

## Grundläggande kommunikationsoperationer

Grama et al.  
Introduction to Parallel Computing  
Kapitel 4  
Robert Granat

## Översikt

### Kommunikationsoperationer

- 1 Enkla meddelanden
  - 1-till-1
- 1 Broadcast (utsändning)
  - 1-till-alla
  - alla-till-alla
- 1 Reduktion, prefix summering
- 1 Personlig kommunikation
  - 1-till-alla
  - alla-till-alla
- 1 Cirkulära shift

### Topologier

- 1 Ring
- 1 2D-nät
- 1 3D-nät
- 1 Hyperkub

### Routing-tekniker

- 1 Store-and-forward (SF)
- 1 Cut-through (CT)

## Kommunikationskostnader

- 1  $t_s$ : start-upp-tid
  - Preparera meddelande & header
  - Exekvera routing-algoritm
- 1  $t_h$ : per-hopp-tid
  - Tid för header från en processor till granne
- 1  $t_w$ : per-ord-tid
  - tid för överföring per ord i meddelandet
  - om bandbredd  $r$  ord/s så är  $t_w = 1/r$

## Routing-tekniker

Meddelande av storlek  $m$  skickas över  $l$  länkar

Store-and-forward (lagra-vidarebefordra)

- 1 Varje länk tar tiden  $t_h$  för headern och  $m t_w$  för resten
  - $\Rightarrow t_{comm} = t_s + l(t_h + m t_w)$
  - (förenklas ofta till  $t_{comm} = t_s + l m t_w$  då  $t_h$  litet jämfört med  $m$ )

Cut-through (direkt-igenom)

- 1 Tar totalt tiden  $l t_h$  för headern  
Hela meddelandet anländer  $m t_w$  efter headern anlänt
  - $\Rightarrow t_{comm} = t_s + l t_h + m t_w \approx t_s + m t_w$

Teknikerna likvärdiga om  $l = 1$  eller  $m$  litet

## Förutsättningar & antaganden

- 1 Logiska topologier: ring, 2-dimensionellt nät, hyperkub
- 1 Dubbelriktade länkar
- 1 2 proc. kan samtidigt skicka meddelande av storlek  $m$  till varandra på tiden  $t_s + m t_w$
- 1  $t_s$  = startupp-tid,  $t_w$  = per-ord-tid
- 1 1 Processor kan endast
  - sända på en länk i taget
  - ta emot på en länk i taget
  - ... men ...
- 1 1 processor kan både skicka och ta emot samtidigt (1 eller olika länkar), jmf MPI\_Sendrecv

## Tid för ett meddelande (1 proc till 1 annan)

Lagra-vidarebefordra (store-and-forward)

- 1 Storlek  $m$  över  $l$  länkar:  $t_s + m l t_w$ 
  - $l$  är maximalt:  $\lfloor p/2 \rfloor$  för ring
  - $l$  är maximalt:  $2 \lfloor p^{1/2}/2 \rfloor$  för 2-dim nät (kvadr., wrap-around)
  - $l$  är maximalt:  $\log p$  för en hyperkub
- 1 Övre gräns för ett meddelande
  - $t_s + m t_w \lfloor p/2 \rfloor$  för ring
  - $t_s + m t_w \lfloor p^{1/2}/2 \rfloor$  för 2-dim nät
  - $t_s + m t_w \log p$  för en hyperkub

Direkt-igenom (cut-through)

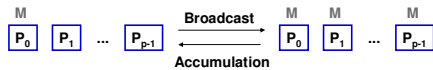
- 1  $t_s + m t_w + l t_h \approx t_s + m t_w$  där  $t_h$  är tid för ett hopp

## 1-till-alla utsändning (broadcast)

- 1 skickar samma data till alla
- Ex: matris-vektormult, gausselimination, kortaste vägen
- Dual operation: 1-nods-ackumulering (sum, max, etc.)

