MEMOIRS OF SHONAN
INSTITUTE OF TECHNOLOGY
Vol. 38, No. 1, 2004

相互結合網におけるデッドロック回復方法の分類

高畠俊徳*

Classification of Deadlock Recovery Routings in Interconnection Networks

Toshinori TAKABATAKE*

In interconnection networks of massively parallel computer systems, deadlock recovery has been studied in routing strategy. However, there has not been established the proper classification of deadlock recovery routings. The routing strategy for the deadlock recovery is intended to optimize the routing performance when deadlocks do not occur. On the other hand, it is important to improve the routing performance by handling deadlocks if they occur. For this purpose, suitable criterion is needed to deal with the deadlock recovery routings. In this paper, by investigating the researches in conventional deadlock recovery routings and by organizing them, the classification of the deadlock recovery routings is proposed. This leads to make it possible to treat the deadlock recovery routings systematically and comprehensivelly.

Key words: interconnection network, deadlock recovery, routing algorithm, dependability, fault tolerance

1. まえがき

並列計算機システムの相互結合網における問題の一つに，ルーティングにより発生するデッドロックがある[9]。従来，この問題を解決するルーティング手法には，フロー制御技術[6]や仮想チャネルを使用してルーティングに制限[5),7)]を設け，デッドロックを防止[9]や回避するための手法[5),7),10),11)]がある（付録 A）。これによって，デッドロックフリーのみならずルーティングの性能向上もなされ，適応ルーティングアルゴリズムは商用並列計算機へ適用されている。しかし，網内の物理/仮想チャネルが十分に備えられたとき，チャネルを選択する適応性が増すことになり，デッドロックはまれにしか発生しないという研究[19),27)]が報告された。

最近，デッドロック回避の手法と対照に，ルーティングの規則を制限せずに，デッドロックが発生したならば，これを検出/解消し，パケットの転送を再開する回復のための手法[1)–3),12),14),15),22)]がある（付録 B）。回復のためのルーティングの目的は，網にデッドロックが発生しないとき，網内の物理/仮想チャネルを最大限に利用することにより，ルーティング性能を最適化することである。この利点は。真の完全適応ルーティング，及び任意の網またはスイッチング技術が適用できることである。またデッドロックを扱うための仮想チャネルが必要ない，単純なルータ設計ができることが挙げられる。

更に，システム稼働中に発生した故障要素の原因による，メッセージ転送の途中で分断されたこのメッセージの回復は，システムのスループットや信頼性の向上のためにルーティングで重要となる。そして，このようなメッセージをどうのように回復するかについて，デッドロック回復及びフォールトトレラント・ルーティングによる研究[12),14)]がなされている。また，これらの研究は，動的故障（付録 C）からのメッセージレベル回復方法を扱ったフォールトトレラント・ルーティングの研究[8),13)]に含まれる。

本稿では，従来の相互結合網におけるデッドロック回復に関する適切な分類が存在していないことより，先行研究を調査し，これらを整理して，新たな分類を提案する。これにより，系統的かつ包括してデッドロック回復を扱うことができる。本稿の構成は以下のようになっている。2 節で準備を述べる。3 節でデッドロック回復の研究を示し，新たな分類基準を導入する。4 節でデッドロック回復とフォールトトレランスの関係について述べる。5 節で今後の課題を述べる。そして，最後にまとめる。

* 情報工学科　講師
平成 15 年 10 月 10 日受付

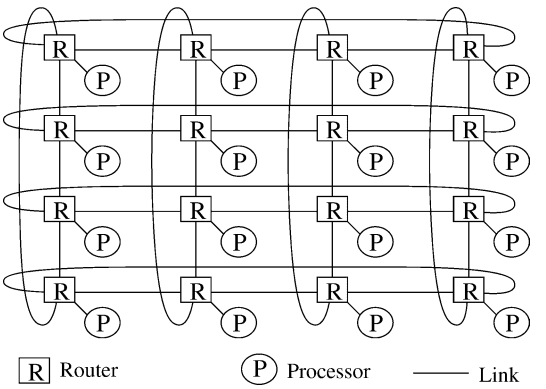


図1　An example of network configuration.

2. 準　　備

この節では，ネットワークモデルとスイッチング技術，デッドロックについて述べる。

- **ネットワークモデル：** 以下に述べるデッドロック回復は任意のネットワークへ適用できるが，ここでは，説明を簡単にするためトーラス（k-ary n-cube）を対象にする。図1にその例を示す。簡単のために，各ノードはルータとプロセッサからなるとする。ルータとプロセッサ，及び隣接ルータ間は単方向の物理チャネル（リンク）により連結する[9]。
- **スイッチング技術：** スイッチング方式はワームホール[9]とする。パケットはフリットを単位として転送される。ルーティングはヘッダによりチャネルを選択し，経路を統制する。そのヘッダからそれに続くフリットはパイプライン状に，出発ノードから目的ノードへ転送される。各ノード（出発ノード）は，任意の長さの複数のパケットを，任意の割合で生成し，かつ，任意のそれ以外のノード（目的ノード）へ転送する。目的ノードへ到着した1つのフリットは随時その目的ノードへ取り込まれる[9),17)]。
- **デッドロック：** 与えられたネットワーク上のルーティングにおいて，デッドロックとは，1つ以上のメッセージ（パケット）から生成する循環チャネル依存関係，つまり，サイクルが存在している状況である[27)]。

3. デッドロック回復

ここでは，デッドロック回復の概要とその分類を示す。更に，この分類に基づくデッドロック回復のための各種ルーティング手法について示す。

3.1 デッドロック回復の概要

デッドロック回復のためのルーティングとは，デッドロックを検出する機構を設けることにより，デッドロックを検出して，デッドロックのない状態へルーティングを回復するものである[13),14),18),20),22),26)]。またこれは，デッドロックが発生していない時のルーティングの性能を最適化することを目的としている。この目的は，全ての物理チャネルおよび仮想チャネルに対して真の完全適応ルーティング（非制限）を許すことによって，また，デッドロックが発生した場合にこれを効果的に処理することによって達成される。デッドロック回復ルーティングは，起こり得るデッドロックの状況に対して常時ある数の物理チャネル及び仮想チャネルを利用しルーティングを制限するという，デッドロック防止および回避ルーティング[9]（制限）と対照するものである。

