論　文

全到達可能性によるハイパキューブの耐故障経路選択算法

金子　敬一[†]　　伊藤　秀男[†]

A Fault-Tolerant Routing Algorithm for Hypercube Systems Based on Full Reachability

Keiichi KANEKO[†] and Hideo ITO[†]

あらまし　故障したノードをもつハイパキューブシステムでは，各ノードが隣接するノードの状態を分類して保持しておくことで，効率の良い動的経路選択法が実現されている．本論文において，我々はハミング距離 l のすべての非故障ノードに長さ l の経路で到達できるという全到達可能性と呼ぶ概念を導入して，従来の算法を改良した新しい算法を提案し，その評価結果を報告する．我々の算法は，各ノードにその隣接ノードの全到達可能性に関する簡単な分類情報をもたせることを特徴とする．シミュレーション実験の結果，従来の算法では検出できなかった最短経路を発見することができることがわかった．また，低次元のハイパキューブにおいては，最小の記憶容量を利用するだけで十分な効果を得ることが判明した．

キーワード　ハイパキューブ，経路選択算法，耐故障性，全到達可能性，安全ノード

1. まえがき

近年，並列処理に対する関心が急速に高まり，さまざまな方式の並列処理システムが出現している．最近の研究では，非常に多数のプロセッサを結合する，いわゆる超並列処理を目指したものも多い．しかしながら，プロセッサ数が大きくなるにつれ，故障プロセッサの発生確率も上昇する．このため故障プロセッサをう回する通信経路を構成することが不可欠となる．

本研究では，対象とする並列処理システムとして規則的な構造と比較的小さな直径をもつハイパキューブシステム（hypercube system）[3],[5],[6]を選択する．故障プロセッサ発生後に起きるシステムの性能低下を抑えるために，複数の故障プロセッサをもつハイパキューブシステムにおける，プロセッサ間のメッセージ通信時の動的経路選択の方法に着目する．

並列処理システムでは，プロセッサ間で通信を実行するときにシステム内に故障プロセッサがある場合，目的プロセッサまでの最短な経路を選択することが重要である．この際，システム内のすべてのプロセッサが，それ自身を含めたシステム内のすべてのプロセッサの状態を把握することができれば，最適な経路選択は可能である．しかしながら，各プロセッサに対する計算量の制約により，この方法を実行することは，困難である．ハイパキューブシステムでは，プロセッサを表す各ノードを，隣接するノードの状態で分類しておき，各ノードに隣接ノードの分類情報を保持させることで，効率的な経路選択算法が提案されている[2],[4]．Chiuらの算法[2]は，Leeらの算法[4]を改良したものである．しかしながら，Chiuらの算法におけるノードの分類法は単純なため，最短経路を発見できない場合がある．このため本研究では，更に全到達可能性（full reachability）と呼ぶ概念を導入して，これによる分類情報もノードに保持させることで，経路選択算法の改善を図る．

2. 経路選択算法

2.1 経路選択問題

[定義 1]　（d 次元ハイパキューブ）

d 次元ハイパキューブシステムは 2^d 個のノードからなり，各ノードのアドレスは d ビットの2進数で表現される．ノード n から，そのアドレスの各ビットを反転した d 個のノードへの枝が存在する．

ノード n_1，n_2 間のハミング距離（Hamming distance）を $H(n_1, n_2)$ が与えることとすると，二つの

[†] 千葉大学工学部情報工学科，千葉市
Faculty of Engineering, Chiba University, Inage-ku, Chiba-shi, 263-8522 Japan