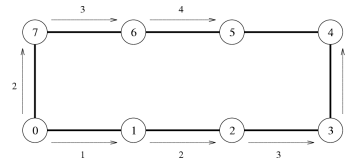
### Lagra-vidarebefordra (store-and-forward)

- Undvik onödiga meddelanden:
  - källprocessor skickar till de den når direkt (grannar)
  - de som fått meddelande i ett steg: skicka vidare i nästa steg



M: m ords meddelande

## 1-till-alla på ring



**Figure 3.2** One-to-all broadcast on an eight-processor ring with SF routing. Processor 0 is the source of the broadcast. Each message transfer step is shown by a numbered, dotted arrow from the source of the message to its destination. The number on an arrow indicates the time step during which the message is transferred. Copyright (r) 1994 Benjamin/Cummings Publishing Co.

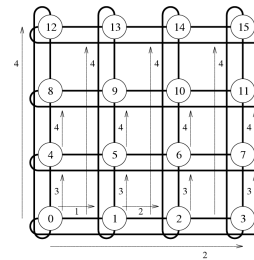
## 1-till-alla på ring

### Lagra-vidarebefordra

- Tar totalt  $\lceil p/2 \rceil$  steg
  - Varje steg tar  $t_s + m t_w$  tid
- Totalt krävs**  $(t_s + m t_w) \lceil p/2 \rceil$  tid

Hur kan detta nyttjas i en kolumn av ett 2-dimensionellt nät?

## 1-till-alla på 2-dim nät



**Figure 3.4** One-to-all broadcast on a 16-processor mesh with SF routing. Copyright (r) 1994 Benjamin/Cummings Publishing Co.

## 1-till-alla på kvadratisk 2-dim nät

### Lagra-vidarebefordra

- Broadcast i startnodens processorråd  
tid:  $(t_s + m t_w) \lceil p^{1/2}/2 \rceil$
- Varje processor i startnodens rad startar broadcast i sin processorkolumn (// i alla kolumner)  
tid:  $(t_s + m t_w) \lceil p^{1/2}/2 \rceil$

**Totalt krävs**  $2 (t_s + m t_w) \lceil p^{1/2}/2 \rceil$  tid

### 3-dim nät:

- Samma algoritim i tre steg - 1 för varje dimension:

**Totalt krävs**  $3 (t_s + m t_w) \lceil p^{1/3}/2 \rceil$  tid

## 1-till-alla på hyperkub

### Lagra-vidarebefordra

- Betrakta  $2^d$ -kub som  $d$ -dimensionellt nät med 2 processorer i varje dimension
- Samma algoritim, nu i  $d$  steg ( $= \log p$ )
- Varje steg är enbart en vanlig sändning (1-till-alla broadcast i ring med enbart 2 processorer)

**Totalt krävs**  $(t_s + m t_w) \log p$

## 1-till-alla på hyperkub

$2^3$ -kub  $\rightarrow d = 3$   
 3D nät  
 2 proc i varje dim

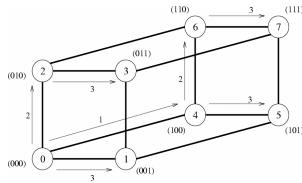


Figure 3.5 One-to-all broadcast on a three-dimensional hypercube. The binary representations of processor labels are shown in parentheses. Copyright (r) 1994 Benjamin/Cummings Publishing Co.