デッドロック回復ルーティングの特徴や，利点，欠点を集約すると，それぞれ以下のことが挙げられる。まず初めに特徴を示す。デッドロックを検出する機構を設けてこれを検出する。そして，このデッドロックに巻き込まれているメッセージの回復処理をする。ルーティングを制限せず，多様なルーティングのアルゴリズムとの併用ができる。次に利点を示す。真の完全適応ルーティングが許される；デッドロックの回復処理のために仮想チャネルや物理チャネルは必要ない；任意のトポロジおよびスイッチング技術が適用できる；高速性かつ柔軟性のあるルータの単純な設計ができる；ルータのハードウェアオーバヘッドが抑えることができる。欠点としては，回復のためのルーティングの実行性能（処理速度）は，デッドロックが検出される頻度によって決定される。正確にデッドロックを検出できるか否かは，デッドロックを検出する機構に依存する。また，デッドロックを検出する機構としてタイムアウト機構を主に利用しており，そのタイムアウト値を適切に設定することは困難である。性能評価の多くは計算機実験に依存しており，定量的な評価はあまりなされていない，といったことが挙げられる。

3.2 デッドロック回復の分類

図2にデッドロック回復のためのルーティングの分類を示す。デッドロック回復のためのルーティングは，デッドロックが検出される場所により，また，これを検出後にそのメッセージ（パケット）の処理の方法により，以下の4つに分類される。

相互結合網におけるデッドロック回復方法の分類

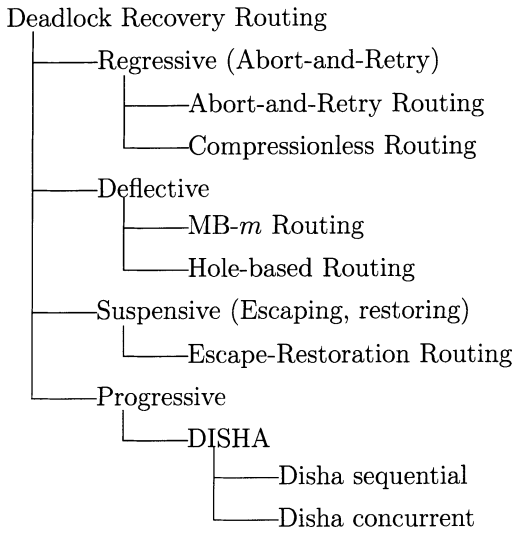


図 2　Classification of deadlock recovery routings.

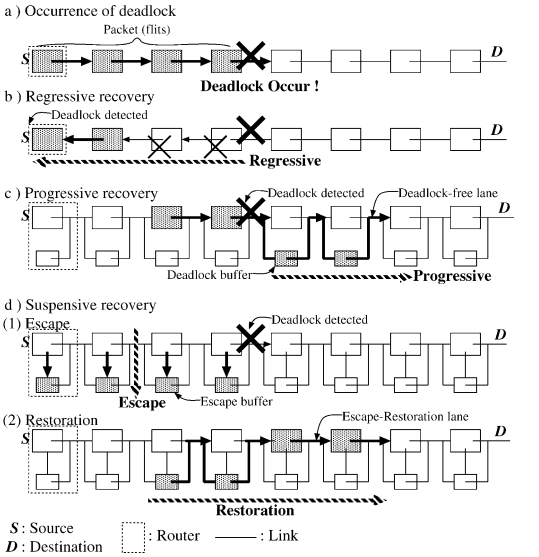


図 3　The routing concepts of deadlock recovery.

- **後退型** (regressive) **の回復**
- **偏向型** (deflective) **の回復**
- **一時停止型** (suspensive) **の回復**
- **前進型** (progressive) **の回復**

以下，簡単にこれらの概要を示す。

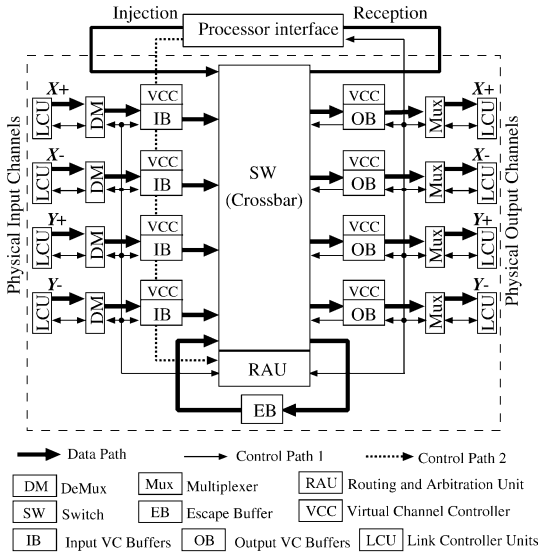
後退型の回復[14),21),22)] **では，デッドロックを検出した後にこれに関係するパケットは出発ノードへ向かって処理される（図** 3(a) **と** (b) **参照）。この回復処理においてデッドロックが検出される場所は出発ノードである。偏向型の回復**[4),12),13),18)] **では，デッドロックを検出した後にそのパケットはミスルートまたはバックトラックにより，デッドロックしたメッセージが処理される。この回復処理においてデッドロックの検出される場所は出発ノードまたは中間ノードである。なお，偏向型の回復は前進型の回復の中に分類される場合もある**[9)]**。前進型の回復**[1)–3),23)] **では，デッドロックを検出した後にそのメッセージは目的ノードへ向かって処理される（図** 3(c) **参照）。この回復処理においてデッドロックの検出される場所は中間ノードである。一時停止型の回復**[25),26)] **は，デッドロックした** 1 **つのパケットの回復処理のために，各ルータ内に設けられた少量の専用バッファへのこのパケットの一時的な退避により，適格パケットの転送が一時停止する。後で退避したパケットの復元により，一時停止した転送が再開する（図** 3(d) **参照）。デッドロックの検出される場所は中間ノードである。**