ノード n_1，n_2 間の最短路の長さは $H(n_1,n_2)$ となる．

出発ノード（source node）s から目的ノード（target node）t へメッセージを送信する場合を考える．まずノード s の隣接ノードの集合を $N(s)$ と表す．

$$N(s)=\{n \mid H(s,n)=1\}$$

更に，隣接ノードのうち目的ノードへ近づくことのできるものの集合を $D(s,t)$ と表す．

$$D(s,t)=\{n \mid H(n,t)=H(s,t)-1, n\in N(s)\}$$

このとき $|D(s,t)|=H(s,t)$ であり，$D(s,t)$ に含まれるノードのうち任意のノード c にメッセージと t のアドレスを送る．ノード c を新たに出発ノード s として，目的ノードに到達するまで以上の手続きを繰り返すことでメッセージを送信することが可能となる．

故障ノード（faulty node）をもつハイパキューブシステムでは，出発ノードから目的ノードへの経路のうち故障ノードを含まない経路があるとき，到達可能であると言い，その経路を見つけてメッセージを送信する必要がある．この際，各ノードは隣接するノードの情報を保持しておき，この情報と目的ノードのアドレスとを利用して通信のための経路を動的に選択することができる．また，最短経路を発見できない場合でも，う回経路を発見する必要がある．このための算法が経路選択算法であり，経路を発見できた場合に通信可能であると言う．このとき，できるだけ簡単な情報を保持しつつ，なるべく多くの場合に最短経路を発見できるような算法であることが望ましい．

2.2 Chiu らの算法

Chiu らの算法 [2] では，まずハイパキューブシステム内の非故障ノードを安全ノード（safe node）と危険ノード（unsafe node）に分類して，更に危険ノードを常危険（ordinary unsafe）なものと強危険（strongly unsafe）なものに分類する．ここでは，彼らの主な定義と定理を示した後，その算法を示す．

[定義 2]（安全ノードと危険ノード）

非故障ノード n に対して，以下の条件のいずれかが成立するとき，n を危険ノードと呼ぶ．

- n が二つ以上の故障ノードと隣接する．
- n の隣接ノードのうち故障ノードと危険ノードを合わせた数が 3 以上となる．

非故障ノード n が危険ノードでなければ安全ノードと呼ぶ．

言い換えれば，安全ノードとは，たかだか 2 個の非安全ノードと隣接するノードであり，かつ，2 個の非安全ノードと隣接している場合でも，その両方がともに故障ノードであるようなことはないノードである．

[定義 3]（強危険ノードと常危険ノード）

危険ノード n に隣接するすべてのノードが，危険ノードか故障ノードならば n を強危険ノードと呼ぶ．危険ノード n が強危険ノードでなければ常危険ノードと呼ぶ．

常危険ノードとは，危険ノードであり，かつ安全ノードと隣接するノードである．また，強危険ノードは，常危険ノードではない危険ノードである．

以上の定義に基づくノードの分類例を図 1 に示す．このときハイパキューブシステムの性質から，以下の二つの定理が成り立つ [2]．

[定理 1] 出発ノード s，目的ノード t のいずれかが安全であれば，長さ $H(s,t)$ なる最短経路でメッセージを通信することができる．

[定理 2] 出発ノード s が常危険ノードで，目的ノード t が危険ノードならば，長さがたかだか $H(s,t)+2$ の経路で通信可能である．

定理 1 は，出発ノード s が安全ノードであり，目的ノード t とのハミング距離 $H(s,t)$ が 3 以上ならば，隣接ノードのうち目的ノードへ近づくことのできるものの集合 $D(s,t)$ に必ず安全ノードが含まれるという性質による．これから，出発ノードが安全ノードでなくとも，$D(s,t)$ に安全ノードが含まれていれば，直ちにそのノードへメッセージを送ることで，以降は最短経路で送信できることになる．また，定理 2 は，出発ノード s が常危険ノードであれば，その隣接ノード $N(s)$ に安全ノードが含まれているので，そのノード