Källa: proc 0

I steg  $i$ : skicka till processor som skiljer sig åt  $i$  bit i  
 Börja (t.ex.) med mest signifikanta biten

## 1-till-alla på hyperkub - algorit

```

1. procedure ONE_TO_ALL_BC( $d, msg\_id, X$ )
2. begin
3.    $mask := 2^d - 1$ ; /* Set all  $d$  bits of  $mask$  to 1 */
4.   for  $i := d - 1$  downto 0 do /* Outer loop */
5.     begin
6.        $mask := mask \text{ XOR } 2^i$ ; /* Set bit  $i$  of  $mask$  to 0 */
7.       if  $(msg\_id \text{ AND } mask) = 0$  then
8.         /* If the lower  $i$  bits of  $msg\_id$  are 0 */
9.         if  $(msg\_id \text{ AND } 2^i) \neq 0$  then
10.          begin
11.             $msg\_destination := msg\_id \text{ XOR } 2^i$ ;
12.            send  $X$  to  $msg\_destination$ ;
13.          end
14.        else
15.          begin
16.             $msg\_source := msg\_id \text{ XOR } 2^i$ ;
17.            receive  $X$  from  $msg\_source$ ;
18.          end
19.        end
20.      end
21.    end
22.  end ONE_TO_ALL_BC
    
```

Program 3.1 One-to-all broadcast of a message  $X$  from processor 0 of a  $d$ -dimensional hypercube. AND and XOR are bitwise logical-and and exclusive-or operations, respectively. Copyright (r) 1994 Benjamin/Cummings Publishing Co.

## 1-till-alla utsändning - (cut-through)

- Hyperkub: ingen tidsvinst
- Ring: använd hyperkubsalgoritmen
  - Tid för ett meddelande begränsas av  $\theta(m+1)$
  - 1:a steget:  $l = p/2$
  - 2:a steget:  $l = p/4$  etc

$$\begin{aligned}
 1. \text{ Totalt krävs: } T &= \sum_{i=1}^{\log p} (t_s + t_w m + t_h p / 2^i) \\
 &= t_s \log p + t_w m \log p + t_h (p-1)
 \end{aligned}$$

Hint:  
 $\sum_{i=1}^{\log p} 1/2^i = (p-1)/p$

## 1-till-alla på ring med hyperkubsalgorit

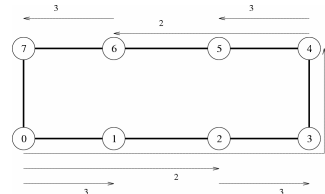


Figure 3.6 One-to-all broadcast with CT routing on an eight-processor ring. Copyright (r) 1994 Benjamin/Cummings Publishing Co.

## Alla-till-alla broadcast

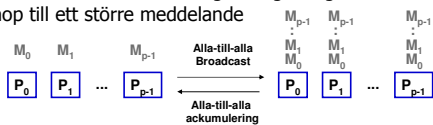
- $p$  processorer gör samtidigt broadcast (olika meddelanden från olika proc.)
- Dual operation: alla-till-alla ackumulering

### Naiv metod:

- Alla gör 1-till-alla utsändning

### Effektivare metod:

- Meddelanden som i ett steg skall gå längs samma väg slås ihop till ett större meddelande



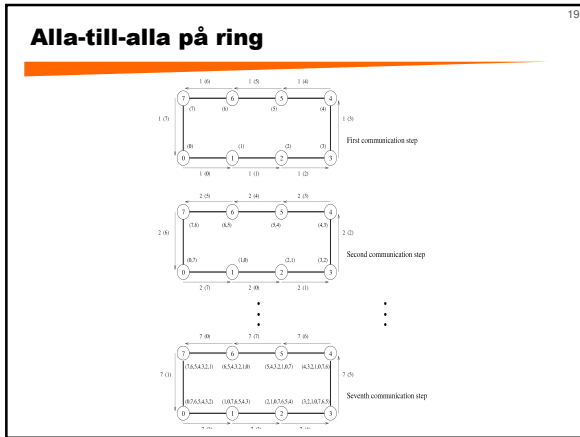
## Alla-till-alla på ring

- 1-till-alla utsändning ger utnyttjande av maximalt 2 länkar samtidigt
- För alla-till-alla utsändning har varje processor i varje steg något den kan vidarebefordra

**Steg 1:** Varje processor skickar sitt meddelande till en granne  
**Steg 2, 3...:** Varje processor skickar vidare det den tog emot i förra steget