なお，これらの回復方法は，デッドロックが解消される効果や，単純性，予測性，実用性の観点によりそれぞれ異なることに注意することが必要である。

3.2.1　後退型の回復

後退型の回復方法[14),21),22)] **は，デッドロックに巻き込まれているパケットを削除して，そのパケットの占有している資源の割当が解除される。後にそれらをネットワーク内に再注入して再転送する。この回復方法には，**abort-and-retry routing[21),22)] **と** compressionless routing[14)] **がある。**

これらの回復方法は，多重にパケット転送を中止する可能性があり，削除されたパケットにより予測できない回復遅延を引き起こすという欠点がある。また，この回復処理中に潜在的なネットワークのバンド幅が非効率的に利用されてしまう。更に，この回復処理（パケットのパディング，削除信号，修正されたインジェクタ，受信インタフェースなど）の実装に際して，この必要となる追加的な資源がルータの複雑度を増すこととなる要因となる（図 4, 5, 6 **参照）。したがって，このような機構は実装に際して実用性のないものとなる。しかしながら，この複雑性が増すこととなる主な要因は，バッファのサイズ及びネットワークの直径が増加するのみである。また，大量のバッファが必要となるパケットスイッチイングまたはバッファワームホールスイッチイングを実装しない低次元ネットワークは，大量のバッファを必要とし**



(a) Router node block diagram.

図4　A router block diagram.

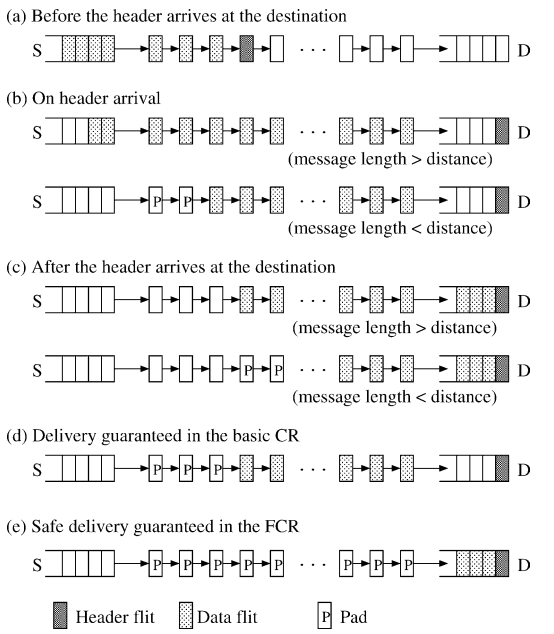


図5　CR with message routing and padding.

ないため有効である。

図7と図8に，compressionless routing **手法のタイムアウト機構のアルゴリズムとこれを実現するための機構を**

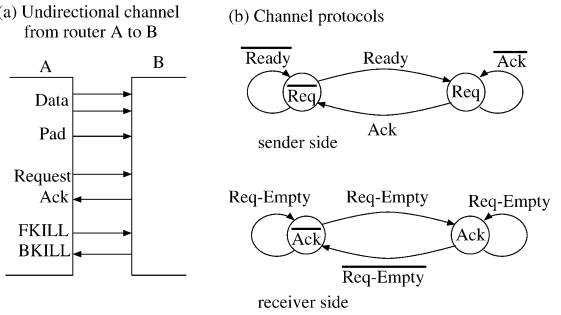


図6　CR property.

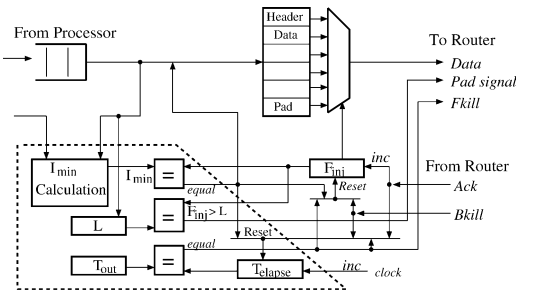
CR Injection Algorithms

1. `If` no message, waite;
2. Move a message into injection buffer;
3. $I_{min} = \max(L, B_{cap} \cdot D)$;
4. `If` $I_{min} \neq L$, pad the massage;
5. `Initialize` $F_{inj} = 0, T_{elapse} = 0$;
6. `While` $T_{elapse} < T_{out}$ `do`;
7. `If` a flit is injected,
8. $F_{inj} = F_{inj} + 1$; $T_{elapse} = 0$;
9. `If` $F_{inj} = L$, assert PAD signal;
10. `If` $F_{inj} > I_{min}$, reset PAD signal;
11. goto 1;
12. `Else` $T_{elapse} = T_{elapse} + 1$;
13. `Enddo`
14. Send out $FKILL$ signal;
15. Wait for T_{gap} before reinjection;
16. Goto 5;

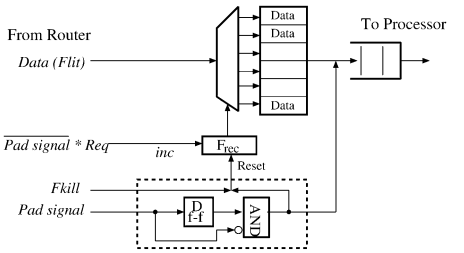
CR Reception Algorithms

1. `If` no message, waite;
2. `Initialize` $F_{rec} = 0$;
3. `While` PAD signal is low `do`;
4. `If` a flit accepted, $F_{rec} = F_{rec} + 1$;
5. `If` $FKILL$ signal is asserted, goto 1;
6. `Enddo`
7. `While` PAD signal is high `do`;
8. Accept and remove a pad flit;
9. `If` $FKILL$ signal is asserted, goto 1;
10. `Enddo`
11. Move the received message to message buffer;
12. Goto 1;

図7.　The time-out mechanism of CR algorithms.