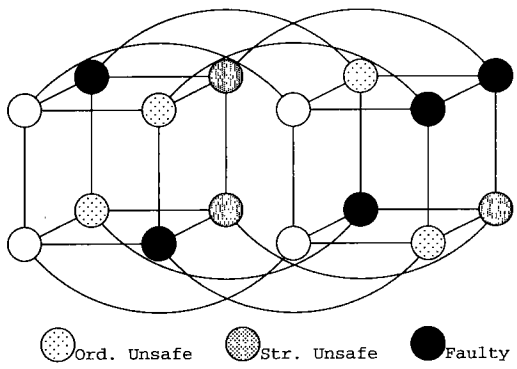

図 1 Chiu らによるノードの分類
Fig. 1 Node classification by Chiu, et al.

```
procedure route(c, t)
begin
  l := H(c, t); N := N(c); D := D(c, t);
  if l = 0 then deliver the message to c and exit
  else if ∃n ∈ D∩S then nxt := n
  (* a comment line for future replacement *)
  else if ∃n ∈ D∩Ū then nxt := n
  else if ∃n ∈ D∩Ū and (c ∈ Ũ or l ≤ 2) then
       nxt := n
  else if ∃n ∈ (N−D)∩S then nxt := n
  else if ∃n ∈ (N−D)∩Ū then nxt := n
  else error('unable to deliver the message');
  route(nxt, t)
end
```

図2 Chiuらによる経路選択法 route
Fig. 2 Routing algorithm route by Chiu, et al.

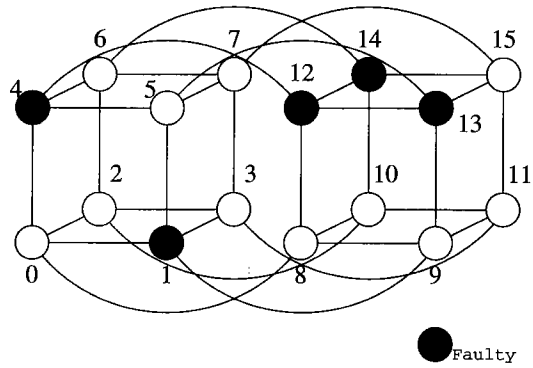

図3 故障ノードをもつ4次元ハイパキューブ
Fig. 3 An example of 4-cube with faulty nodes.

にメッセージを送ることで，以降は最短経路で送信できることになり，最悪でも長さ $H(s,t)+2$ の経路でメッセージを送信できることを示している．

またハイパキューブシステムの性質として以下の全危険性（full unsafeness）を定義する．

[定義 4]（全危険ネットワーク）

ハイパキューブシステムにおいて，すべての非故障ノードが危険ノードであるならば，そのハイパキューブは全危険であると言う．

このとき次の定理が成り立つ[2]．

[定理 3] 全危険ではないハイパキューブでは，強危険ノードは常危険ノードと隣接する．従って，全危険ではないハイパキューブにおいて，出発ノード s が強危険ノードで，目的ノード t が危険ノードならば，長さがたかだか $H(s,t)+4$ の経路で通信可能である．

d 次元ハイパキューブが全危険でないと仮定すると定理 1，2，3 からハイパキューブの各ノードは隣接ノードと情報を交換して，隣接ノードを安全ノード，常危険ノード，強危険ノード，故障ノードのいずれかに分類しておくことで，効率的な経路選択を実現することができる．

以下，安全ノードの集合を S，常危険ノードの集合を $\bar{U}$，強危険ノードの集合を $\tilde{U}$ で表すこととする．Chiuらの算法を図2に示し，これを route と呼ぶこととする．

3. 全到達可能性に基づく算法

3.1 距離に関する安全ノード

我々は，メッセージの配送のために経由すべきノードを隣接ノードの集合から選択する際に，目的ノードまでの距離の情報も利用することができる点に着目する．このため，あるノードが，そのノードから特定の距離にある非故障ノードすべてに最短経路で到達可能であるか否かという性質を定義する．

[定義 5]（距離に関する全到達可能性）

故障ノードをもつハイパキューブにおいて，ある非故障ノード n からハミング距離 l のすべての非故障ノードに対し，長さ l の経路で到達可能ならば，ノード n は距離 l に関して全到達可能（fully reachable）であると言う．