- Varje processor tar emot  $p-1$  olika meddelanden i  $p-1$  steg

Totalt krävs:  $(t_s + m t_w) (p-1)$



### Alla-till-alla på 2-dim nät

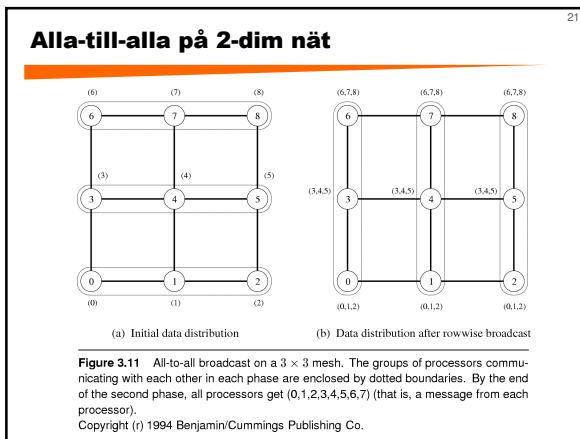
**Steg 1:**

- Alla-till-alla utsändning i varje processorrad  
 $Tid = (t_s + t_w m) (p^{1/2} - 1)$

**Steg 2**

- Alla-till-alla utsändning i varje processorkolumn (varje meddelande är  $m p^{1/2}$ )  
 $Tid = (t_s + t_w m p^{1/2}) (p^{1/2} - 1)$

**Totalt krävs:**  $2 t_s (p^{1/2} - 1) + t_w m (p - 1)$

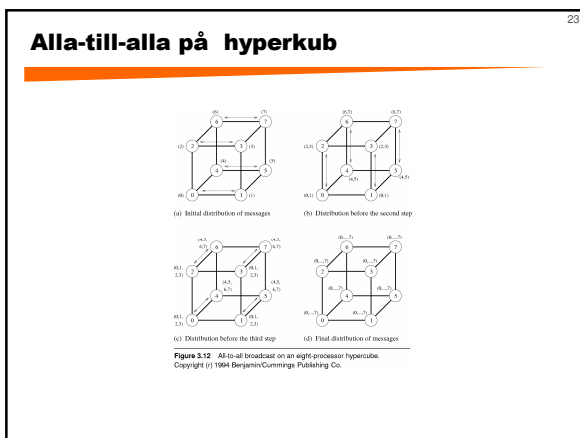


### Alla-till-alla på hyperkub

- 2-dim-nät-algoritmen generaliserad till  $\log p$  dimensioner (i  $\log p$  tidssteg)
- I varje steg utbyter par av processorer meddelanden
- Meddelandena slås ihop till ett (dubbel storlek) i nästa steg
- Storlek av meddelande i steg  $i$  är  $2^{i-1} m$
- Tid för detta:  $t_s + 2^{i-1} t_w m$

**Totalt krävs:**  $\sum_{i=1}^{\log p} (t_s + 2^{i-1} t_w m) = t_s \log p + t_w m (p - 1)$

Hint:  
 $\sum_{i=0}^{p-1} 2^i = p - 1$



### Alla-till-alla broadcast på hyperkub

```

1. procedure ALL_TO_ALL_BC_HCUBE(my_id, my_msg, d, result)
2. begin
3.   result := my_msg;
4.   for i := 0 to d do
5.     begin
6.       partner := my_id XOR 2^i;
7.       send result to partner;
8.       receive msg from partner;
9.       result := result U msg;
10.    endfor;
11. end ALL_TO_ALL_BC_HCUBE
  
```

**Program 3.6** All-to-all broadcast on a  $d$ -dimensional hypercube.  
 Copyright (r) 1994 Benjamin/Cummings Publishing Co.