(a) A message injection interface



(b) A message recection interface

図 8. A message injection and reception interface.

参考のため示す。ここで，図 7 中の記号は，D: 目的ノードまでの距離，L: メッセージ長，B_{cap}; チャネルバッファの深さ (flits/channel), I_{min}: 転送を保証する最小フリット注入，T_{out}: タイムアウト値，F_{inj}, F_{rec}: 注入と受信のためのカウンター，T_{elapse}: 直前のフリットが注入された後の経過時間，T_{gap}: 再注入前の待ちサイクルをそれぞれ表す。

3.2.2 偏向型の回復

偏向型の回復方法[4),12),13),18)] は，デッドロックしたパケットをミスルートまたはバックトラックにより，デッドロックからルーティングを回復する。Hole-based routing[4),18)] は，穴（空バッファ）を利用する。これは，デッドロックに巻き込まれている 1 個のパケットを穴へ流し込むことを許す。つまり，ネットワーク内に空バッファ（パケットサイズ分のバッファ容量）を用意して，そこへパケットを伝搬させるルーティングである。このルーティングは，virtual cut-through switching または store-and-forward switching を適用することにより，また，パケットの注入を制限することにより，空バッファの利用を可能としている。この空バッファの利用の方法は，1 個のパケットを穴へ移動させる時に，複数の空バッファがパケット進行の反対方向へそれぞれ伝搬する。そして，その穴は，そのパケットが送信された場所（出発ノード）に置き換えられる。一時的に移動する複数の穴はデッドロックしたルータへ伝搬する。また，一度，1 個の穴がルータへ伝搬すると，そのサイクル依存関係を断ち切るために，デッドロックした（適格な）パケットはその穴へ偏向する。このように，適格なパケットに回復処理を優先させるために，また偏向することによるライブロックを防止するために，その穴の利用に制限があるという欠点がある。また，穴の移動は確率的なため回復遅延は予測できない。この回復方法の空のバッファを利用するという考え方は，後で述べる前進的な回復方法に近いものである。

MB-m routing[12),13)] は，中間ノードにおいてデッドロックが検出される。デッドロックしたパケットのヘッダからその要求している出力チャネル以外のチャネルへある決められた距離 (m) に対してバックトラックの後に，このパケットは目的ノードへ転送される。この回復処理は，後退的な回復のようにパケットを削除しない。しかしながら，バックトラックにおいて，そのパスを探索するための実行遅延がある。この回復方法は，PCS (Pipeline Circuit Switching) に対して，耐故障ルーティングのために開発された。この回復方法のバックトラックの操作は，後退的な回復方法に近いものである。

3.2.3 前進的な回復

前進的な回復方法[1)–3),23)] は，正常な動作をしているパケットから資源の割当を解除して，デッドロックしたパケットをこれ専用の資源（デッドロックバッファ）に再割当する。そして，再割当されたそのパケットは，連結なデッドロックしない回復パス上にアクセスすることにより，サイクル依存関係のないパス上を通り目的ノードへ転送される。この連結なデッドロックしない回復パスの構成方法に関して，制限されたアクセス法 (Disha sequential)[1),2)] 及び構造化されたアクセス法 (Disha concurrent)[3),23)] がある。

図 9 に DISHA のルーティングの手順を示す。これら回復方法の特徴は，連結なデッドロックしない回復パスを形成することである。この連結なデッドロックしない回復パス上へアクセスするときに，サイクル依存関係のないパス，例えば，Hamiltonian Ordering, Eulerian Ordering, Up-Down Direction Ordering などが形成される。そして，この回復パスを通り，デッドロックした全てのパケットは，他のパケットからチャネルを横取りしながらそれぞれの目的ノードへ向かって進行する。仮想チャネ

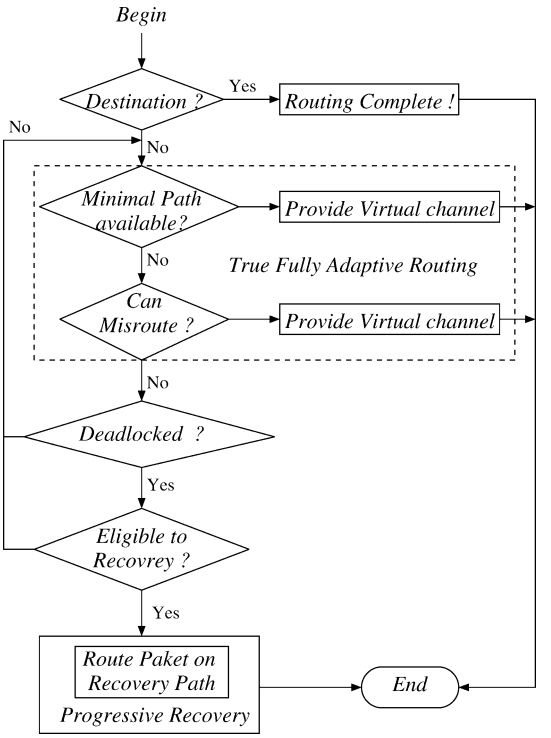


図 9. DISHA routing procedure.

