アルゴリズム　１７－６
（1990．９.28）

Information Disseminating Schemes for Fault Tolerance in Computer Networks

計算機ネットワーク上の情報散布方式と耐故障性

Kumiko Kanai, Kinya Miura and Yoshihide Igarashi
金井 久美子、三浦 欽也、五十嵐 善英

Department of Computer Science, Gunma University
Kiryu, 376 Japan
群馬大学工学部情報工学科

あらまし
同期して働く N 個のプロセッサを含むネットワーク上のいくつかの情報散布方式について論じる。ここで考えるネットワークは散布方式に応じてリンクを張るもので、リンクの総数は $\Theta(N \log N)$ である。１方向／ラウンド散布可能な時、Han-Finkel の方式ではネットワークに１個の故障を含む場合、$\lceil \log_2 N \rceil + 2$ ラウンドで情報散布が完了することが示されているが、本論文で提案した方式の１つはこの場合、$\lceil \log_2 N \rceil + 1$ ラウンドで情報散布が完了する。さらに、ここで示した t 方向／ラウンド散布可能な方式の１つは故障の数が t 以下の時、特別な場合を除いて $\lceil \log_{(t+1)} N \rceil + 2$ ラウンドで情報散布が完了することを示す。

Abstract
We describe several information disseminating schemes in computer networks with N processors. These processors are synchronized with a global clock, and are linked according to disseminating schemes. For any of our schemes the total number of links is $\Theta(N \log N)$. Han and Finkel discussed a scheme that can send a message to one direction at each round. They showed that their scheme can broadcast information from any source to all destinations within $\lceil \log_2 N \rceil + 2$ rounds, if the network contains at most one faulty processor. In this case one of our schemes can broadcast information within $\lceil \log_2 N \rceil + 1$ rounds. In the case where each processor can send a message to t directions at each round, one of our schemes can broadcast information within $\lceil \log_{(t+1)} N \rceil + 2$ rounds except for a certain case if the number of faulty processors is at most t.

1. 序

ブロードキャストは情報源のプロセッサからあるメッセージをネットワーク中の他の全てのプロセッサに行き渡らせることである。この研究は並列分散システムの発展と共に盛んに行なわれている [B][FP][HHL]。ブロードキャストは並列分散システムの基本的な操作の一つである。より大規模なシステムほどブロードキャストの重要性が高くなり、システムのための効果的なネットワークモデルとしてハイパーキューブ、メッシュ、CCC などがよくとり上げられている [F][IKM][JH]。また、ネットワーク中に故障したプロセッサや結線が存在する場合にも、いかに正しく効率良くブロードキャストを行なうか、という問題が考えられ、これは Fault-Tolerant 問題と呼ばれ広く知られている [HF][L][RS]。

ここでネットワークモデルとして扱うグラフは、ノード（プロセッサ）の個数を固定しておき エッジ（リンク）については情報散布の方式に応じて適宜張られるものとする。リンクの総数は、ハイパーキューブと同様で $\Theta(N \log N)$ である。

プロセッサは同期して働き、情報の送受という一連の動作を一つの基本動作として行なう。この同期して行なう１回の動作を１ラウンドと呼ぶ。ブロードキャストに必要なラウンド数を情報散布方式の評価基準とする。

故障したプロセッサはメッセージの受信は可能だが送信は不可能であるとする。また故障していても、誤ったメッセージを送信するというような場合はここでは考えない。情報源であるソースは故障していないものと仮定する。

本論文で示す情報散布の方式は、次の２つの条件を満たす。１つめは任意のプロセッサをソースにすることが可能であり、２つめは任意のラウンドからスタートすることが可能である。

第２節では、１ラウンドごとに１方向送受信可能な場合の散布方式をいくつか述べ、第３節でその拡張として１ラウンドごとに t 方向送受信可能な場合の散布方式を述べる。第４節では、ある条件のもとで第２節の散布方式の耐故障性を、また第５節では第３節の散布方式の耐故障性を評価する。第６節はまとめである。

2. １方向／ラウンド送受信可能な散布方式

ここでは、ネットワークのプロセッサ数を N とし、それらは０から $N-1$ まで番号付けされているものとする。

１ラウンドごとに１方向送受信可能な場合の方式を３種類示す。ここでは故障したプロセッサやリンクは考慮せず、耐故障性については第４節で示す。

2.1. １方向／ラウンド送受信可能な散布方式 その１

procedure 1-disseminate1(N)
repeat
  for *round* := 0 **to** $\lceil \log_2 N \rceil - 1$ **do**

各々のプロセッサ i はメッセージを
プロセッサ $((i+2^{round}) \bmod N)$ へ送る。
repeatforever

各々のプロセッサ $i\ (0 \le i \le N-1)$ のメッセージ送り先は プロセッサ $((i+2^{round}) \bmod N)$ であり、*round* の値は 0 から $\lceil \log_2 N \rceil - 1 (= n-1)$ まで連続的に変化する。

1-disseminate1 は [HF] によるもので、任意のソース、任意のラウンドからスタートし、連続した $n\ (= \lceil \log_2 N \rceil)$ 回のラウンドでブロードキャストを完了することができる。以下に、$N=7$ の場合の例を示す。

Example 1. 7個のプロセッサが結合されているネットワークを考える。どのプロセッサ間にリンクが存在するかは Table 1 の散布表に基づいて決定される。

	processors						
round	0	1	2	3	4	5	6
0	1	2	3	4	5	6	0
1	2	3	4	5	6	0	1
2	4	5	6	0	1	2	3

Table 1: $N=7$ の場合の散布表 (散布方式 その1)

ソースをプロセッサ2、スタートラウンド1 とする。まず ラウンド 1 で プロセッサ 2 から プロセッサ 4 へメッセージが送られ、次のラウンド 2 で プロセッサ {2,4} から プロセッサ {6,1} へ送られる。更にラウンド 0 で プロセッサ {1,2,4,6} から プロセッサ {2,3,5,0} へ送られる。こうして全てのプロセッサにメッセージが行き渡りブロードキャストが完了する。 □

2.2. 1方向 / ラウンド送受信可能な場合の散布方式 その2

procedure 1-disseminate2(N)
repeat
for *round* := 0 **to** $r = \lceil \log_2 N \rceil - 1$ **do**
各々のプロセッサ i はメッセージを
プロセッサ $((i+2^{r-round}) \bmod N)$ へ送る。
repeatforever