(Över alla dimensioner med början från den lägsta dimensionen)

## Reduktion på hyperkub

25

**Exempel:** global summering

- 1 Alla processorer har ett värde
- 1 Alla processorer vill veta summan av alla processorer motsvarande värde
- 1 Andra operationer: max, min, OR, AND etc.
- 1 **Alternativ 1:** alla-till-1 ackumulering följt av summering på en nod och 1-till-alla utsändning  $\rightarrow 2(t_s + m t_w) \log p$
- 1 **Alternativ 2:** Kommunikationsmönster som i alla-till-alla utsändning, men istället för att konkatenera meddelanden så adderas resultatet (eller min, max etc).  
– Varje meddelande är endast av storlek  $m$

**Totalt krävs:**  $(t_s + m t_w) \log p$  ( $m = 1$  om reduktion av ett tal)

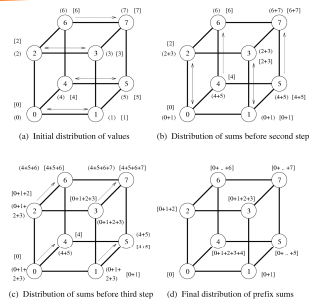
## Prefix summering på hyperkub

26

- 1 Givet  $p$  tal,  $n_0, n_1, \dots, n_{p-1}$  (ett på varje processor)
- 1 Beräkna  $s_k = n_0 + \dots + n_k$  för alla  $k = 0, \dots, p-1$ , så att  $s_k$  ligger på processor  $p_k$
- 1 Som global summering (reduktion med +), men spara varje processors delresultat i speciell buffert
- 1 Inkommande resultat adderas till buffert endast om avsändare är processor med lägre processornummer

## Prefix summering på hyperkub

27



**Figure 3.13** Computing prefix sums on an eight-processor hypercube. At each processor, square brackets show the local prefix sum accumulated in a buffer and parentheses enclose the contents of the outgoing message buffer for the next step.  
Copyright (c) 1994 Benjamin/Cummings Publishing Co.

## Alla-till-alla utsändning med cut-through

28

- 1 Hyperkubsalgoritmen på ring och 2-dimensionellt nät ger ej förbättrad prestanda  $p$  g a kollisioner
- 1 Generellt ej bättre prestanda med direkt-igenom för dessa kommunikationsmönster
- 1 För alla arkitekturer är  $mt_w(p-1)$  undre gräns för tid att utföra alla-till-alla utsändning (mängden data alla tar emot)

**Med lagra-vidarebefordra fick vi:**

- 1 ring:  $(t_s + m t_w) (p-1)$
- 1 2-dim nät:  $2 t_s(p^{1/2}-1) + m t_w(p-1)$
- 1 hyperkub:  $t_s \log p + m t_w(p-1)$

## 1-till-alla personlig kommunikation

29

- 1 Benämns även single-node scatter
- 1 1 proc skickar olika meddelanden till alla
- 1 Dual operation: single-node gather
- 1 Komplexitet ungefär som alla-till-alla utsändning: undre gräns ges av  $p-1$  meddelanden av storlek  $m$ , dvs  $mt_w(p-1)$

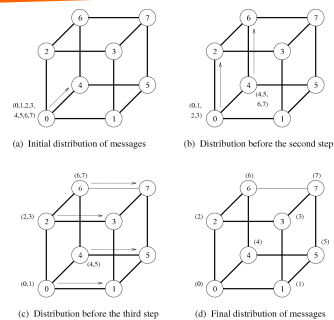
**På hyperkub:**

- 1 Steg 1: 1 proc skickar hälften av meddelandena till granne
- 1 Steg 2, ..., log  $p$ : skicka hälften av meddelandena till granne
- 1 Kommunikation som 1-till-alla utsändning, större meddel.
- 1 Tid =  $t_s \log p + mt_w(p-1)$

(Samma tid som för alla-till-alla broadcast)

## 1-till-alla personlig komm.:hyperkub

30



**Figure 3.16** One-to-all personalized communication on an eight-processor hypercube.  
Copyright (c) 1994 Benjamin/Cummings Publishing Co.

### Alla-till-alla personlig kommunikation

- 1 Kallas även *total exchange*
- 1 Alla skickar olika meddelande till varje annan processor
- 1 Kommunikationsmönster som alla-till-alla utsändning för samtliga 3 arkitekturer (endast storlek skiljer)

#### Ring:

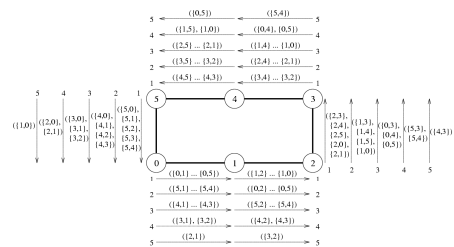
- 1 Steg 1: Varje processor skickar alla  $p-1$  meddelanden till granne (samma riktning)
- 1 Steg 2, ...,  $p-1$ : Varje processor behåller sin del av inkommet meddelande och skickar vidare resten
- 1 I steg  $i$  skickar alla meddelande av storlek  $m(p-i)$  till granne

**Totalt krävs:**

$$\sum_{i=1}^{p-1} (t_s + t_w m(p-i)) = t_s(p-1) + \sum_{i=1}^{p-1} i t_w m$$

$$= (t_s + \frac{1}{2} t_w m p)(p-1)$$

### Alla-till-alla personlig kommunikation: Ring



**Figure 3.18** All-to-all personalized communication on a six-processor ring. The label of each message is of the form  $\{x, y\}$ , where  $x$  is the label of the processor that originally stored the message, and  $y$  is the label of the processor that is the final destination of the message. The label  $\{x_1, y_1\}, \{x_2, y_2\}, \dots, \{x_n, y_n\}$  indicates a message that is formed by concatenating  $n$  individual messages. Copyright (c) 1994 Benjamin/Cummings Publishing Co.

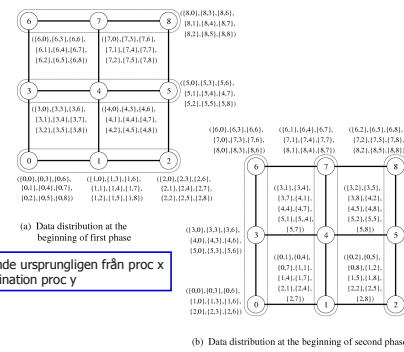
### Alla-till-alla personlig kommunikation: 2D nät

- 1 Varje processor grupperar data efter mottagarprocessors kolumnplacering
- 1 Steg 1: alla-till-alla personlig kommunikation i varje processorrad. Tid som för ring med  $p^{1/2}$  processorer och meddelandestorlek  $mp^{1/2}$
- 1 Steg 2: Motsvarande kommunikation i processorkolumn tar samma tid som steg 1

— Varje processor grupperar data efter "mottagarens" radplacering

**Totalt krävs:**  $(2t_s + t_w mp)(p^{1/2} - 1)$

### Alla-till-alla personlig komm.: 2-dim nät



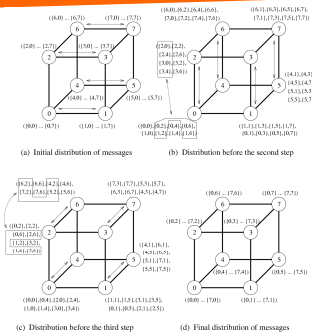
$\{x,y\}$ : meddelande ursprungligen från proc  $x$  med slutlig destination proc  $y$

### Alla-till-alla personlig komm.: hyperkub

- 1 Generalisering av 2-dim-nät-algoritmen till  $\log p$ -dim nät som i  $\log p$  steg utbyter data av storlek  $mp/2$  mellan 2 processorer varje gång

**Totalt krävs:**  $(t_s + t_w mp/2) \log p$

### Alla-till-alla personlig komm.: hyperkub



**Figure 3.20** All-to-all personalized communication on a three-dimensional hypercube with SF routing. Copyright (c) 1994 Benjamin/Cummings Publishing Co.