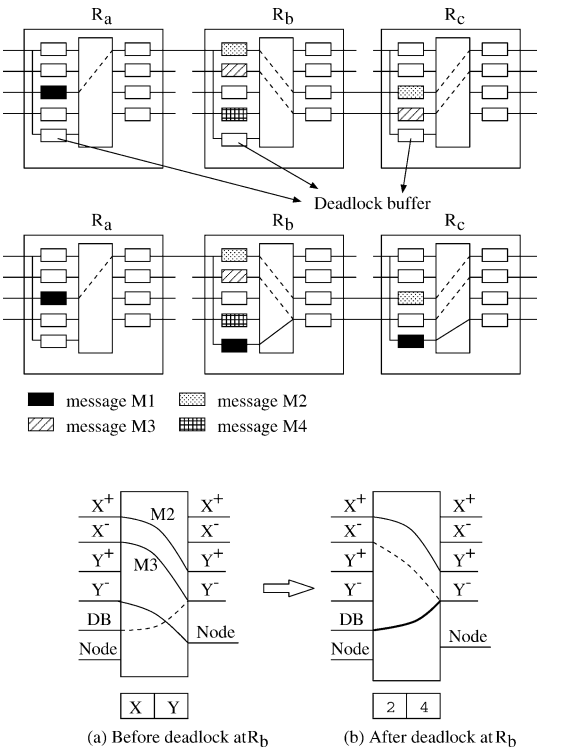


図 10 The deadlock buffer and the reconfiguration of input-outpts connection.

ル及び物理チャネルを特別に利用することにより，この回復パスを実装する必要がないという利点もある。もしデッドロックが発生した場合，他パケットのネットワーク資源を横取りすることにより，チャネルのバンド幅の動的な割当が許される。したがって，適格なデッドロックが検出された時，デッドロックを処理するためにネットワークのバンド幅が割当られる（図 10 参照）。そうでない時，全てのネットワークのバンド幅はパケットの真の完全適応ルーティングのために効果的に利用される。この回復方法は， wormhole routing switching **及び** virtual cut-through switching **に適用できる。**

前進型の回復は，チャネルを解放することに関連して，後退型の回復のようなメッセージの削除または再注入する機構を必要としない。また，偏向型の回復のような空のある一時的な移動かつライブロック保護の機構を必要としない。したがって，前進型の回復方法は，ネットワークのバンド幅を最大に利用できる。しかしながら，パケット（メッセージ）長が長い時に，回復処理に時間がかかる問題がある。この問題はネットワークのバンド幅を減少させて，ルーティングの遅延を増加させることとなる。

Time-out Mechanism

1. Initialize $T_{elapse} = 0$;
2. While $T_{elapse} < T_{out}$ do;
3. If the header flit is able to send out,
4. $T_{elapse} = 0$; exit;
5. Else $T_{elapse} = T_{elapse} + 1$;
6. Enddo.

図 11 The time-out mechanism at an intermediate node.

図 11 は，中間ノードにおいて，デッドロックを検出する機構の手順が示されている。ここで， T_{out}**: タイムアウト値，** T_{elapse}**: 直前のフリットが注入された後の経過時間をそれぞれ表す。図 12 により洗練されたデッドロックの 検出機構である** FC3D (Flow Control-based Distributed

Deadlock Detection) を示す[16]。

3.2.4 一時停止型の回復

一時停止型の回復方法[25),26)]は，wormhole routing switching に対して，デッドロックしたパケット（フリット）をデッドロックパケット退避専用のバッファ（退避バッファ）へ一旦格納して（退避させる）その資源の割当を解除する。そして，任意時間経過後に解放されたパスの接続状態および退避させたパケットのフリットからその解放パスおよびパケットを復活させて（復元させる）パケットの転送を途中から再開する。

この手法は，前進型と後退型の回復方法の特徴と機能の一部をそれぞれ有している。つまり，前進型の回復方法のように，退避バッファを設けるという特徴である。また，任意時間経過後にデッドロックしたパケットの転送を途中から再開するという機能は，一方の後退型の回復方法のような再転送を行う機能に近いものである。また，一時停止型の回復手法は，後退型の回復のようなパケットを削除することを排しており，さらに，解放されたパスおよびパケットを再構成（復元）することにも特徴がある。よって，この手法は，退避・復元ルーティング (Escape-Restoration Routing) と呼ばれている。

退避・復元ルーティングは，デッドロックしたパケットのフリットを各ノード内の退避バッファへ一旦格納させることにより，そのパケットの占有しているパスを一旦解放する。したがって，この解放パス（チャネル）が他のパケットに対して利用可能となり，このチャネルの利用率が高められる利点がある。つまり，ルーティングの性能が向上する。しかしながら，解放されたパスの接続状態と退避させたパケットのフリットからその解放パスとパケットを再構成（復元）するという一連の処理は任意時間経過後に実行される。これらの一連の処理のタイミングが適切に設定されなければ，ルーティングの性能を低下させることとなる問題がある。

3.2.5 ソフトウェアによる回復法

メッセージ注入を制限する機能を提案し，ネットワークが飽和することによるシステム性能の低下を避ける手法である[15)]。この手法により，完全適応ルーティングが使用された時でさえ，デッドロックの発生する確率を無視できる値まで減少させている。また，CR と DISHA による機能を発展させたデッドロック検出の機構を提案している。これは隣接ノードのチャネルの利用状況である局所情報を使用して，全てのデッドロックの可能性を検出し，更にデッドロックを検出することが誤りである確率を減少させている。このデッドロック検出の機構で，ネットワークの混雑とブロックを区別している。この機能は単純であり，分散機能も有する。メッセージの要求したチャネルの時間は，全ての出力チャネルで監視されている。1 個のカウンターが各出力チャネルに接続している。このカウンターは，各クロックサイクル時に増加し，1 つのフリットが物理チャネルを渡った時にリセットされるという機構である（図 12 参照）。デッドロック回復は，各ノードにおいてデッドロックしたメッセージを吸収し，後で，その目的ノードへ向けてそのメッセージを再注入する[15)]。利点は，DISHA 法のデッドロックバッファのような付加的なハードウエアを必要としないことである。

4. デッドロック回復とフォールトトレランス

APCS (Acknowledged Pipelined Circuit-Switching)[8),12),13)] は，動的なリンクとノードの障害によるメッセージレベ