ハイパキューブ内の非故障ノードのうち，距離 l に関して全到達可能なノードの集合を，R_l とおく．目的ノードまでのハミング距離が $l+1$ であるようなノード n が，メッセージを受け取って経路を選択するとき，その隣接ノードの各々について R_l に属するか否かを知ることができれば，経路選択において不必要にう回することを回避できる．しかしながら，それを知ることは難しいので，R_l の近似を定義する．

[定義 6]（距離に関する安全ノード）

すべての非故障ノードを距離1に関して安全なノードと呼ぶ．このとき，距離 $l-1$ に関して安全なノードと $d-l+1$ 個以上隣接する非故障ノードを距離 l に関して安全なノードと呼ぶ．

以下では距離 l に関して安全なノードの集合を S_l と表すこととする．例えば，4次元ハイパキューブの各ノードに図3のようにアドレスを与え，故障ノードの集合を $\{1,4,12,13,14\}$ としたとき，

$$R_1 = S_1 = \{0,2,3,5,6,7,8,9,10,11,15\},$$
$$R_2 = \{2,3,7,8,10,11,15\},$$
$$S_2 = \{2,3,7,8,10,11\},$$

$R_3 = \{0,2,3,6,7,9,10,11,15\}$,
$S_3 = \{0,2,3,6,9,10,11,15\}$

となる．ノード 15 は，そこからハミング距離 2 の非故障ノード 3，5，6，9，10 に距離 2 の経路で到達可能であるため，距離 2 に関して全到達可能である．しかしながら，距離 2 に関して安全なノードとはなっていない．同様にノード 7 は，距離 3 に関して全到達可能であるが，距離 3 に関して安全ではない．

S_l と R_l に関して，次の定理が成立する．

[定理 4] 任意の距離 l に対して $S_l \subset R_l$.

(証明) l に関する数学的帰納法による．$n \in S_1$ とするとノード n は故障ノードではない．従って $H(n,n')=1$，すなわち隣接する非故障ノード n' に対しては，直ちに長さ 1 の経路で到達できる．故に $S_1 \subset R_1$.

今 $l<k$ について $S_l \subset R_l$ が成立すると仮定する．このとき $n \in S_k$ とすると，定義 6 よりノード n は少なくとも $d-k+1$ 個の距離 $k-1$ に関して安全なノードと隣接する（図 4 参照）．すなわち，距離 $k-1$ に関して安全でない隣接ノードはたかだか $k-1$ 個となる．ここで，$H(n,n')=k$ なる任意の非故障ノード n' に対して，n の隣接ノードのうち，目的ノード n' へ近づくことのできるものの集合 $D(n,n')$ を考える．この $D(n,n')$ に含まれるノードの個数は k である ($|D(n,n')|=k$)．従って，$D(n,n')$ 中の k 個のノードのうち，少なくとも 1 個は距離 $k-1$ に関して安全なノードとなる．帰納法の仮定より $S_{k-1} \subset R_{k-1}$．よって，そのノードを経由すれば長さ k の経路で n' へと到達できる．故に $H(n,n')=k$ なる任意の非故障ノード n' に対して，距離 k の経路で到達可能となる．従って $n \in R_k$．以上から $S_k \subset R_k$ となり，任意の l に対して，$S_l \subset R_l$ となる． □

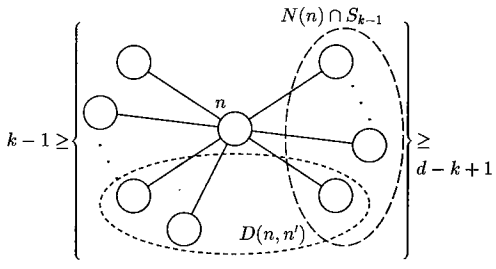

図 4 ノード n ($\in S_k$) とその隣接ノードの関係
Fig. 4 Relationship between node n ($\in S_k$) and its neighbor nodes.

一般にノード n が $N(n) \cap R_l$ を求めることは難しいが，S_l は隣接ノードとの情報交換だけで決定できるため簡単に得られる．定理 4 より $S_l \subset R_l$ なので，メッセージを経由しても安全なノードの集合として，S_l を経路選択に利用できる．前処理として，$S_2,\cdots,S_k$ を求めておけば，図 2 のコメント部分 (*…*) を

```
else if l ≦ k+1 and ∃n ∈ D ∩ S_{l-1} then
    nxt := n
```

と変更して新たな経路選択法 FR を得る（図 5 参照）．

算法 FR では，次の定理 5 が示すように，算法 route よりも安全に経由できるノードが増えている．

[定理 5] 任意の距離 l に対して $S \subset S_l$.

(証明) l に関する数学的帰納法による．$S \subset S_1$ は明らか．$n \in S$ とするとノード n は故障ノードとたかだか 1 個しか隣接しないので定義 6 より $n \in S_2$．よって $S \subset S_2$.

今 $l<k$ について $S \subset S_l$ が成立すると仮定する．このとき $n \in S$ とすると，n が安全ノードであることからたかだか 2 個の非安全ノードと隣接する．よってノード n は，少なくとも $d-2$ 個の安全ノードと隣接する ($|N(n) \cap S| \geq d-2$)．帰納法の仮定より $S \subset S_{k-1}$．従って，$N(n) \cap S \subset N(n) \cap S_{k-1}$ となり，ノード n は，少なくとも $d-2$ 個の距離 $k-1$ に関して安全なノードと隣接する ($|N(n) \cap S_{k-1}| \geq d-2$)．今，$k \geq 3$ より，$d-2 \geq d-k+1$．よってノード n は，少なくとも $d-k+1$ 個の距離 $k-1$ に関して安全なノードと隣接するので，定義 6 より $n \in S_k$．従って $S \subset S_k$.

```
procedure FR(c, t)
begin
  l := H(c, t); N := N(c); D := D(c, t);
  if l = 0 then deliver the message to c and exit
  else if ∃n ∈ D ∩ S then nxt := n
  else if l ≦ k+1 and ∃n ∈ D ∩ S_{l-1} then
        nxt := n
  else if ∃n ∈ D ∩ Ū then nxt := n
  else if ∃n ∈ D ∩ Ũ and (c ∈ Ũ or l ≦ 2) then
        nxt := n
  else if ∃n ∈ (N − D) ∩ S then nxt := n
  else if ∃n ∈ (N − D) ∩ Ũ then nxt:= n
  else error('unable to deliver the message');
  FR(nxt, t)
end
```

図 5 全到達可能性に基づく経路選択算法 FR
Fig. 5 Routing algorithm FR based on full reachability.

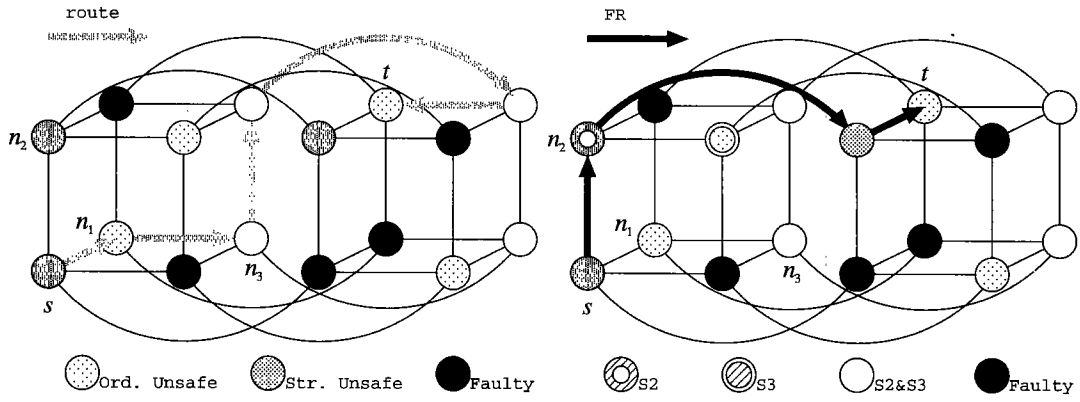

図 6 route と FR による経路選択例
Fig. 6 A routing example by route and FR.

故に任意の l に対して，$S \subset S_l$ となる． □

図 6 に示す例では，出発ノード s が強危険であるため，算法 **route** は，隣接ノードのうち目的ノード t へ近づくことのできるノードの集合 $D(s,t)$ の中で，唯一つの常危険ノード n_1 へメッセージを転送する．しかしながら，$D(n_1,t)$ はすべて故障ノードなので，安全ノード n_3 へとう回している．一方，算法 **FR** では，$D(s,t)$ の中で，**route** では強危険ノードとされたノード n_2 が，距離 2 に関して安全ノードとなる．従って，距離に基づく全到達可能情報より矢印で示すように最短経路を選択することが可能となる．

3.2 初 期 化

本節では，算法 **FR** で，故障ノードをもつハイパキューブシステムの各非故障ノードが，隣接ノードの情報を保持するための算法を示す．各ノードは隣接ノードとの間のリンクにバッファをもち，メッセージの送信と受信は一定時間で実行できるものとする．また，一定時間で故障隣接ノードの検出も可能であると仮定する．

図 7 に **FR** のためのノード c における初期化手続き **init** を示す．変数 $\sigma_{n,l}$ は，ノード n の距離 l に関する分類を保持する．また，変数 T_{l-1} は，ノード c の隣接ノードのうち距離 $l-1$ に関して安全なノードの集合を表しており，この集合の濃度によって，$\sigma_{c,l}$ の値を決定している．

この手続きは，算法 **route** のための初期化手続きに加えて実行する必要がある．しかしながら，算法 **route** のための初期化手続きの時間計算量は $O(d^2)$ であり [2]，手続き **init** の時間計算量は $O(kd)$ であることから，全体の計算量を悪化させることはない．

```
procedure init(c, k)
begin
  σ_{c,1} := SAFE;
  Detect N(c) ∩ S_1;
  for l := 2 to k do
    begin
      send σ_{c,l-1} to N(c) ∩ S_1;
      for every n ∈ N(c) ∩ S_l do receive σ_{n,l-1} from n;
      T_{l-1} := {n | n ∈ N(c) ∩ S_l, σ_{n,l-1} = SAFE}
      if |T_{l-1}| ≥ d − l + 1 then σ_{c,l} := SAFE
      else σ_{c,l} := UNSAFE
    end
end
```

図 7 経路選択法 FR の初期化手続き init
Fig. 7 Initialization procedure init for routing algorithm FR.

4. 評 価

4.1 計 算 量

d 次元ハイパキューブを考える．Chiu らの算法では，各ノードが隣接ノードの情報を保持するために必要な記憶領域量は $O(d)$ である．一方，算法 **FR** では，$O(kd)$ を必要とする．但し，k は 3 以上 $d-1$ 以下の整数であり，ノードにおいて利用可能な記憶領域量に応じて選択できるパラメータである．また，Chiu らの算法では，各ノードにおける経路選択に $O(l)$ の時間計算量を必要とする．但し，l は目的ノードまでのハミング距離である．従って，全体として経路選択に $O(d^2)$ かかる．我々の算法でも同様に $O(d^2)$ が必要である．

4.2 シミュレーション実験

まず算法 **FR** の能力を検証するため，以下の手順を繰り返して，ネットワークが全危険にならないような

故障ノードのすべての配置と目的ノードの位置のすべての組合せに対して，シミュレーション実験を行う．

（1） d 次元ハイパキューブシステムにおいて，その対称性から出発ノードを常にアドレス0のノードとする．

（2） システム内に，出発ノードを除いて f 個の故障ノードを設定する．

（3） 各ノードを，故障，安全，常危険，強危険に分類する．また，S_l $(l=1,\cdots,k)$ を求める．この際，ネットワークが全危険になるものは以下の対象から除外する．

（4） 出発ノード以外の非故障ノードを目的ノードとする．

（5） route と FR を呼び出して，それぞれについて，最短経路でメッセージを送信できずにう回してしまう場合を数える．

結果を表1に示す．但し表の数字は，目的ノードまで最短経路で到達できずに，route がう回してしまったうち，FR によって最短経路で到達できるようになった割合を百分率で表している．d 次元ハイパキューブにおける算法 FR による経路選択では，$k<d$ なる距離 k に関する安全なノードの情報のみを必要とするので，表中で $d\leq k$ なる項目は対象から除外している．結果として route が最短経路で送信できない場合の約25%から70%について FR が解決している．更に算法 FR の場合，改善の度合は，あまり k の値に依存していない．このことから，実験を行った次元の低いハイパキューブでは，FR は，各ノードに十分な記憶領域を確保できない場合でも，比較的小さな k を用いて効果を上げることができるという利点をもつことが言える．

ネットワークが全危険でなければ，route や FR は，必ずメッセージを目的ノードに送信することが可能である．しかし多くの故障ノードが存在する場合，ネットワークが全危険でないことを確かめることは困難である．ネットワークが全危険ならば，route や FR は，メッセージを目的ノードに送信できない可能性がある．その場合は，深さ優先探索など他の経路選択手法に切り換える必要がある [1]．残念ながら，これらの手法は非常に効率が悪いので，できるだけその使用を避けることが望ましい．このため，全危険なネットワークに対しても，算法 FR がどの程度の能力を発揮できるのかを調べるために次の実験を行う．この場合も，故障ノードと目的ノードの位置のすべての組合せに対して，シミュレーション実験を行う．ネットワークが全危険な場合は，常危険ノードが存在せず，route，FR ともにう回は生じない．このため，実験では，目的ノードに到達できるか否かのみを調べる．

（1） d 次元ハイパキューブシステムにおいて，その対称性から出発ノードを常にアドレス0のノードとする．

（2） システム内に，出発ノードを除いて f 個の故障ノードを設定する．

（3） 各ノードを，故障，安全，常危険，強危険に分類する．また，S_l $(l=1,\cdots,k)$ を求める．この際，ネットワークが全危険にならないものは以下の対象から除外する．

（4） 出発ノード以外の非故障ノードを目的ノードとする．

（5） route と FR を呼び出して，メッセージの送信が可能にもかかわらず，送信に失敗する場合を数える．

結果を表2に示す．表の数字は目的ノードまで到達可能にもかかわらず，route が送信に失敗した場合，FR によって，それがどのくらい減ったかを百分率で表している．表1と同様な理由で，$d\leq k$ なる項目は対象から除外している．また，$d=6$，$f=6$ の場合は，算法 route，FR が不必要な送信の失敗を起こさなかっ

表1 算法 FR によるう回の改善率（%，非全危険）
Table 1 Improvement ratio of sidetracks by routing algorithm FR (%, not fully unsafe).

d	f	$k=3$	$k=4$	$k=5$
4	5	25.0	—	—
5	5	51.3	51.3	—
5	6	58.1	58.1	—
5	7	53.2	53.2	—
5	8	44.0	44.0	—
5	9	34.0	34.0	—
6	5	38.7	38.7	38.7
6	6	57.3	57.4	57.4
6	7	69.5	69.7	69.7

表2 算法 FR による失敗の改善率（%，全危険）
Table 2 Improvement ratio of failures by routing algorithm FR (%, fully unsafe).