### Alla-till-alla personlig komm.: Cut-through

37

#### Ring (och 2-dim nät)

- 1 Varje proc skickar totalt meddelanden av storlek  $m(p-1)$
- 1 Genomsnittligt avstånd är  $p/2$
- 1  $p$  processorer gör samma sak
- 1 Total nätbelastning =  $p m(p-1) p/2$
- 1 Antal länkar som delar på belastningen är  $p$
- 1 En undre gräns ges av  $t_w m(p-1) p/2$
- 1 Om startupp-tid försummas är detta samma som SF-algoritmerna (som alltså ej är långt ifrån optimala)
- 1 Notera: Vi har endast använt oss av 1 riktning i varje ring. Om båda riktningarna utnyttjas kan tiden för  $t_w$  halveras både i algoritmerna och i den undre gränsen!

### Alla-till-alla personlig komm.: Cut-through

38

#### Hyperkub:

- 1 Genomsnittligt avstånd =  $(\log p)/2$ , total storlek =  $m(p-1)$  för varje av  $p$  processorer ger **undre gräns:**  
 $(t_w m(p-1)(p \log p)/2) / ((p \log p)/2) = t_w m(p-1)$

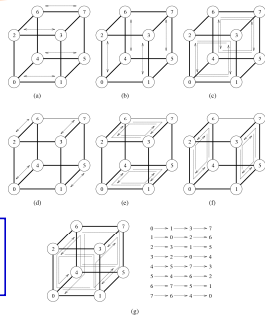
#### Bättre algoritm med cut-through:

- 1 **Steg i:** varje processor skickar direkt meddelande till processors  $i$  minst signifikanta bitar skiljer sig från de egna
- 1 Tid att skicka över  $i$  hops =  $t_s + t_w m + t_h i$
- 1 Summan av alla  $i$ :  $0+1 + \dots + p-1 = (p \log p)/2$

**Totalt krävs:**  $(t_s + t_w m)(p-1) + t_h (p \log p)/2$

### Alla-till-alla personlig komm.: Cut-through

39



Inga kollisioner  
Parvis utbyte  
Dubbelriktade länkar

Figure 3.21 Seven steps in all-to-all personalized communication on an eight-processor hypercube with CT routing. Copyright (c) 1994 Benjamin/Cummings Publishing Co.

### Cirkulär skiftning

40

- 1 Processor  $i$  skickar till processor  $(i+q) \bmod p$  ( $0 < q < p$ )

#### 2-dim nät:

- **Steg 1:** Allt data skickas  $(q \bmod p^{1/2})$  steg längs raderna.
- **Steg 2:** Allt data skickas  $\lfloor q/p^{1/2} \rfloor$  längs kolumnerna
- **Steg 1.5:** I steg 1 skiftades en del data över wrap-around-länken till processor med lägre nummer. Allt detta data skall som korrigerings skickas 1 steg längs kolumnerna.
- **Förbättring:** Skift gör i den riktning det är närmast (t ex i 4-processors rad motsvarar ett 3-skift ett 1-stegs bakåtskift). Detta ger maximalt  $\lfloor p^{1/2}/2 \rfloor$  steg i varje riktning

### Cirkulär skiftning på 2-dim nät

41

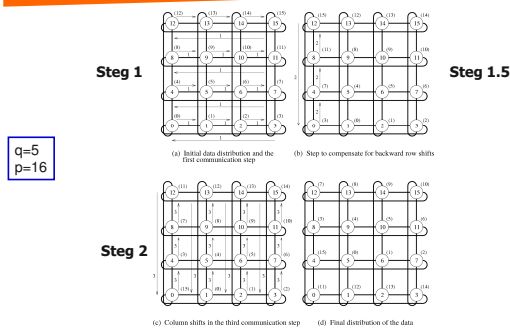


Figure 3.22 The communication steps in a circular S-shift on a  $4 \times 4$  mesh. Copyright (c) 1994 Benjamin/Cummings Publishing Co.