各プロセッサ $i\ (0 \le i \le N-1)$ のメッセージ送り先は プロセッサ $(i+2^{r-round}) \bmod N$ である。*round* の値は、0 から $\lceil \log_2 N \rceil - 1 (= n-1)$ まで連続的に増加し、1-disseminate2 では変数 r を用いている。この方式は 1-disseminate1 の拡張であるが、ブロードキャストに必要なラウンド数は同様に n 回であり、スタートのラウンドやソースのノードは任意で構わない。

Example 2. 7個のプロセッサが 結合されているネットワークを考える。どのプロセッサ間にリンクが存在するかは Table 2 の散布表に基づいて決定される。

	processors						
round	0	1	2	3	4	5	6
0	4	5	6	0	1	2	3
1	2	3	4	5	6	0	1
2	1	2	3	4	5	6	0

Table 2: $N=7$ の場合の散布表 (散布方式 その2)

ソースをプロセッサ 0、スタートラウンド 0 とする。まずラウンド 0 で プロセッサ 0 からプロセッサ 4 へメッセージが送られ、次のラウンド 1 でプロセッサ {0,4} からプロセッサ {2,6} へメッセージが送られる。さらにラウンド 2 でプロセッサ {0,2,4,6} から {1,3,5,0} へメッセージが送られ、ブロードキャストが完了する。 □

2.3. 1方向 / ラウンド送受信可能な場合の散布方式 その3

procedure 1-disseminate3(N)
$M := N$
for $i := 0$ **to** $\lceil \log_2 N \rceil - 1$ **do**
$M := \lceil M/2 \rceil$
$T[i] := M$
repeat
for *round* := 0 **to** $\lceil \log_2 N \rceil - 1$ **do**
各々のプロセッサ i はメッセージを
プロセッサ $((i+T[round]) \bmod N)$ へ送る。
repeatforever

各々のプロセッサ $i\ (0 \le i \le N-1)$ のメッセージ送り先は プロセッサ $(i+T[round] \bmod N)$ であり、*round* の値は 0 から $\lceil \log_2 N \rceil - 1 (= n-1)$ まで連続的に変化する。配列 T は、各々の N に対して計算される。また、便宜上 $T[-1] = N$ とおくこととする。

Example 3. 9個のプロセッサが 結合されているネットワークを考える。どのプロセッサ間にリンクが存在するかは Table 3 の 散布表に基づいて決定される。

i	0	1	2	3
T[i]	5	3	2	1

	processors								
round	0	1	2	3	4	5	6	7	8
0	5	6	7	8	0	1	2	3	4
1	3	4	5	6	7	8	0	1	2
2	2	3	4	5	6	7	8	0	1
3	1	2	3	4	5	6	7	8	0

Table 3: $N=9$ の場合の散布表 (散布方式 その3)

ソースをプロセッサ4、スタートラウンド 0 とする。まずラウンド 0 でプロセッサ 4 からプロセッサ 0 へメッセージが送られ、次にラウンド 1 でプロセッサ {0,4} から プロセッサ {3,7} へ、またラウンド 2 で {0,3,4,7} から {2,5,6,0}、さらにラウンド 3 で 0 から 1、7 から 8、と送られブロードキャストが完了する。 □

この方式も、1-disseminate1,2 と同様に、スタートのラウンドやソースのノードに依存せず、ブロードキャストに必要なラウンド数は n 回である。以下にそれを示す。

Lemma 1 $0 \le i \le n-1$ なる、任意の i に対して、$0 \le j < T[i-1]$ なる、任意の j を、

$$j = 0[+T[i]][+T[i+1]]\cdots[+T[n-1]]$$

と表すことができる。

但し、[,] で囲まれた項は、オプショナルを表しており、加算するか否かのいずれかである。

Proof. i に関する数学的帰納法で示す。

1. $i=n-1$ のとき、$0 \le j < T[i-1] = T[n-2] = 2$ より $j=0$ または 1。明らかに $T[n-1]=1$ であるから、これは、$j = 0[+T[n-1]](=0[+1])$ と表せる。
2. $i = k+1$ で成立すると仮定する。このとき、$0 \le j < T[k-1]$ なる j について、
   (a) $0 \le j < T[k]$ のとき、帰納法の仮定より、$j = 0[+T[k+1]]\cdots[+T[n-1]]$。
   これは、$j = 0 + 0[+T[k+1]]\cdots[+T[n-1]]$ と書き直すと、
   $j = 0[+T[k]][+T[k+1]]\cdots[+T[n-1]]$ に含まれる。
   (b) $T[k] \le j < T[k-1]$ のとき、定義より、$T[k-1] \le 2T[k]$。ゆえに、
   $j - T[k] < T[k-1] - T[k] \le 2T[k] - T[k] = T[k]$。よって、帰納法の仮定より、
   $j - T[k] = 0[+T[k+1]]\cdots[+T[n-1]]$ と表せる。これを j について解くと、
   $j = 0 + T[k][+T[k+1]]\cdots[+T[n-1]]$ となり、
   $j = 0[+T[k]][+T[k+1]]\cdots[+T[n-1]]$ に含まれる。

   ゆえに $i=k$ でも成立する。

故に、$0 \le i \le n-1$ なる任意の i に対して、成り立つ。□

Theorem 1 任意のプロセッサ数 N、スタートラウンド r、プロセッサ s に対して、プロセッサ i から他の全てのプロセッサへの情報散布は、故障したプロセッサがない場合、$n = \lceil \log_2 N \rceil$ ラウンドで完了する。

Proof. 対称性より、$s=0$ としても一般性は損なわれない。この時、スタートラウンド r によらず、n 回のラウンドで情報が伝達されるプロセッサは、

$$j = (0[+T[0]][+T[1]]\cdots[+T[n-1]]) \bmod N$$

で表される。**lemma 1** より、$0[+T[0]][+T[1]]\cdots[+T[n-1]]$ は、0 以上 $N(=T[-1])$ 未満の任意の数を表せるから、

$$j = (0[+T[0]][+T[1]]\cdots[+T[n-1]]) \bmod N$$

で表される j は、任意のプロセッサを表すことになる。

ゆえに、任意のプロセッサ s から他の全てのプロセッサへの情報散布は、$n = \lceil \log_2 N \rceil$ ラウンドで完了する。□

3. t 方向 / ラウンド送受信可能な場合の散布方式

第 2 節で述べた方式は、ネットワーク中で 1 つのプロセッサが 1 ラウンドごとに 1 つのプロセッサにしかメッセージを送れない場合であった。これらを拡張して 1 ラウンド毎に 1 つのプロセッサから複数のプロセッサへ同じメッセージを送れる方式について示す。

ネットワークは第 2 節と同様に、0 から $N-1$ まで番号付けされた N 個のプロセッサから成り、リンクは各々示す方式に応じて適宜張られるものとする。以下に 3 種類の方式を示すが、各々は第 2 節の拡張である。

3.1. t 方向 / ラウンド送受信可能な場合の散布方式 その 1

procedure t-disseminate1(N)
repeat
  for $round$:= 0 **to** $\lceil \log_{(t+1)} N \rceil - 1$ **do**
    各々のプロセッサ i は
    プロセッサ $((i + 1 \times (t+1)^{round}) \bmod N)$
    $((i + 2 \times (t+1)^{round}) \bmod N)$
    $((i + 3 \times (t+1)^{round}) \bmod N)$
    $\vdots$
    $((i + t \times (t+1)^{round}) \bmod N)$
    にメッセージを送る。
repeatforever

各々のプロセッサ i $(0 \le i \le N-1)$ のメッセージ送り先は プロセッサ $((i + j \times (t+1)^{round}) \bmod N)$ $(1 \le j \le t)$ であり、$round$ の値は 0 から $\lceil \log_{(t+1)} N \rceil - 1$ まで連続的に変化する。

この方式は 1-disseminate1 の拡張であり、同じように任意のソース、任意のラウンドからスタートし他の全てのプロセッサに $\lceil \log_{(t+1)} N \rceil$ ラウンドでブロードキャストを完成することができる。

Example 4. 11 個のプロセッサが 結合され 1 度に 2 つのプロセッサにメッセージを送ることのできるシステムを考える ($t=2$)。どのプロセッサ間にリンクが存在するかは Table 4 の散布表に基づいて決定される。

ソースをプロセッサ 10、スタートラウンドを 2 とする。まずラウンド 2 でプロセッサ 10 からプロセッサ {8,2} へメッセージが送られ、次にラウンド 0 に戻り、プロセッサ {2,8,10} からプロセッサ {{3,4},{9,10},{0,1}} へ送られる。最後にラウンド 1 で {0,1,2,···} から {{3,6},{4,7},{5,8},···} というように 3 回のラウンドでブロードキャストが完了する。□

3.2. t 方向 / ラウンド送受信可能な場合の散布方式 その 2

procedure t-disseminate2(N)
repeat
  for $round$:= 0 **to** $r = \lceil \log_{(t+1)} \rceil - 1$ **do**
    各々のプロセッサ i は
    プロセッサ $(i + 1 \times (t+1)^{r-round}) \bmod N$
    $(i + 2 \times (t+1)^{r-round}) \bmod N$
    $(i + 3 \times (t+1)^{r-round}) \bmod N$
    $\vdots$
    $(i + t \times (t+1)^{r-round}) \bmod N$
    にメッセージを送る。
repeatforever

各々のプロセッサ i $(0 \le i \le N-1)$ のメッセージ送り先は プロセッサ $((i + j \times (t+1)^{r-round}) \bmod N)$ $(1 \le j \le t)$ であり、$round$ の値は 0 から $\lceil \log_{(t+1)} N \rceil - 1$ まで連続的に変化する。この方式は 1-disseminate2 の拡張であり、同様に任意のソース、ラウンドからスタートし全てのプロセッサに $\lceil \log_{(t+1)} N \rceil$ ラウンドでブロードキャストを完成することができる。

Example 5. 11 個のプロセッサが 結合され 1 度に 2 つのプロセッサにメッセージを送れるシステムを考える ($t=2$)。どのプロセッサ間にリンクが存在するかは Table 5 の散布表に基づいて決定される。

ソースをプロセッサ 8、スタートラウンドを 2 とする。まずラウンド 2 でプロセッサ 8 からプロセッサ {9,10} へメッセージが送られ、次にラウンド 0 でプロセッサ {8,9,10} からプロセッサ {{6,0}{7,1}{8,2}} へ送られる。最後にラウンド 1 でプロセッサ {0,1,2,6,7,8,9,10} から {{3,6}{4,7}{5,8} ···} というようにして 3 回のラウンドでブロードキャストが完了する。このように任意のソース、ラウンドからスタートし、$\lceil \log_{(t+1)} N \rceil$ 回のラウンドでブロードキャストを完了できる。□

round	processors 0	1	2	3	4	5	6	7	8	9	10
0	1,2	2,3	3,4	4,5	5,6	6,7	7,8	8,9	9,10	10,0	0,1
1	3,6	4,7	5,8	6,9	7,10	8,0	9,1	10,2	0,3	1,4	2,5
2	9,3	10,4	0,5	1,6	2,7	3,8	4,9	5,10	6,0	7,1	8,2

Table 4: $N = 11, t = 2$ の散布表 (散布方式 その 1)

round	processors 0	1	2	3	4	5	6	7	8	9	10
0	9,3	10,4	0,5	1,6	2,7	3,8	4,9	5,10	6,0	7,1	8,2
1	3,6	4,7	5,8	6,9	7,10	8,0	9,1	10,2	0,3	1,4	2,5
2	1,2	2,3	3,4	4,5	5,6	6,7	7,8	8,9	9,10	10,0	0,1

Table 5: $N = 11, t = 2$ の散布表 (散布方式 その 2)

T[i,j]

i \ j	1	2
0	4	8
1	2	4
2	1	2

round	processors 0	1	2	3	4	5	6	7	8	9	10	11
0	4,8	5,9	6,10	7,11	8,0	9,1	10,2	11,3	0,4	1,5	2,6	3,7
1	2,4	3,5	4,6	5,7	6,8	7,9	8,10	9,11	10,0	11,1	0,2	1,3
2	1,2	2,3	3,4	4,5	5,6	6,7	7,8	8,9	9,10	10,11	11,0	0,1

Table 6: N=12 の場合の散布表 (散布方式 その 3)

3.3 t 方向 / ラウンド送受信可能な場合の散布方式 その 3

prosedure t-disseminate3(N)
for i := 0 **to** $\lceil \log_{(t+1)} N \rceil - 1$ **do**
  $N := \lceil N/(t+1) \rceil$
  for j := 1 **to** t **do**
    $T[i,j] := j \times N$
repeat
  for *round* := 0 **to** $\lceil \log_{(t+1)} N \rceil - 1$ **do**
    各々のプロセッサ i は
    プロセッサ $((i + T[round, 1]) \bmod N)$
         $((i + T[round, 2]) \bmod N)$
         $((i + T[round, 3]) \bmod N)$
         $\vdots$
         $((i + T[round, t]) \bmod N)$
    にメッセージを送る。
repeatforever

この方式は 1-disseminate3 の拡張で、複数のプロセッサにメッセージを送れるものである。プロセッサ i $(0 \leq i \leq N-1)$ のメッセージ送り先はプロセッサ $((i+T[round, j]) \bmod N)$ $(1 \leq j \leq t)$ であり、*round* の値は 0 から $\lceil \log_{(t+1)} N \rceil - 1$ まで連続的に変化する。配列 T$[i, j]$ の値は N によって与えられる。

Example 6. 12 個のプロセッサが結合され 1 度に 2 つのプロセッサにメッセージを送ることのできるシステムを考える。どのプロセッサ間にリンクが存在するかは Table 6 の散布表に基づいて決定される。

ソースをプロセッサ 3、スタートラウンドを 2 とする。まずラウンド 2 でプロセッサ 3 からプロセッサ {4,5} へメッセージが送られ、次のラウンド 0 でプロセッサ {3,4,5} からプロセッサ {{7,11}{8,0}{9,1}} へ送られる。最後にラウンド 1 でプロセッサ {0,1,3,4, $\cdots$ } から {{2,4}{3,5}{5,7} $\cdots$ } にメッセージが送られブロードキャストが完了する。 □

4. 1 方向 / ラウンドの散布方式の耐故障性

ネットワーク中に故障したプロセッサが存在する時、第 2 節で示した 1 ラウンドごとに 1 方向送受信可能な情報散布方式を用いると、ブロードキャストに必要なラウンド数はどれくらいになるかを論じる。

4.1 1-disseminate1 の耐故障性

N 個のプロセッサを持つネットワーク中に故障したプロセッサが 1 つだけ存在する場合、1-disseminate1 を用いると、ブロードキャストに必要なラウンド数は $\lceil \log_2 N \rceil + 2$ ラウンドとなる。この証明は、[HF] で述べられている。

4.2 1-disseminate2 の耐故障性

この方式は 1-disseminate1 の拡張で対称性があるので、故障したプロセッサが 1 個の場合、[HF] の結果と同様にブロードキャストに必要なラウンド数は $\lceil \log_2 N \rceil + 2$ となることが推測される。この事実が正しいことは、第 5.2 節でより一般的な場合について証明する。

4.3 1-disseminate3 の耐故障性

N 個のプロセッサを持つネットワーク中に故障したプロセッサが 1 つだけ存在する場合、1-disseminate3 を用いると、ブロードキャストに必要なラウンド数は $\lceil \log_2 N \rceil + 1$ となる。証明を以下に示す。

まず、ラウンド 0 からラウンド i までの間に、プロセッサ 0 から情報が伝達されるプロセッサのうち、冗長なもの以外のものの集合 U_i を定義する。

Definition 1

$$\begin{cases} U_{-1} = & \{0\} \\ U_i = & U_{i-1} \cup \{u \mid u - T[i] \in U_{i-1}, u < N, \ and \\ & \qquad all\ w \in U_{i-1}, w > u - T[i] \Rightarrow u < w\}, \\ & (0 \leq i \leq n-1) \end{cases}$$ □

任意の i に対して、U_i の元を昇順に並べた時、最初の二つは $0, T[i]$ である。また、隣合う元の差は、1 以上 $T[i]$ 以下であり、U_i の最大の元と N との差も、1 以上 $T[i]$ 以下である。

U_i に含まれるプロセッサの間での、標準的な情報伝達関係を、以下のように定義する。

Definition 2 $\{0, 1, \cdots, N-1\}$ 上の二項関係 $\prec$ を、以下のように定義する。

- $u \in U_i, u \notin U_{i-1}$ のとき、$u - T[i] \prec u$

またこの二項関係 $\prec$ の半順序関係への拡張 $\overset{+}{\prec}$ を、以下のように定義する。

1. $u \prec v$ ならば $u \overset{+}{\prec} v$
2. $u \overset{+}{\prec} v$ 、$v \overset{+}{\prec} w$ ならば、$u \overset{+}{\prec} w$ □

Proposition 1 $u \overset{+}{\prec} v$ ならば、$u < v$。また、$u \overset{+}{\prec} v$ かつ $v \in U_i, (0 \le i \le N-1)$ ならば、$u \in U_{i-1}$。

Proof. Definition 2 より明らか。 □

Definition 3 U_i の元 u に対して、$u \prec v$ なる v が U_i 中に存在する時、u は、(U_i において) 中継点であるという。そのような v が U_i 中に存在しない時、u は (U_i において) 終端点であるという。 □

Lemma 2 $U_i(0 \le i \le N-1)$ の二つの元を $u, v(u < v)$ とする時、$v - u < T[i]$ ならば、u は終端点である。

Proof. u が中継点であると仮定すると、$u \prec w$ なる w が U_i 中に存在する。また、$u < v$ より $v \neq 0$ であるから、$v \in U_i$ より、$x \prec v$ なる x が U_i 中に存在する。定義より明らかに $w - u = T[j] \ge T[i], v - x = T[k] \ge T[i](j, k \le i)$ であるから、$v - u < T[i]$ より、$x < u < v < w$。この時、$j \le k$ と仮定すると、$x, u \in U_{k-1}, v \notin U_{k-1}, v \in U_k$ となり、U_k の定義に反する。また、$j > k$ と仮定すると、$u, v \in U_{j-1}, w \notin U_{j-1}, w \in U_j$ となり、U_j の定義に反する。よって、u は中継点ではない。 □

Theorem 2 任意のプロセッサ数 N、スタートラウンド r、故障していないプロセッサ s に対して、プロセッサ s から他の全てのプロセッサへの情報散布は、故障したプロセッサが唯一の場合、その故障したプロセッサの位置に関わらず、$n+1(n = \lceil \log_2 N \rceil)$ 回のラウンドで完了する。

Proof. 対称性より、$s = 0$ としても一般性は損なわれない。故障したプロセッサを $f(\neq 0)$ とし、その位置によって場合分けする。

1. $1 \le f \le T[r] - 1$ のとき、ラウンド $n-1$ が終了した時、$0 \cdots 2T[r]-1$ で確実に情報を受けとっているプロセッサの集合は、

$$S_0 = \{g \bmod N | g = 0 \text{ or } T[r] \le g \le 2T[r] - 1\}$$

情報を受けとっていない可能性のあるプロセッサの集合は、

$$F_0 = \{g \bmod N | 1 \le g \le T[r] - 1\}$$

となる。さらに、ラウンド 0 からラウンド $r-1$ までの情報散布が終了した時、確実に情報を受けとっているプロセッサの集合は、

$$S = \bigcup_{u \in U_{r-1}} u + S_0$$

情報を受けとっていない可能性のあるプロセッサの集合は、

$$F = \bigcup_{u \in U_{r-1}} u + F_0$$

となる。但し、$u + \{g_1, g_2, \cdots, g_k\}$ は、$\{(u+g_1) \bmod N, (u+g_2) \bmod N, \cdots, (u+g_k) \bmod N\}$ を表す。この時、F に含まれる個々のプロセッサの $T[r]$ 後にあるプロセッサの集合は、

$$\begin{aligned} F' &= \bigcup_{u \in U_{r-1}} (u + T[r]) + F_0 \\ &= \bigcup_{u \in U_{r-1}} u + \{(g + T[r] \bmod N | 1 \le g \le T[r] - 1\} \\ &= \bigcup_{u \in U_{r-1}} u + \{g' \bmod N | T[r] + 1 \le g' \le 2T[r] - 1\} \end{aligned}$$

となり、これは S に含まれる。よって、情報が伝達されていない (S に含まれていない)、いかなるプロセッサも、その $T[r]$ 前のプロセッサは、F に含まれていない。ゆえに、$n+1$ 回目のラウンド (2 度目のラウンド r) で、情報散布が完了する。

2. $T[r] \le f \le T[r-1] - 1$ のとき、ラウンド $n-1$ が終了した時、$0 \cdots T[r-1]-1$ で確実に情報を受けとっているプロセッサの集合は、

$$S_0 = \{g \bmod N | 0 \le g \le T[r] - 1\}$$

情報を受けとっていない可能性のあるプロセッサの集合は、

$$F_0 = \{g \bmod N | T[r] \le g \le T[r-1] - 1\}$$

となる。さらに、ラウンド 0 からラウンド $r-1$ までの情報散布が終了した時、確実に情報を受けとっているプロセッサの集合は、

$$S = \bigcup_{u \in U_{r-1}} u + S_0$$

情報を受けとっていない可能性のあるプロセッサの集合は、

$$F = \bigcup_{u \in U_{r-1}} u + F_0$$

となる。この時、F に含まれる個々のプロセッサの $T[r]$ 前にあるプロセッサの集合は、

$$\begin{aligned} F' &= \bigcup_{u \in U_{r-1}} (u - T[r]) + F_0 \\ &= \bigcup_{u \in U_{r-1}} u + \{(g - T[r]) \bmod N | \\ &\quad T[r] \le g \le T[r-1] - 1\} \\ &= \bigcup_{u \in U_{r-1}} u + \{g' \bmod N | \\ &\quad 0 \le g' \le T[r-1] - T[r] - 1 \le T[r] - 1\} \end{aligned}$$

となり、これは S に含まれる。よって、$n+1$ 回目のラウンド (2 度目のラウンド r) で、情報散布が完了する。

3. $T[r-1] \le f \le N-1 (r > 0)$ の時、ラウンド $n-1$ が終了した時点では、$\{g | 0 \le g \le 2T[r] - 1\}$ に含まれ

る全てのプロセッサは、確実に情報が伝達されている。U_{r-1} の元を昇順に並べたものを $u_0, u_1, \cdots, u_{n_r-1}$ とする時、$u_i \leq f \leq u_{i+1}-1$ なる $i(>1)$ が存在する。(但し、$u_{n_r-1} \leq f(\leq N-1)$ の時は、$i=n_r-1$ とする。) U_{r-1} の部分集合 $U' = \{u | u \in U_{r-1}, (0 \leq) u_i - T[r-1] < u \leq u_i\}$ を考えると、明らかに $u_i \in U'$。また、u_i 以外の U' の元は全て、U_{r-1} における終端点となる。(Lemma 2 より明らか。) 故に、U_{r-1} における任意の中継点 u に対して、$u \neq u_i$ ならば、$f < u$ または $u + T[r-1] \leq f$。

(a) u_i が終端点の時、任意のプロセッサを v とすると、$v = u_j + t, (u_j \in U_{r-1}, 0 \leq t < T[r-1])$ と表せる。この v へは、ラウンド r から $n-1$ までの間にプロセッサ 0 から t までの情報伝達が行なわれ、ラウンド 0 から $r-1$ までの間に $t \to v_1+t \to v_2+t \to \cdots \to v_k+t = u_j+t = v, (v_i \in U_{r-1}, 0 \prec v_1 \prec \cdots \prec v_k = u_j)$ と情報伝達が行なわれるので、多くとも合計 n 回のラウンドで情報が伝達される。よって、任意のプロセッサへの情報散布は、n 回のラウンドで完了する。

(b) u_i が中継点の時、ラウンド 0 から $r-1$ までの情報散布が終了した時に、情報が伝達されていない可能性のあるプロセッサの集合は、

$$F = \bigcup_{u \in U_{r-1}, u_i \overset{+}{\prec} u} \{(u + (f - u_i)) \bmod N\}$$

となる。

i. $u_i \leq f \leq u_i + T[r] - 2$ のとき、F に含まれているプロセッサの $T[r]$ 後にあるプロセッサの集合は、

$$\bigcup_{u \in U_{r-1}, u_i \overset{+}{\prec} u} \{(u + (f - u_i + T[r])) \bmod N\}$$

となる。この時、仮定より $T[r] \leq f - u_i + T[r] \leq 2T[r] - 2 \leq T[r-1] - 1$ であるから、この集合に含まれるプロセッサには、プロセッサ $f - u_i + T[r]$ を経由して、情報が伝達されているはずである。ゆえに、この時点で情報が伝達されていないプロセッサの $T[r]$ 前のプロセッサは、F に含まれていない。よって、$n+1$ 回目のラウンド (2 度目のラウンド r) で、情報散布が完了する。

ii. $f = u_i + T[r] - 1$ のとき、ある $u \in U_{r-1}$ について、$u + 2T[r] - 1 = f$ と仮定すると、$u_i - u = T[r] < T[r-1]$ となるから、Lemma 2 より u は終端点となる。ゆえに、U_{r-1} における任意の中継点 u に対して、$u \neq u_i$ ならば、$f < u$ または $u + 2T[r] \leq f$。一方、$f - u_i + T[r] = 2T[r] - 1$ であるから、3(b)i と同様に、$n+1$ 回のラウンドで、情報散布が完了する。

iii. $u_i + T[r] \leq f \leq u_{i+1} - 1 \leq u_i + T[r-1] - 1$ のとき、($i = n_r - 1$ の場合は、$u_i + T[r] \leq f \leq N-1 \leq u_i + T[r-1] - 1$ のとき、) F に含まれているプロセッサの $T[r]$ 前にあるプロセッサの集合は、

$$\bigcup_{u \in U_{r-1}, u \geq u_i} \{(u + (f - u_i - T[r])) \bmod N\}$$

となる。この時、仮定より、$0 \leq f - u_i - T[r] \leq T[r-1] - T[r] - 1 \leq T[r] - 1$ であるから、この集合に含まれるプロセッサには、プロセッサ $f - u_i - T[r]$ を経由して、情報が伝達されているはずである。よって、$n+1$ 回目のラウンドで (2 度目のラウンド r) で, 情報伝達が完了する。

以上により、定理は証明された。 □

こうして、Theorem 2 より 1-disseminate3 の耐故障性として、故障したプロセッサが唯一の場合 $\lceil \log_2 N \rceil + 1$ ラウンドでブロードキャストを完了することがいえる。

5. t 方向 / ラウンドの散布方式の耐故障性

ネットワーク中に故障したプロセッサが存在する時、第 3 節で示した 1 ラウンドごとに 1 方向送受信可能な情報散布方式を用いると、ブロードキャストに必要なラウンド数はどれくらいになるか論じる。

5.1 t-disseminate1 の耐故障性

t-disseminate1 の耐故障性については現在のところ未解決な問題が多い。t-disseminate1 と t-disseminate2 には対称的な関係があることが推測できる。よって、次の第 5.2 節で示す定理がこの方式にも当てはまるのではないか、ということが予想される。

5.2 t-disseminate2 の耐故障性

この節では、N 個のプロセッサを持つネットワーク中に故障したプロセッサの数が t 以下の場合、t-disseminate2 を用いると、$\lceil \log_{t+1} N \rceil + 2$ ラウンドでブロードキャストを完了することができることを示す。

$r = \lceil \log_{t+1} N \rceil - 1$ とおく。各プロセッサを長さ $\lceil \log_{t+1} N \rceil$ の $t+1$ 進数で表す。但し、$t+1$ 進数 $a_0 \cdots a_r$ が N 以上の時、プロセッサ $a_0 \cdots a_r$ はプロセッサ $[a_0 \cdot a_r]_N$ を意味する。ここで、$[a_0 \cdots a_r]_N$ は $(a_0 \cdots a_r) \bmod N$ を表す。$b_0 \cdots b_s$ が $t+1$ 進数で $s \leq r$ の時、$[b_0 \cdots b_s]$ はプロセッサの集合 $\{b_0 \cdots b_s a_{s+1} \cdots a_r | \text{for each } i (s+1 \leq i \leq r), 0 \leq a_i \leq t\}$ を表す。このようなプロセッサの集合を大きさ $(t+1)^{r-s}$ のブロックという。

Theorem 3 故障したプロセッサの数が t 以下の N 個のプロセッサを持つネットワーク上で t-disseminate2 により情報散布を行なった場合、スタートラウンドが 0 でなければ任意の故障のないソースから全てのプロセッサに $\lceil \log_{t+1} N \rceil + 2$ ラウンド以内で情報散布を完了することができる。スタートラウンド 0 の時は、$\lceil \log_{t+1} N \rceil + 3$ ラウンド以内で情報散布を完了する。

Proof. 一般性を失うことなくプロセッサ 0 をソースとしても良い。以下の各場合について各々証明する。

1. スタートラウンドが r の時。
   故障したプロセッサの数は高々 t であるので、$r+1$ ラウンドの実行が終った時 (ラウンド $r-1$ の終り) では任意の連続した $t+1$ 個のプロセッサの中に少なくとも 1 つ故障していないプロセッサですでに情報を受けとっているものがある。従って次のラウンドで情報散布は完了する。

2. スタートラウンドが 0 でも r でもない時。
   スタートラウンドをラウンド $i(\neq 0, \neq r)$ とする。大きさ $(t+1)^{r-i+1}$ の最初のブロックに入っていない大きさ $(t+1)^{r-i}$ の各ブロックについて次のことが成立する:

(*) $[b_0 \cdots b_i]$ を上で述べた条件を満足する大きさ $(t+1)^{r-i}$ のブロックとする。$[b_0 \cdots b_i - t], \cdots, [b_0 \cdots b_i - 1], [b_0 \cdots b_i]$ の中の少なくとも 1 つのブロックは故障したプロセッサを含まず、ラウンド $i-1$ の終了時で情報散布は完了している。

上のことから次のラウンド (2 度めのラウンド i) でブロック $[b_i \cdots b_i]$ の情報散布は完了している。従って大きさ $(t+1)^{r-i+1}$ の最初のブロック以外は $\lceil \log_{t+1} N \rceil + 1$ ラウンド以内で情報散布は完了している。

次に大きさ $(t+1)^{r-i+1}$ のブロックに含まれている、大きさ $(t+1)^{r-i}$ のブロックについて考える。故障したプロセッサの数は高々 t であるので、これらのブロックの中にブロック内の全てのプロセッサは故障しておらず $r+1$ ラウンド以内で情報散布が完了しているブロックが少なくとも 1 つ存在する。$[0^i p]$ をこのようなブロックとする。$p \leq q \leq t$ であるようなブロック $[0^i q]$ について、このブロック内の全てのプロセッサは 2 回目のラウンド i でブロック $[0^i p]$ のプロセッサから情報を受けとる。$0 \leq q < p$ であるようなブロック $[0^i q]$ についてはこの証明の前の部分を適用する。$i = r-1$ の時は、この証明の場合 1 からラウンド r をスタートラウンドと考えることにより $\lceil \log_{t+1} N \rceil + 2$ 以内で全体の情報散布が完了していることがいえる。$i \geq \lceil \log_{t+1} N \rceil - 3$ と仮定する。最初のラウンド i でソースから情報を受けとり、ブロック $[0^i q]$ のどのプロセッサよりもプロセッサ番号が大きいプロセッサが存在する。このプロセッサをソースと考える。ラウンド $i+1$ をスタートラウンドと考えれば、この証明の場合 2 の前半が適用できる。すなわち $\lceil \log_{t+1} N \rceil + 2$ で情報散布が完了する。

3. スタートラウンドが 0 の場合。

   スタートラウンド 0 の時、上の 2 つの場合から $\lceil \log_{t+1} N \rceil + 3$ で情報散布が完了するのは明らかである。 □

スタートラウンドが 0 の時、$\lceil \log_{t+1} N \rceil + 3$ ラウンドが情報散布を完了させるのに必要なラウンド数であるかどうかは未解決である。スタートラウンドが 0 の時、どのような場合に $\lceil \log_{t+1} N \rceil + 2$ ラウンドで十分であるかどうかを以下で検討する。

故障したプロセッサの数は高々 t であるので、ブロック [0] の中にない任意のプロセッサ u について、$u, u-(t+1)^{r-1}, \cdots, u-t(t+1)^{r-1}$ の中に故障していなくて、かつ $\lceil \log_{t+1} N \rceil + 1$ ラウンド以内で情報を受理しているプロセッサが少なくとも 1 つ存在する。従って、2 回目のラウンド 1 で u は情報を受理している。すなわち、u は $\lceil \log_{t+1} N \rceil + 2$ 以内で情報を受理する。

1. ブロック 0 の全てのプロセッサが故障していない場合。

   明らかにブロック [0] の全てのプロセッサは $\lceil \log_{t+1} N \rceil$ ラウンド以内で情報を受理する。

2. ブロック [0] の中に故障したプロセッサが存在する場合。

   故障したプロセッサの総数は t 以下であるから、ブロック [0] の中に故障したプロセッサを含まなくて大きさが $(t+1)^{r-1}$ のブロックが少なくとも 1 つ存在する。このようなブロックの最初のものを $[0p]$ とする。$\lceil \log_{t+1} N \rceil$ ラウンドで $[0p]$ の全てのプロセッサは情報を受理している。従って、次のラウンド 1 でブロック $[0p], [0(p+1)], \cdots, [0t]$ の中の全てのプロセッサは情報を受理する。すなわち、これらのブロック内のプロセッサはいずれも $\lceil \log_{t+1} N \rceil + 2$ ラウンド以内で情報を受理する。しかしこの場合ブロック $[00], \cdots, [0(p-1)]$ の全てのプロセッサがこのラウンド数内で情報を受理するかどうかは未解決である。

$t = 1$ または 2 の時は、$\lceil \log_{t+1} N \rceil + 2$ ラウンドでブロードキャストを完了させることを示すことができる。

Theorem 4 $t = 1$ または 2 とする。N 個のプロセッサを持つネットワークが高々 t 個の故障したプロセッサを含む時、*t-disssseminate2* を用いると、任意の故障していないソースから全てのプロセッサへの情報散布は $\lceil \log_{t+1} N \rceil + 2$ ラウンド以内で完了する。

Proof. 我々はすでにスタートラウンドが 0 以外の時は、上の主張が正しいことは証明済みである。ここではスタートラウンドを 0 と仮定し、一般性を失うことなくソースはプロセッサ 0 とする。

$t = 1$ の場合について証明する。

この場合、ブロック [00] の全てのプロセッサが $\lceil \log_2 N \rceil + 2$ ラウンド以内で情報を受けとることを示せば十分である。ブロック [00] に故障しているプロセッサを含まない場合は、上の主張は明らかに成立する。ブロック [00] に故障したプロセッサを含むと仮定する。故障したプロセッサの数は高々 1 であるので、ブロック [00] に含まれないどのプロセッサも故障していない。

1. $[0^r - 10^{r-1}]_N < 010^{r-1}$ の時。

   ブロック $[00] \cap [11]$ の全てのプロセッサは 2 回目のラウンド 0 で [01] のプロセッサから情報を受けとり、$[00] - [11]$ の全てのプロセッサは 2 回目のラウンド 1 で [01] のプロセッサから情報を受けとる。従ってこの場合は $\lceil \log_2 N \rceil + 2$ ラウンドで情報散布が完了する。

2. $[0^r - 10^{r-1}]_N \geq 010^{r-1}$ の時。

   ブロック [00] の全てのプロセッサは 2 回目のラウンド 1 で $[1] - [00]$ のプロセッサから情報を受けとる。従ってこの場合も $\lceil \log_2 N \rceil + 2$ ラウンドで情報散布が完了する。

次に $t = 2$ の場合について証明する。

1. ブロック [00] とブロック [01] のいずれも故障したプロセッサを含む場合。

   この場合、$[00] \cup [01]$ 以外のプロセッサは全て故障していない。

   (a) $[0^r - 10^{r-1}]_N \leq 010^{r-2}$ の時。

   ブロック [00] の全てのプロセッサは 2 回目のラウンド 1 で $([10] \cup [02]) - [00]$ のプロセッサから情報を受けとる。従ってこの場合は $\lceil \log_3 N \rceil + 2$ ラウンドで情報散布が完了する。