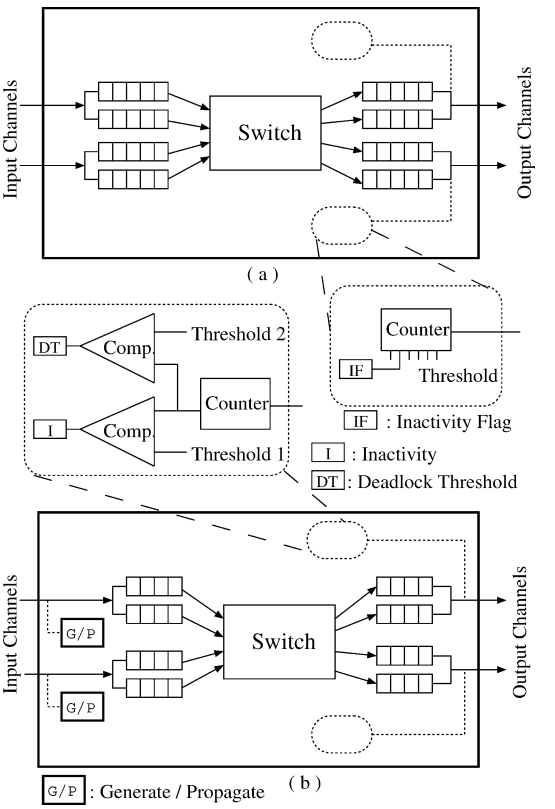


図 12 FC3D.

ルの回復機構及び耐故障性を実現するスイッチング方式である。この回復機構は，タイムアウト機構を必要とせず，また仮想チャネルフローコントロール機構を利用して，故障によって引き起こされるデッドロックからも回復する。

- **特徴：** (1) データフリットが目的ノードに完全に送信された時，最終の ack を出発ノードへ送る。(2) 各物理チャネルに対して 3 つの仮想チャネルに分割し，それぞれ，データフリット用や，ヘッダフリット用，ack フリット用またはバックトラックのヘッダフリット用のチャネルを持つ。このチャネルのどれか 1 つでも故障した時，3 つのチャネルは故障としてマークされる。図 13 には，S から D へメッセージを転送中にリンクの故障により，回復処理が実行される様子が示されている。
- **デッドロック回復機構：** メッセージを転送している最中に，故障要素によって分断されたメッセージを回復する。障害発生時にメッセージの占有しているチャネルを解放するため，リンクコントローラーが，出発ノード（後）と目的ノード（前）にそれぞれ Kill flit や，forward flit, reverse flit の信号を送る。以下の条件でデッドロック回復機能を実現している。
  1. 前 Kill flit が後 Kill flit と衝突した場合，ネットワークから両方の Kill flit を移動する。
  2. 前 Kill flit がメッセージヘッダ（ルーティングヘッダ）と衝突した場合，前 Kill flit とメッセージヘッダを移動する。
  3. 後 Kill flit がその出発ノードに到達した場合，メッセージが転送失敗であることを PE (Processing Element) に知らせるか，または，高レベル耐故障操作を行なう。
  4. 前 Kill flit がその目的ノードに到達した場合，受信した部分メッセージを破棄する。
  5. 前 Kill flit がセットアップ ack と衝突した場合，その ack をネットワークから移動し，その Kill flit は目的ノードに沿って進行する。
  6. 前 Kill flit がメッセージ ack と衝突した場合，両方をネットワークから移動する。
- **利点：** (1) タイムアウト機構を必要としない。(2) 既存のルーティングアルゴリズムと併用で，回復機能を実現できる。(3) 計算機実験から多くのリンクやノードの障害を扱える。
- **欠点：** (1) ack の送受信がトラヒックに及ぼす影響は大きい。(2) 仮想チャネルの制御機能によりハードウェアオーバヘッドは大きくなる。(3) 構成要素の故障検出の機能が必要となる。(4) デッドロック回復ではなく，あくまで，故障要素からのメッセージ回復機構である。

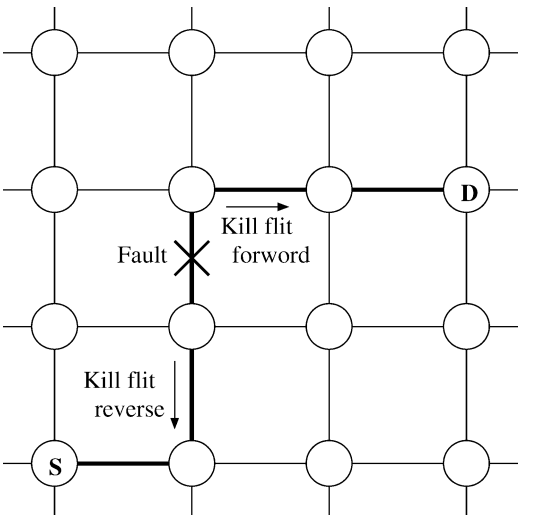


図 13　APSC.

5. むすびと今後の課題

本稿では，従来の相互結合網におけるデッドロック回復に関する先行研究を調査し，また。これらを整理することにより新たな分類を導入した。これにより，系統的かつ包括してデッドロック回復の手法や機構を扱うことができる。また本稿では，先行するデッドロック回復の手法により，特に，デッドロック検出の機能について，メッセージの目的ノードへの分布及びメッセージ長に強くは影響されないこと，また，デッドロックしている真のメッセージと長い間ブロックされたメッセージを区別することが可能となっていることが分かった [16]。

今後の課題としては，デッドロック回復の処理性能に関して，具体的には，以下のことが挙げられる。

- FC3D の方法 [16] において，デッドロックの判定と至るまでに多くの条件を通過しなくてはならない。したがって，その分の処理量は多くなり，その判定に時間がかかること
- その判定処理及び判定時間を減らすために，タイムアウト値を適切に設定すること

といったことを解決する課題がある。更に，いつも安定したシステムの動作を得るために，ネットワークにかかる負荷を解析することにより，トラヒックを予め防止す

るといった，ネットワークの安定性を解析することがある。

参考文献

1) K. V. Anjan and T. M. Pinkston, “DISHA: A deadlock recovery scheme for fully adaptive routing,” *Proc. 9th Int’l Parallel Processing Symposium*, pp. 537–543, Apr. 1995.
2) K. V. Anjan and T. M. Pinkston, “An efficient, fully adaptive deadlock recovery scheme: DISHA,” *Proc. 22th Int’l Symposium on Computer Architecture*, pp. 201–210, June 1995.
3) K. V. Anjan, T. M. Pinkston, and J. Duato, “Generalized theory for deadlock-free adaptive routing and its application to Disha concurrent,” *Proc. 10th Int’l Parallel Processing Symposium*, pp. 815–821, Apr. 1996.
4) M. Coli and P. Palazzari, “An adaptive deadlock and livelock free routing algorithm,” *Proc. 3rd Euromicro Workshop on Parallel and Distributed Processing*, pp. 288–295, Jan. 1995.
5) W. J. Dally and H. Aoki, “Deadlock-free adaptive routing in multicomputer networks using virtual channels,” *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 4, pp. 446–475, Apr. 1993.
6) W. J. Dally, “Virtual-channel flow control,” *IEEE Trans. Parallel and Distributed Systems*, vol. 3, no. 2, pp. 194–205, Mar. 1992.
7) W. J. Dally and C. L. Seitz, “Deadlock-free message routing in multiprocessor interconnection networks,” *IEEE Trans. Comput.*, vol. C-36, no. 5, pp. 547–553, May 1987.
8) B. V. Dao, J. Duato and S. Yalamanchili, “Dynamically configurable message flow control for faulttolerant routing,” *IEEE Trans. Parallel and Distributed Systems*, vol. 10, no. 1, pp. 7–22, Jan. 1999.
9) J. Duato, S. Yalamanchili and L. Ni, *Interconnection networks—An engineering approach*, IEEE Computer Society Press, 1997.
10) J. Duato, “A new theory of deadlock-free adaptive routing in wormhole networks,” *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 12, pp. 1320–1331, Dec. 1993.
11) E. Fleury and P. Fraigniaud, “A general theory for deadlock avoidance in wormhole-routing networks,” *IEEE Trans. Parallel and Distributed Systems*, vol. 9, no. 7, pp. 626–638, July 1998.
12) P. T. Gaughan, V. Dao, S. Yalamanchili, and D. E. Schimmel, “Distributed, deadlock-free routing in faulty, pipelined, direct interconnection networks,” *IEEE Trans. Comput.*, vol. 45, no. 6, pp. 651–665, June 1996.
13) P. T. Gaughan and S. Yalamanchili, “A family of fault-tolerant routing protocols for direct multiprocessor networks,” *IEEE Trans. Parallel and Distrib. Syst.*, vol. 6, no. 5, pp. 482–497, May 1995.
14) J. H. Kim, Z. Liu, and A. A. Chien, “Compressionless routing: A framework for adaptive and faulttolerant routing,” *IEEE Trans. Parallel and Distrib. Syst.*, vol. 8, no. 3, pp. 229–244, Mar. 1997.
15) J. M. Martinez, P. Lopez, J. Duato, and T. M. Pinkston, “Software-based deadlock recovery technique for true fully adaptive routing in wormhole networks,” *Proc. 1996 Int’l. Conf. on Parallel Processing*, pp. 182–189, Aug. 1997.
16) J. M. Martinez, P. Lopez, and J. Duato, “Impact of buffer size on the efficiency of deadlock detection,” *Proc. 5th Int’l. Symp. on High Performace Computer Architecture*, pp. 315–318, Jan. 1999.
17) P. Mohapatra, “Wormhole routing techniques for directly connected multicomputer systems,” *ACM Computing Surveys*, vol. 30, no. 3, p. 374–410, Sept. 1998.
18) P. Palazzari and M. Coli, “Virtual cut-through implementation of the HB packet switching routing algorithm,” *Proc. 6th Euromicro Workshop on Parallel and Distributed Processing*, pp. 416–421, Jan. 1998.
19) T. M. Pinkston and S. Warnakulasuriya, “On deadlocks in interconnection networks,” *Proc. 24th Int’l Symp. on Computer Architecture*, June 1997.
20) T. M. Pinkston, “Flexible and efficient routing based on progressive deadlock recovery,” *IEEE Trans. Comput.*, vol. 48, no. 7, pp. 649–669, July 1999.
21) D. S. Reeves and E. F. Gehringer, “Adaptive routing for hypercube multiprocessors: A performance study,” *Int’l. J. High-Speed Computing*, vol. 6, no. 1, pp. 1–29, Jan. 1994.
22) D. S. Reeves, E. F. Gehringer and A. Chandiramni, “Adaptive routing and deadlock recovery: A simulations study,” *Proc. 4th Conf. on Hypercube Concurrent Computers and Applications*, pp. 331–337, Mar. 1989.
23) F. Silla, A. Robles, and J. Duato, “Improving performance of networks of workstations by using Disha concurrent,” *Proc. 1998 Int’l. Conf. on Parallel Processing*, pp. 80–87, Aug. 1998.
24) M. Singhal, “Deadlock detection in distributed systems,” *IEEE Comput.*, vol. 22, no. 11, pp. 37–48, Nov. 1989.
25) T. Takabatake, M. Kitakami, and H. Ito, “Escape and restoration routing: Suspensive deadlock recovery in interconnection networks,” *IEICE Trans. Inf. & Syst.*, vol. E85-D, no. 5, May 2002.
26) T. Takabatake, M. Kitakami, and H. Ito, “Escape and restoration routing: Suspensive deadlock recovery in interconnection networks,” *Proc. 2001 Pacific Rim Int’l. Symp. Dependable Computing (PRDC’2001)*, pp. 127–134, Dec. 2001.
27) S. Warnakulasuriya and T. M. Pinkston, “Characterization of deadlocks in interconnection networks,” *Proc. 11th Int’l. Parallel Processing Symposium*, Apr. 1997.
28) P. Pramanik and P. K. Das, “A deadlock-free communica-

tion kernel for loop architecture," *Information Processing Letters*, Vol. 38, No. 3, pp. 157–161, May 1991.