### Cirkulär skiftning på hyperkub

42

- 1 Avbilda en ring på kub med processorer numrerade i BRGC-ordning
- 1 Processorer med differens  $2^0$  är grannar
- 1 Processorer med differens  $2^i$ ,  $i > 0$  är på avstånd 2
- 1 Skiftning i  $q$  steg görs genom att  $q$  tolkas binärt och ett stegs skift görs för varje 1:a i den binära koden
- 1 T ex  $q = 5$ , ger binär kod = 101. Vi skiftar fyra processornummer för 100 och ett 1-skift för 001
- 1 Maximalt antal steg är  $\log p$ , där alla steg utom granngranne är över två länkar

**Övre gräns för tid:**  $(t_s + t_w m)(2 \log p - 1)$

- 1 Genom att utnyttja både bakåt och framåt-skift kan detta minskas till  $(t_s + t_w) \log p$

## Cirkulär skiftning på hyperkub

43

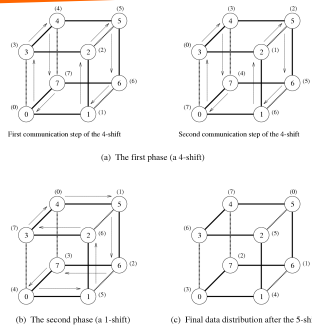


Figure 3.23 The mapping of an eight-processor ring onto a three-dimensional hypercube to perform a circular 5-shift as a combination of a 4-shift and a 1-shift. Copyright © 1994 Benjamin/Cummings Publishing Co.

## Optimering av 1-till-1-kommunikation

44

### Skicka ett meddelande i delar på hyperkub

- 1 Mellan varje par av noder existerar  $\log p$  distinkta vägar, varav  $l$  stycken är av längden  $l$  och  $\log p - l$  är av längden  $l+2$  (där  $l$  är antal olika bitar i processornumren)
- 1 Meddelandet kan delas i  $\log p$  delar och skickas längs olika vägar med lång väg först
- 1 Sista paketet tar maximalt  $\log p$  steg

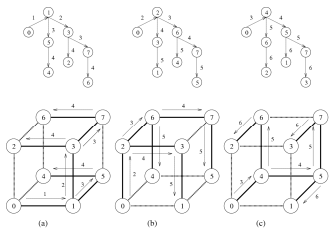
**Totalt krävs maximalt:**  $2(t_s \log p + t_w m)$

- 1 I vår tidigare algoritm var övre gränsen  $t_s + t_w m \log p$

## Optimering av 1-till-alla-utsändning

45

- 1  $p$  stycken olika uppspännade binomiala träd kan avbildas på en hyperkub med samma nod som rot
- 1 Meddelande som skall skickas delas i  $\log p$  delar. En del skickas längs varje träd



## Sammanfattning av kommunikationstider

46

Table 4.1 Summary of communication times of various operations discussed in Sections 4.1–4.7 on a hypercube interconnection network. The message size for each operation is  $m$  and the number of nodes is  $p$ .

Operation	Hypercube Time	B/W Requirement
One-to-all broadcast, All-to-one reduction	$\min(t_s + t_w m) \log p, 2(t_s \log p + t_w m)$	$\Theta(1)$
All-to-all broadcast, All-to-all reduction	$t_s \log p + t_w m(p - 1)$	$\Theta(1)$
All-reduce	$\min(t_s + t_w m) \log p, 2(t_s \log p + t_w m)$	$\Theta(1)$
Scatter. Gather	$t_s \log p + t_w m(p - 1)$	$\Theta(1)$
All-to-all personalized	$(t_s + t_w m)(p - 1)$	$\Theta(p)$
Circular shift	$t_s + t_w m$	$\Theta(p)$