   (b) $[0^r - 10^{r-1}]_N > 010^{r-2}$ の時。

   ブロック [00] の全てのプロセッサは 2 回目のラウンド 1 で $([1] \cup [2]) - ([00] \cup [01])$ のプロセッサから情報を受理する。従ってこの場合も $\lceil \log_3 N \rceil + 2$ ラウンドで情報散布が完了する。

2. ブロック [00] とブロック [02] のいずれも故障したプロセッサを含む場合。

(a) $[0^r - 10^{r-1}]_N \leq 010^{r-2}$ の時。
ブロック [00] の全てのプロセッサは、2回目のラウンド1で $([10] - [00]) \cup [01]$ のプロセッサから情報を受けとる。

(b) $[0^r - 10^{r-1}]_N > 010^{r-2}$ の時。
ブロック [00] の全てのプロセッサは、2回目のラウンド1で $([1] \cup [2]) - [0]$ のプロセッサから情報を受けとる。

3. ブロック [00] は故障したプロセッサを含むが、ブロック [01] もブロック [02] も故障したプロセッサを含まない場合。

(a) $[0^r - 10^{r-1}]_N \leq 010^{r-2}$ の時。
ブロック [00] の全てのプロセッサは2回目のラウンド1で $[01] \cup [02]$ のプロセッサから情報を受けとる。

(b) $010^{r-2} < [0^r - 10^{r-1}]_N \leq 020^{r-2}$ の時。
ブロック [00] の全てのプロセッサは2回目のラウンド0で $[01] \cup [02]$ のプロセッサから情報を受けとる。

(c) $020^{r-2} < [0^r - 10^{r-1}]_N \leq 10^{r-1}$ の時。
$[1] \cap [00]$ の全てのプロセッサは、2回目のラウンド0で $[01] \cup [02]$ のプロセッサから情報を受けとる。$[00] - [1]$ の各々のプロセッサは、2回目のラウンド1で [11] のあるプロセッサと [12] のあるプロセッサから情報を受けとるはずであるが、そのいずれかは故障している可能性がある。しかしこの場合は、これらのプロセッサはいずれもブロック [00] に存在しないのでいずれかは故障していない。したがっていずれの場合も $\lceil \log_3 N \rceil + 2$ ラウンド以内で情報を受けとる。

(d) $10^{r-1} \leq [0^r - 10^{r-1}]_N$ で、$010^{r-2} \leq [0^r - 20^{r-1}]_N \leq 020^{r-2}$ の時。
ブロック [00] の全てのプロセッサは2回目のラウンド0で $[01] \cup [02]$ のプロセッサから情報を受けとる。

(e) $10^{r-1} \leq [0^r - 10^{r-1}]_N$ で、$020^{r-2} < [0^r - 20^{r-1}]_N$ の時。
ブロック [00] の各々のプロセッサは2回目のラウンド1で $[2] - [00]$ の2つのプロセッサから情報を受けとるはずである。この2つのプロセッサの少なくとも1つはすでに情報を受けとっており、故障していない。従って、ブロック [00] のいずれのプロセッサも $\lceil \log_3 N \rceil + 2$ ラウンド以内で情報を受けとる。 □

5.3 t-disseminate3 の耐故障性

第4節の結果から1方向 / ラウンド情報散布可能な時、故障が1つの場合、1-disseminate1 または 1-disseminate2 よりも 1-disseminate3 の方が耐故障性に関しては優れている。しかし、t 方向 / ラウンド情報散布可能な場合では この事実が正しいかどうかは今のところ明らかでない。

6. まとめ

1-disseminate1 で故障の数が1の場合、ブロードキャストに必要なラウンド数は $\lceil \log_2 N \rceil + 2 (= n+2)$ であった。そこで、一般に故障の数を k とした時、ブロードキャストに必要なラウンド数は $n + k + 1$ と表されるのでは、と推測されるが、計算機でシミュレーション実験を行なったところ、$n + k + 5$ ラウンドまでかかる例が見つかった。適当な定数 x で必要なラウンド数を $n + k + x$ の形で表すことができるかどうかは今後の課題である。

1-disseminate1 と 1-dissemanate2、また t-disseminate1 と t-disseminate2 は対称的な方式であるのでこれらの方式について類似の関係が成立すると考えられるが、これらは未解決であり今後の課題である。

謝辞

計算機によるシミュレーション実験に協力していただいた群馬大学工学部情報工学科の大澤新吾氏に感謝する。

参考文献

[B] D. Bienstock, "Broadcasting with Random Graphs", Discrete Applied Mathematics 20, pp.1-7 (1988).

[F] P. Fraigniaud, "Performance analysis of broadcasting", Proceedings of The 1st European Workship on Hypercubes and Destributed Computers, Rennes, France, pp.331-327 (1989).

[FP] U. Feige, D. Peleg and P. Raghavan, "Randamized broadcast in networks", Proceedings of SIGAL International Symposium on Algorithms, Tokyo, Japan, Lecture Notes in Computer Science, Vol. 450, pp.128-137 (1990).

[HF] Y. Han and R. Finkel, "An optimal scheme for disseminating information", Proceedings of 1988 International Conference on Parallel Processing, Chicago, Illinois, pp.198-203 (1988).

[HHL] S. M. Hedetniemi, S. T. Hedetniemi and A. L. Liestman, "A Survey of Gossiping and Broadcasting in Communication Neetworks", NETWORKS, Vol. 18, pp.319-349 (1988).

[IKM] Y. Igarashi, K. Kanai and K. Miura, "Information Disseminating Schemes for Fault Tolerance in Hypercubes", 電子情報通信学会技術研究会報告 (コンピュテーション), COMP 90-28, pp.81-86 (1990).

[JH] S. L. Johnsson and Ching-Tien Ho, "Optimal broadcasting and personalized communication in hypercubes", IEEE Trans. on Computers, Vol.38, pp.1249-1268 (1989).

[L] A. L. Liestman, "Fault-Tlerant Broadcast Graphs", NETWORKS, Vol. 15, pp.159-171 (1985).

[RS] P. Ramanathan and K. G. Shin, "Reliable broadcasting in hypercube multiprocessors", IEEE Trans. on Computers, Vol.37, pp.1654-1657 (1988).