A. 代表的なルーティング方式

回線交換 (Circuit Switching)： 一度物理リンクが確立して，それが明示的に解放されるまで保持される。送受信ノード間に物理リンクを確立するまでに時間がかかる。大量データの高速転送に適している。また，結合網の接続形態を長時間変更しないような応用に有効である。

パケット交換 (Paket Switching)： 送受信ノード間に物理リンクを確立せず，メッセージの送受信ができる。1つのパケットの転送が終了すると，使用された物理リンクは解放され，他のパケットが利用できる。以下に代表的な三つの手法の概略を示す。

- **蓄積交換** (Store and Forward)： 中継ノード間はパケットを単位としてメッセージ転送が行なわれる。各ノード内に1パケット分のバッファがいくつか用意される。ノードに送られてくるパケットが全てそのバッファに格納された後に，次のノードへそのパケットは転送される。各ノードに少なくとも1パケット分のバッファが必要であり，ハードウエア量が多くなる。メッセージが多く短い時に有効である。メッセージ転送遅延は到達ノード間の距離に比例する。
- **バーチャルカットスルー** (Virtual Cut-Through)： ワームホールでノード間のデータ転送はフリット単位で行なわれる。出力チャネルがブロックされている場合に，そのパケット（フリット）を一旦退避用のバッファへ格納し，それまで獲得してきた入出力チャネルを後尾フリットから解放し，そのチャネルを他のメッセージに渡す。各ノードには少なくとも1パケット分のバッファが必要であり，ハードウエア量が多くなる。メッセージがノードでバッファされる時，ネットワークの負荷に比例してネットワークバンド幅を消費する。つまり，負荷が低い時は，WHのような振る舞いと性能を示し，負荷が高い時は，ほぼパケット交換のような働きと性能を示す。
- **ワームホール** (Wormhole)： パケットはさらにフリット単位に分割され，同じ経路上をパイプラインのように次々に転送される。先頭フリット（ヘッダ）によって出力チャネルが選択され，後続フリットはそれを使用し，最後尾フリットがチャネルを解放する。パケットのバッファリングが行なわれない。各ノードに数フリット分のバッファが必要であり，ハードウエア量が少なくてすむ。他のメッセージによりネットワークバンド幅のアクセスが妨げられた時，メッセージが多重のルーター上で占有していたバッファやチャネルをブロックする。平均メッセージ遅延は小さくなるが，メッセージ遅延の変化は大きくなる。バッファ要求が少ないため，ネットワークでの衝突は，ネットワークの一部において，メッセージ遅延をかなり増加させる。

B. デッドロック検出と回復

概要： デッドロック検出は，主にタイムアウト機構が使用される。これは閾値を設定し，その設定値とメッセージがブロックされている時間との比較によりデッドロックを判定する。メッセージ（フリット）の転送がチャネル間で成功したかどうか決定するリンクレベルとメッセージが受信されたかどうかを決定するノードレベルがある。デッドロック回復は，デッドロックされたパケットに，占有しているメッセージの資源を解放させ，他のメッセージに解放された資源を使用させるという手法である。その特徴は，ルーティング機能は制限されず，仮想チャネルによるハードウエアオーバーヘッドを抑えられる。デッドロックが希にしか起こらない場合に有効である。

デッドロック検出アルゴリズム： 分散処理システムの研究分野で，デッドロック検出アルゴリズムがある[24),28)]。有向TWF (Transaction-Wait-For) グラフをモデルとし，ノード（サイト）はトランザクション（プロセスと資源）を表す。このTWFグラフはトランザクションの依存関係の状態を表し，TWFグラフに有向サイクルが存在している時のみ，デッドロックと見なされる。このサイクルを検出するアルゴリズムに対して，集中的，分散的，階層的デッドロック検出アルゴリズムの三つに分類されている。

- **集中的デッドロック検出アルゴリズム：** 1つの制御サイトが，グラフの大域状態を構築し，待ちサイクルを探す。
- **分散的デッドロック検出アルゴリズム：** 大域デッドロック検出の役割を多くのサイトで共有する。
- **階層的デッドロック検出アルゴリズム：** サイトは階層的に配置され，デッドロックはその局所的なクラスタの中で検出される。

C. 故障モデル

相互結合網のルーティングでは，故障検出の機能を前提とし，故障要素はノードとリンクである。1個のノー

ド故障の場合，その隣接するリンクは故障としてマークされる。リンク故障の場合，更に物理チャネル故障と仮想チャネル故障の場合を考慮しなければならない。

- **静的故障モデル：** ルーティングの前に，故障要素が分かっている。仮想チャネルを渡るメッセージヘッダは，ルーティングする前にそのチャネルが故障であることが知られている。よって，複数のメッセージは，1本の仮想チャネルの故障や混雑の状態に基づいて，適応的にルーティング可能である。
- **動的故障モデル：** ルーティング中の任意時に，構成要素に故障や障害が発生する。仮想チャネルが任意時に故障した場合，メッセージの進行が中断する。ヘッダフリットのみルーティング情報を含んでいるので，この故障チャネルは，ソースノードに近いデータフリットをブロックし，ルーティングされない。デッドロック回避のために，中断されたメッセージに占有された資源は解放される。そのメッセージは，出発ノードから再転送しなければならない。