Basic Communication Operations

• Possible variants
  – # of nodes involved
    • Point-to-point vs collective operation
  – routing scheme
    • Store-and-Forward (S&F), Cut-Through (CT) and Packet Routing

• Usually point-to-point implemented in hardware, collective in software

• Many of the collective have a dual operation
  – the dual can be performed reversing the direction and sequence of messages in the original operation
Point-to-point

- Store-and-forward => $t_{comm} \approx t_s + l m t_w$
  - ring
    - $l = \lceil p/2 \rceil$
    - $t_{comm} = t_s + \lceil p/2 \rceil m t_w$
  - mesh
    - $l = 2 \lceil \sqrt{p/2} \rceil$
    - $t_{comm} = t_s + 2 \lceil \sqrt{p/2} \rceil m t_w$
  - hypercube
    - $l = \log p$
    - $t_{comm} = t_s + m t_w \log p$

- Cut-through (or Packet) => $t_{comm} = t_s + l t_h + m t_w$
  - Small messages: $CT \approx S&F \approx t_s + l t_h$
  - