場合の結果は，ホットスポットが 1 つの場合の結果と比較すると手法ごとのレイテンシ削減率の差が少なくなっていることがわかる．表 5.7 に，Ma を Turn: 1 としたときの最大レイテンシ削減率を示す．Turn: 1 は，他の手法と比較して，レイテンシ削減率は 10% 以上である．これは，ホットスポットが 2 つ存在するため，1 つの場合に比べて，ネットワーク内の混雑が分散したからだと考えられる．そのため，さらにホットスポットの数が増えると，ネットワークの全体が混雑するため，Passage-XY の方が性能がよくなると考えられる．

以上のことから，通過に基づく 2 次元メッシュ NoC における耐故障ルーティング法では，決定的ルーティングと適応的ルーティングでは，手法ごとに向き不向きがあることを明らかにした．そのため，ネットワークの混雑が分散する場合は，決定的ルーティング法の方がレイテンシが低く，ネットワークのある箇所に混雑が集中する場合は，適応的ルーティング法の方がレイテンシが低いことを明らかにした．その結果，通過に基づく 2 次元メッシュ NoC における耐故障ルーティング法では，混雑の回避と故障ノードの通過の両効果を適切に利用すること

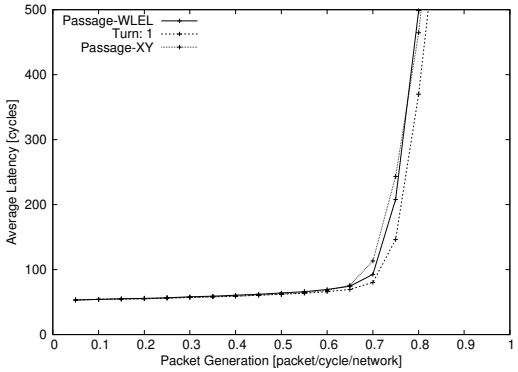
表 5.6: パケット生成パターンがホットスポットの場合の Turn: 1 に対する最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

	f				
Mb	2%	4%	6%	8%	10%
Passage-WLEL	34%	30%	31%	30%	26%
	(0.90)	(0.95)	(0.90)	(0.90)	(0.90)
Passage-XY	40%	34%	39%	32%	30%
	(0.75)	(0.75)	(0.70)	(0.70)	(0.70)

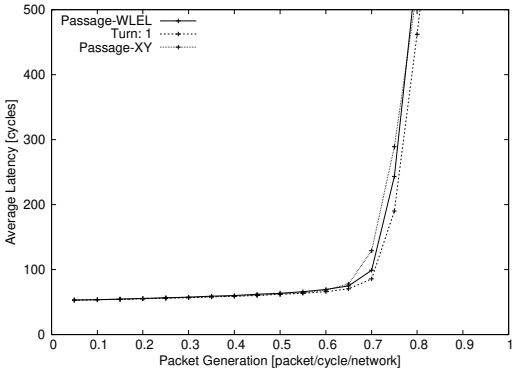
表 5.7: ホットスポットが 2 つの場合の Turn: 1 に対する最大レイテンシ削減率 ($N = 10$, $L_p = 16$)

	f				
Mb	2%	4%	6%	8%	10%
Passage-WLEL	85%	82%	76%	73%	70%
	(0.90)	(0.90)	(0.90)	(0.85)	(0.85)
Passage-XY	10%	13%	11%	16%	20%
	(0.85)	(0.85)	(0.80)	(0.80)	(0.80)

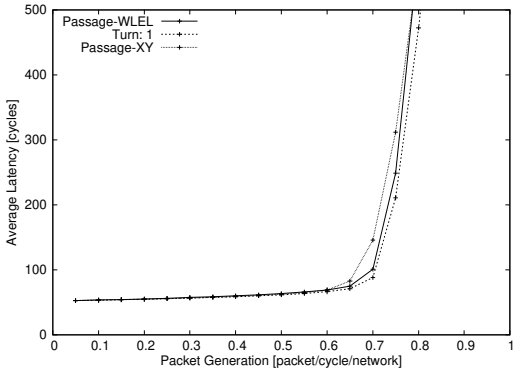
が重要であることがわかった.



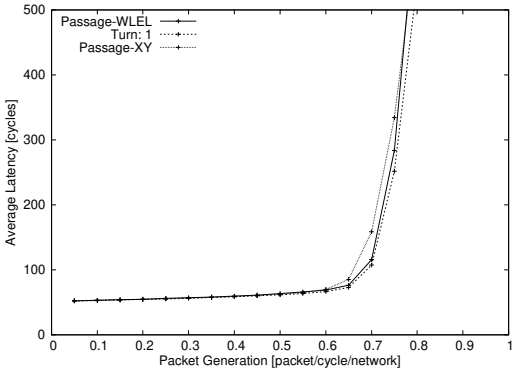
(a)　f=2%



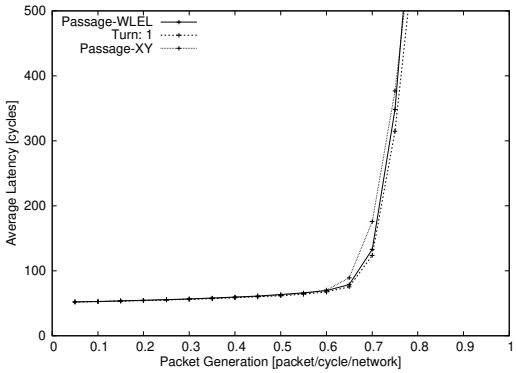
(b)　f=4%



(c)　f=6%

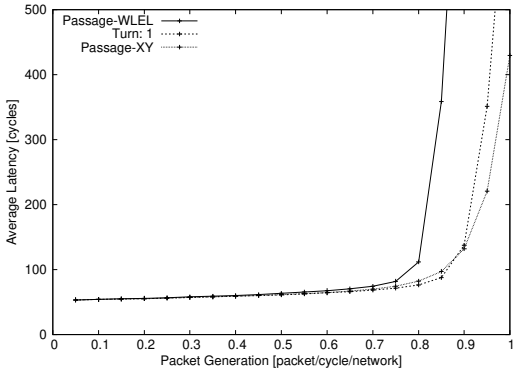


(d)　f=8%

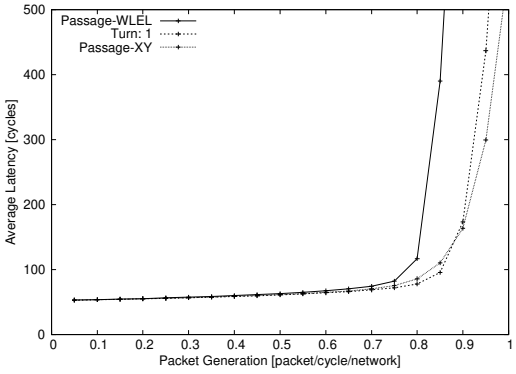


(e)　f=10%

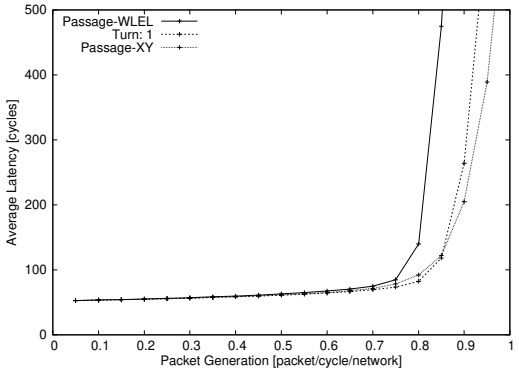
図 5.12: ホットスポットが 1 つの場合のターン制限を行なった場合の平均レイテンシ ($N = 10$, $L_p = 16$)



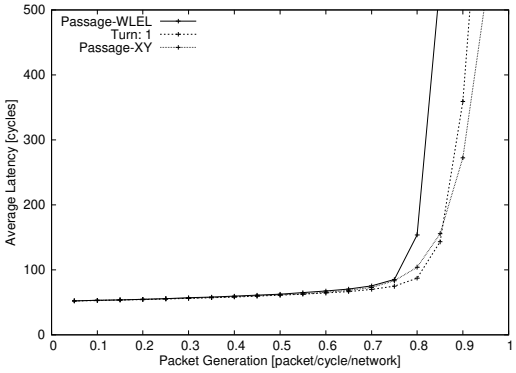
(a)　f=2%



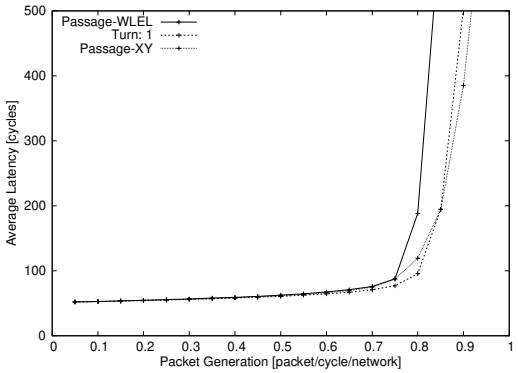
(b)　f=4%



(c)　f=6%



(d)　f=8%



(e)　f=10%

図 5.13: ホットスポットが 2 つの場合のターン制限を行なった場合の平均レイテンシ ($N = 10$, $L_p = 16$)

5.4　むすび

一般的には，決定的ルーティングよりも適応的ルーティングが，ネットワークの混雑回避が優れていると考えられているため，本章では，故障ノードの通過に基づく適応的な耐故障ルーティング法を提案した．

従来の適応的耐故障ルーティング法と比較すると，故障率がいずれの場合においても，最大レイテンシ削減率が約 84% 以上であることを示した．さらに，本章で提案した Passage-WLEL と 4 章で提案した Passage-XY を比較すると，Passage-WLEL の決定的ルーティングである Passage-XY に対する最大レイテンシ削減率が約-1476% であることを示した．この理由を明らかにするために，Passage-WLEL の適応性を制限し，レイテンシを測定した．その結果，より適応的な手法の方がレイテンシがより高くなることを明らかにした．加えて，故障付近の経路は混雑しやすいため，Passage-WLEL は，その経路を回避することにより，通過の経路によるホップ数の削減の機会が少なくなり，レイテンシが高くなることも明らかにした．Passage-WLEL の適応性を制限した手法では，パケットの生成パターンがホットスポットの場合，どの手法よりもレイテンシを最大約 40% 削減できることを示した．

以上の結果から，通過に基づく 2 次元メッシュ NoC における適応的な耐故障ルーティング法では，混雑の回避と故障ノードの通過の両効果を適切に利用することが重要であることを示した．

第 6 章

結言

6.1 結論

VLSI 技術の進歩により，多数のコアを搭載するマルチコアやメニーコアプロセッサが一般的になり，これらのプロセッサにおけるコア間の新たな接続方式としてネットワークオンチップ（Network-on-Chip: NoC）が注目されている．NoC では，各コアにルータを付与したノードをネットワークで接続し，パケット転送を行うことでコア間のデータ通信を行う．NoC では，現在主流の接続方式である共通バス方式とは異なり，ノード間で同時通信が行えるため，通信性能が高い．しかし，NoC が実装される VLSI チップでは，製造時や稼働時において故障が発生するため，正常なルーティングが行えない．そこで，故障ノードを避けて任意のノードへ矛盾なくパケット転送を行う耐故障ルーティング法が必要になる．

これまで，様々な耐故障ルーティング法が提案されており，以下の 2 つの問題が中心的な焦点であった．

1. ノード間の通信遅延を抑制するために，故障ノードをいかに効率よく迂回できるか
2. ルータのオーバヘッドを抑制するために，いかに少ない回路量で実装できるか

しかし，通信遅延と回路量は一般的にはトレードオフの関係にあり，上記の問題を同時に解決し，大規模な NoC の実現に有効な手法は未だに確立されていなかった．

そこで，本論文では，この問題を解決するアプローチとして，ルーティングアルゴリズムを改良する従来の方向性だけではなく，ルーティングが行われるネットワークにも着目し，ネットワークの拡張とルーティングアルゴリズムの改良という方向性から問題の解決を試みた．その方法として，ネットワークの外周部と内部を拡張するアーキテクチャに基づく耐故障ルーティング法を提案した．外周部を拡張する手法では，ネットワークの外周に新たにスイッチとリンクを追加して迂回路を拡張することにより，ネットワークの端の概念をなくすことで，迂

回ルールの統一を可能にした．また，内部を拡張する手法では，各ノード内にスイッチとリンクを追加することにより，ノードが故障した場合にスイッチを切り替えることで，故障ノードの通過を可能にした．以上から，ネットワークの拡張に基づく本提案手法は，通信遅延と回路量の問題を解決するために有効な手法であることを示した．

以下で，本論文の各章の結論をまとめる．

第 2 章 NoC の構成と耐故障ルーティング法

NoC の概要について述べ，NoC におけるルーティングにおける基礎的事項を述べた．その中で，NoC の構成やパケット転送，ルーティング法などを説明した．また，従来提案されてきた耐故障ルーティング法について述べ，大規模な NoC における耐故障ルーティング法の問題点を述べた．

第 3 章 迂回路の拡張に基づく決定的な耐故障ルーティング法

迂回路の拡張に基づく決定的な耐故障ルーティング法を提案した．まず，提案手法で基本手法として用いる領域ベースの耐故障ルーティング法を詳しく説明し，その問題点を指摘した．次に，この問題を解決するために，ネットワークの外周部にスイッチとリンクを拡張した 2 次元メッシュ NoC の構成を述べた．本章では，このアーキテクチャに基づいた，物理的な迂回路の拡張と論理的な迂回路の拡張の 2 つの手法を提案した．通信性能の評価により，本提案手法は，従来手法よりも最大レイテンシ削減率が約 35% であることを明らかにした．また，外周部にスイッチとリンクを追加したにも関わらず，回路量を少なくでき，面積オーバヘッドが 3% 未満と小さいことを示した．

第 4 章 故障ノードの通過に基づく決定的な耐故障ルーティング法

故障ノードの通過に基づく決定的な耐故障ルーティング法を提案した．まず，故障ノードを通過することを可能にするために，ネットワークの内部にスイッチとリンクを拡張した 2 次元メッシュ NoC の構成を述べた．次に，本提案手法で基本手法として用いる XY ルーティングについて述べた．これを基に，Y 軸方向の通過が可能な手法を提案し，さらに，本手法を発展させて，ネットワークを仮想化することで，X 軸と Y 軸方向の両方向の通過が可能な手法を提案した．通信性能の評価により，本提案手法は，どの従来手法よりも最大レイテンシ削減率が 93% 以上であり，ノード使用率が 100% であることを示した．また，内部にスイッチやリンクを追加したにも関わらず，従来手法よりも回路量が約 63% 減少可能なことを示した．

第 5 章 故障ノードの通過に基づく適応的耐な耐故障ルーティング法

故障ノードの通過に基づく適応的な耐故障ルーティング法を提案した．本章では，提案手法で基本手法として用いる適応的な West-Last ルーティングと East-Last ルーティングについて述べ，これらを併用した故障ノードの通過に基づく適応的な手法を提案した．本手法の通信性能の評価において，従来の適応的耐故障ルーティング法と比較して最大レイテンシ削減率が約 98% であることを示した．さらに，適応的ルーティングである本提案手法と決定的ルーティングである 4 章で提案した手法を比較すると，本手法は，4 章の手法よりもレイテンシが高いことがわかった．この理由を明らかにするために，適応的な手法にターン回数の制限をかけて比較して，適応性を最も制限した適応的な手法が最も制限していない手法よりも最大レイテンシ削減率が約 97% であることを示した．加えて，特定のノードが混み合うホットスポットのトラヒックの場合，どの手法よりも，適応性を制限した適応的な手法がレイテンシを最大約 40% 削減可能であることを示した．

以上の結果から，故障ノードの通過に基づく耐故障ルーティング法では，混雑の回避と故障ノードの通過の両効果を適切に利用することが重要であることを明らかにした．

6.2　今後の課題

本論文では，大規模な 2 次元メッシュ NoC の耐故障ルーティング法の問題に対して，ネットワークの拡張に基づく耐故障ルーティング法を提案し，その問題を解決可能にした．コア数を多く含む高性能なプロセッサを実装するにあたり，ハードウェアの観点からは，プロセッサコアやルータの故障検知や製造時の放熱問題，プロセッサコアとルータのデータの送受信のための回路実装などが，ソフトウェアの観点からは，故障を含むネットワークのプロセッサコアへのタスクの割り当て（耐故障タスクマッピング）などが残されているが，本研究により提案された耐故障ルーティング法は，高速な通信が可能であり少ない回路量で実装可能なものとして，大きく貢献できるものと期待する．

本研究では，対象とするネットワークが 2 次元メッシュであったため，他のネットワークトポロジにおいても同様の効果が得られるのかを検証することが今後の課題となる．また，メニーコアプロセッサでプログラムを高速実行するために自動並列化の研究が行われている．この研究により，実行プログラムを自動で各タスクに分割することが可能になる．そのため，本研究の通信システムに加えて，自動並列化やタスクマッピングを組み合わせることにより，より高性能な NoC システムを実現することが可能になる．今後は，本研究の耐故障ルーティングと並行してこれらの研究も行う予定である．

ハードウェアからソフトウェアまで必要な研究を行うことにより，一般的な PC やスマート

フォン，スパコンなどの様々な端末において，組み込み可能な低遅延で小面積な NoC の実用化に貢献することを考えている．

謝辞

本研究は，山口大学大学院創成科学研究科システム・デザイン工学系専攻在学中に，福士 将 准教授のもとで行われました．本研究を進めるにあたり，懇切丁寧かつ適切な御指導，御鞭撻を賜りました同専攻 福士 将 准教授に心より深く感謝いたします．加えまして，学部からこれまでに研究への取り組む姿勢を教えて頂いたり，私が上手くいかず模索している際，解決に導くヒントや，時には叱咤激励を頂いたりと，私の芯となる部分を作って頂きました．心からお礼申し上げます．

本論文審査において副査を務めていただきました同専攻 松藤 信哉 教授には，研究に対して私が見えていなかった部分を明らかにして頂きました．心より感謝致します．同専攻 浜本 義彦 教授には，研究結果のデータの信頼度や有意差について，これまでに考えていなかった，データに対する新たな知見を頂きました．同専攻 多田村 克己 教授には，ベンチマークを用いた研究の新たな糸口を見つけて頂きました．また，研究室を飛び越え様々な有益なご意見を頂きました．心より感謝致します．同専攻 山口 真悟 教授には，研究結果の有効性についての考え方について，ご助言を頂きました．心より感謝致します．さらに，先生方には本論文を改善する上でのご指摘やご助言を頂きまして，感謝申し上げます．加えまして，同大学知能情報工学科から同専攻まで先生方の教えにより今の私があります．重ねて深く感謝申し上げます．

最後に，本論文をまとめるにあたって大変お世話になりました，計算機システム工学研究室の皆様に厚く御礼申し上げます．

参考文献

[1] コロナ薬候補を数十種類 スパコン「富岳」で京大発見, “https://www.nikkei.com/article/DGXMZO61137980T00C20A7EA5000/,” （最終閲覧日: 2020 年 10 月 24 日）.

[2] 気象予測とスパコン, “https://www.r-ccs.riken.jp/jp/ fugaku/pi/climate,” （最終閲覧日: 2020 年 10 月 24 日）.

[3] 菅野 卓雄, 堀口 勝治, “ULSI 設計技術,” 電子情報通信学会, 1993.

[4] S. Y. Kung, S. N. Jean, and C. W. Chang, “Fault-Tolerant Array Processors Using Single-Track Switches,” IEEE Transactions on Computers, Vol. 38, No. 4, pp. 501–514, April 1989.

[5] I. Takanami and M. Fukushi, “A Built-in Circuit for Self-Reconfiguring Mesh-Connected Processor Arrays with Spares on Diagonal,” Transactions on Computational Science XXXIV. Lecture Notes in Computer Science, Vol. 11820, pp. 109–135, August 2019.

[6] M. Fukushi and S. Horiguchi, “An Efficient Self-reconfiguration Algorithm for Degradable Processor Arrays,” Transactions of Information Processing Society of Japan, Vol. 43, No. 4, pp. 844–854, April 2002.

[7] C. P. Low, “An Efficient Reconfiguration Algorithm for Degradable VLSI/WSI Arrays,” IEEE Transactions on Computers, Vol. 49, No. 6, pp. 553–559, Junuary 2000.

[8] W. J. Dally and B. Towles, “Principles and Practices of Inter-connection Networks,” Morgan Kaufman Publishers, 2004.

[9] N. Kadri, M. Koudil, “A survey on Fault-tolerant Application Mapping Techniques for Network-on-Chip,” Journal of Systems Architecture, Vol. 92, pp. 39–52, January 2019.

[10] 鹿股 昭雄, 春日 建, “安全・高信頼システムデザイン入門,” 電気書院, 2013.

[11] 天野 英晴, “並列コンピュータ,” 昭晃堂, 1996.

[12] Intel Xeon Phi X100 Family Coprocessor - the Architecture, “https://software.intel.com/en-us/articles/intel-xeon-phi-coprocessor-codename-knights-corner,” (accessed

10/10/2020).

[13] A. Sodani, R. Gramunt, J. Corbal,H. S. Kim, K. Vinod, S. Chinthamani, S. Hutsell, R. Agarwal, Y. C. Liu, "Knights Landing: Second-Generation Intel Xeon Phi Product," IEEE Micro, Vol. 36, No. 2, pp. 34–46, April 2016.

[14] K. Padmanabhan and D. H. Lawrie, "A Survey of Wormhole Routing Techniques in Direct Network," Computer, pp. 62–76, 1993.

[15] P. Kermani and L. Kleinrock, "Virtual Cut–Through: A New Computer Commnunication Switching Techniques," Computer Networks, Vol. 3, No. 4, pp. 267–286, 1979.

[16] W. J. Dally, "Virtual Channel Flow Control," IEEE Transactions Parallel and Distributed Systems, Vol. 3, No. 2, pp. 194–205, March 1992.

[17] H. K. Hsin, E. J. Chang, C. A. Lin, and A. Y. Wu, "Ant Colony Optimization-based Fault-aware Routing in Mesh-based Network-on-Chip Systems," IEEE Transactions Computer-Aided Design of Integrated Circuits and Systems, Vol. 33, No. 11, pp. 1693–1705, November 2014.

[18] J. Liu, J. Harkin, Y. Li, and L. P. Maguire, "Fault-tolerant Networks-on-Chip Routing with Coarse and Fine-grained Look-ahead," IEEE Transactions Computer-Aided Design of Integrated Circuits and Systems, Vol. 35, No. 2, pp. 260–273, February 2016.

[19] H. Zhao, N. Bagherzadeh, and J. Wu, "A General Fault-tolerant Minimal Routing for Mesh Architectures," IEEE Transactions Computers, Vol. 66, No. 7, pp. 1240–1246, July 2017.

[20] R. G. Mota, J. Silveira, J. Silveira, L. Brahm, A. Zorzo, F. Mor, and Cesar Marcon, "Efficient Routing Table Minimization for Fault-tolerant Irregular Network-on-Chip," Proceedings of IEEE International Conference on Electronics, Circuits and Systems (ICECS), December 2016.

[21] R. Xie, J. Cai, X. Xin, and B. Yang, "Low-Cost Adaptive and Fault-Tolerant Routing Method for 2D Network-on-Chip," IEICE Transactions on Information and Systems, Vol. E100-D, No. 4, pp. 910–913, April 2017.

[22] D. Sinha, A. Roy, K. V. Kumar, P. Kulkarni, and J. Soumya, "Dn-FTR: Fault-tolerant Routing Algorithm for Mesh Based Network-on-Chip," Proceedings of International Conference on Recent Advances in Information Technology (RAIT), March 2018.

[23] T. Moscibroda and O. Mutlu, "A Case for Bufferless Routing in On-chip Networks," Proceedings of the 36th annual international symposium on Computer architecture, pp. 196–207, June 2009.

[24] V. Janfaza and E. Baharlouei, "A New Fault-tolerant Deadlock-free Fully Adaptive Routing in NOC," Proceedings of IEEE East-West Design & Test Symposium (EWDTS), September 2017.

[25] S. Chalasani and R. V. Boppana, "Communication in Multicomputers with Nonconvex Faults," IEEE Transactions on Computers, Vol. 46, No. 5, pp. 616–622, May 1997.

[26] R. V. Boppana, and S. Chalasani, "Fault-Tolerant Wormhole Routing Algorithms for Mesh Networks," IEEE Transactions on Computers, Vol. 44, No. 7, pp. 848–864, July 1995.

[27] K.H. Chen and G.M. Chiu, "Fault–tolerant Routing Algorithm for Meshes without Using Virtual Channels," Journal of Information Science and Engineering, Vol. 14, No. 4, pp. 765–783, 1998.

[28] R. Holsmark and S. Kumar, "Corrections to Chen and Chiu's Fault Tolerant Routing Algorithm," Journal of Information Science and Engineering, Vol. 23, pp. 1649–1662, 2007.

[29] Y. Fukushima, M. Fukushi, and I.E. Yairi, "A Region–based Fault–tolerant Routing Algorithm for 2D Irregular Mesh Network–on–Chip," Journal of Electronic Testing: Theory and Applications, Vol. 29, Issue 3, pp. 415–429, 2013.

[30] B. Fu, Y. Han, H. Li, and X Li, "ZoneDefense: A Fault-Tolerant Routing for 2-D Meshes Without Virtual Channels," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 22, No. 1, pp. 113–126, January 2014.

[31] J.C. Sancho, A. Robles, and J. Duato, "A New Methodology to Compute Deadlock-Free Routing Tables for Irregular Networks," Proceedings of International Workshop on Communication, Architecture and Applications for NetworkBased Parallel Computing (CANPC ' 00), pp. 45–60, January 2000.

[32] J. Flich, T. Skeie, A. Mejia, O. Lysne, P. Lopez, A. Robles, J. Duato, M. Koibuchi, T. Rokicki, and J. C. Sancho, "A Survey and Evaluation of Topology-Agnostic Deterministic Routing Algorithms," IEEE Transactions on Parallel and Distributed System, Vol. 23, No. 3, pp. 405–425, March 2012.

[33] A. Shamaei and H. Sarbazi-Azad, "An Adaptive and Fault-Tolerant Routing Algorithm for Meshes," Proccedings of Computational Science and Its Applications – ICCSA, pp. 1235–1248, 2008.

[34] J. Zhou and F.C.M. Lau, "Adaptive Fault-tolerant Wormhole Routing in 2D Meshes," Proceedings of International Parallel and Distributed Processing Symposium. IPDPS

2001, April 2001.

[35] J. Wu, “A Fault-tolerant and Deadlock-free Routing Protocol in 2D Meshes Based on Odd-even Turn Model,” IEEE Transactions on Computers, Vol. 52, No. 9, pp. 1154–1169, September 2003.

[36] G. M. Chiu, “The Odd-Even Turn Model for Adaptive Routing,” IEEE Transactions on parallel and distributed systems, Vol. 11, No. 7, pp. 729–738, July 2000.

[37] OPen MPI, “https://www.open-mpi.org/,” (最終閲覧日: 2021 年 1 月 1 日).

[38] C. J. Glass and L. M. Ni, “The Turn model for Adaptive Routing,” ACM SIGARCH Computer Architecture News, Vol. 20, No. 2, pp. 278–287, April 1992.

[39] NAS Parallel Benchmarks, “https://www.nas.nasa.gov/publications/npb.html,” (最終閲覧日: 2020 年 12 月 30 日).

[40] J. Duato, “A Theory of Fault-Tolerant Routing in Wormhole Networks,” IEEE Transactions on Parallel and Distributed Systems, Vol. 8, No. 8, pp. 790–802, August 1997.

[41] Y. Miura, K. Shimozono, N. Fukase, S. Watanabe, and K. Matoyama, “An Adaptive Routing Algorithm of 2-D Torus Network Based on Turn Model: The Communication Performance,” International Journal of Networking and Computing, Vol. 5, No. 1, pp. 223–238, 2015,

本論文に関連する研究業績

論文誌

1.Yota Kurokawa and Masaru Fukushi, “Design of An Extended 2D Mesh Network-on-Chip and Development of A Fault-tolerant Routing Method,” IET Computers & Digital Techniques, Vol. 13, Issue 3, pp. 224–232, January 2019.

2. Yota Kurokawa and Masaru Fukushi, “Passage of Faulty Nodes: A Novel Approach for Fault-tolerant Routing on NoCs,” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E102. A, No. 12, pp. 1702–1710, December 2019.

3. 黒川 陽太, 福士 将, “2 次元メッシュ NoC における故障ノードの通過に基づく決定的・適応的耐故障ルーティング法,” 電子情報通信学会論文誌 D, 2021（条件付採録）.

国際学会

1. Yohta Kurokawa and Masaru Fukushi, “A Prototype Router Circuit for Extended 2D Mesh Network on Chips,” Proceedings of IEEE 5th Global Conference on Consumer Electronics, pp. 361–362, 2016.

2. Yota Kurokawa and Masaru Fukushi, “A Fault-tolerant Routing Method for 2D-Mesh Network-on-Chips Based on The Passage of Fault Blocks,” Proceedings of IEEE International Conference on Consumer Electronics-Taiwan, pp. 111–112, 2018.

3. Yota Kurokawa and Masaru Fukushi, “XY Based Fault-tolerant Routing with The Passage of Faulty Nodes,” Proceedings of Sixth International Symposium on Computing and Networking Workshops, pp. 99–104, 2018.

4. Yota Kurokawa and Masaru Fukushi, “Performance Improvement of Region-Based Fault-tolerant Routing Methods Based on The Passage of Fault Blocks,” Proceedings of IEEE International Conference on Consumer Electronics-Taiwan, pp. 413–414, 2019.

5. Yota Kurokawa and Masaru Fukushi, “Effect of Virtual Channels for A Fault-tolerant XY Routing Method with The Passage of Faulty Nodes,” Proceedings of the International

Conference on Information and Education Technology, pp. 267–272, 2020.

6. Yota Kurokawa and Masaru Fukushi, “An Adaptive Fault-tolerant Routing Method for 2D Mesh NoCs Based on the Passage of Faulty Nodes,” Proceedings of International Symposium on Artificial Life and Robotics, pp. 560–565, 2021.

国内学会

1. 黒川 陽太, 福士 将, “ネットワークオンチップのためのルータ回路の設計と評価,” 電子情報通信学会技術報告 機能集積情報システム研究会, FIIS–16–421, pp. 1–6, 2016.

2. 黒川 陽太, 福士 将, “拡張 2 次元メッシュ NoC の設計と評価,” 第 18 回 IEEE 広島支部学生シンポジウム講演論文集, pp. 374–377, 2016.

3. 黒川 陽太，福士 将, “拡張 2 次元メッシュ NoC に対する面積オーバヘッドの評価,” 平成 29 年度 (第 68 回) 電気・情報関連学会中国支部連合大会講演論文集, pp. 1–2, 2017.

4. 黒川 陽太, 福士 将, “耐故障ルーティング法を備えた拡張 2 次元メッシュ NoC の通信性能と面積オーバヘッドの評価,” 電子情報通信学会技術報告 機能集積情報システム研究会, FIIS–17–464, pp. 1–6, 2017.

5. 黒川 陽太, 福士 将, “2 次元メッシュ NoC に対する故障領域の通過に基づく耐故障ルーティング法,” 電子情報通信学会技術報告 機能集積情報システム研究会, FIIS–19–493, 1–6, 2019.

6. 黒川 陽太，福士 将, “故障ノードの通過に基づく耐故障 XY ルーティング法に対する仮想チャネルの適用効果,” 電子情報通信学会技術報告 機能集積情報システム研究会, FIIS–19–515, pp. 1–6, 2019.

7. 黒川 陽太，福士 将, “2 次元メッシュ NoC における耐故障ルーティング法の性能評価,” 電子情報通信学会技術報告 機能集積情報システム研究会, FIIS–20–526, pp. 1–6, 2020.

8. 黒川 陽太，福士 将, “NoC における耐故障・決定的ルーティング法に対する仮想チャネルの適用効果,” 第 33 回 回路とシステムワークショップ講演論文集, pp. 68–73, 2020.

受賞

1. IEEE 広島支部学生シンポジウム 優秀研究賞（2016 年 12 月）
2. 電気・情報関連学会中国支部連合大会　奨励賞（2018 年 3 月）
3. 第 36 回電気通信普及財団賞（テレコムシステム技術学生賞）佳作（2021 年 3 月受賞予定）
4. 第 33 回回路とシステムワークショップ 奨励賞（2021 年 8 月受賞予定）