d	f	$k=3$	$k=4$	$k=5$
4	5	16.5	—	—
5	5	44.4	44.4	—
5	6	41.4	41.7	—
5	7	38.1	38.5	—
5	8	33.1	33.7	—
5	9	27.7	28.5	—
6	6	—	—	—
6	7	63.4	63.8	63.8

たため，測定不能となっている．結果としてrouteがメッセージの送信に失敗する場合の約16%から63%についてFRは送信に成功している．更に全危険なネットワークを含まないときと同様に，算法FRによって改善できる度合は，ほとんどkの値に依存していない．従って，全危険なネットワークに対しても，次元の低いハイパキューブでは，FRは，比較的小さなkでも有効であるという性質を保っていることがわかる．

5. むすび

ハイパキューブシステムにおいて，隣接するノードの状態を全到達可能な距離に基づいて分類することで，より効率的な経路選択算法を提案した．この算法を評価した結果，従来のものでは検出できなかった最短経路を検出することができるようになった．シミュレーション実験により，低次元のハイパキューブでは，各ノードの記憶領域が小さい場合でも，FRが有効であること，良い結果を与えることを確かめた．また，この傾向はネットワークが全危険である場合も成り立つことも判明した．

現在，危険ノードを全到達可能な距離に基づいて分類して，その情報を利用する算法を開発中である．また，現状では計算機の能力によって全数探索によるシミュレーション実験は，ハイパキューブシステムの次元が6，故障数が7程度が限界となっているので，これよりも大きな次数をもつシステムについてのシミュレーション方法を確立する必要がある．更に，実行時に故障ノードが発生したときに各ノードが保持する隣接ノードの情報を更新するための算法を開発することも今後の課題である．

謝辞 千葉大学工学部情報工学科の大豆生田利章助手からは，本研究に関して助言を頂いた．また，同研究生の佐藤貴君には，FR算法において$k=3$の場合を調査して頂いた．ここに感謝する．なお，本研究の一部は（財）栢森情報科学振興財団および文部省科学研究費補助金の援助による．

文　献

[1] M.-S. Chen and K.G. Shin, "Depth-first search approach for fault-tolerant routing in hypercube multicomputers," IEEE Trans. Parallel and Distributed Systems, vol.1, no.2, pp.152–159, April 1990.

[2] G.-M. Chiu and S.-P. Wu, "A fault-tolerant routing strategy in hypercube multicomputers," IEEE Trans. Computers, vol.45, no.2, pp.143–155, Feb. 1996.

[3] W.J. Dally, "Performance analysis of k-ary n-cube interconnection networks," IEEE Trans. Computers, vol.39, no.6, pp.775–785, June 1990.

[4] T.C. Lee and J.P. Hayes, "A fault-tolerant communication scheme for hypercube computers," IEEE Trans. Computers, vol.41, no.10, pp.1242–1256, Oct. 1992.

[5] Y. Saad and M.H. Schultz, "Topological property of hypercubes," IEEE Trans. Computers, vol.37, no.7, pp.867–872, July 1988.

[6] C.L. Seitz, "The cosmic cube," Commun. ACM, vol.28, no.7, pp.22–33, July 1985.

（平成9年11月26日受付，10年2月25日再受付）

金子　敬一　（正員）

昭60東大・工・計数卒．昭62同大大学院修士課程了．同年同大・工・計数助手．平8千葉大・工・情報講師．工博．関数プログラミング言語，部分計算，並列計算等の研究に従事．ACM，情報処理学会，日本ソフトウェア科学会各会員．

伊藤　秀男　（正員）

昭44千葉大・工・電気卒．同年日本電気入社．昭46木更津高専助手．昭48千葉大・工・電子助手．昭61同助教授．平2同情報教授．工博．VLSIの故障検査と検査容易化設計，セルフチェッキングシステム設計，フォールトトレラント並列処理システム設計，VLSIの欠陥回避等の研究に従事．情報処理学会，IEEE Computer Society各会員．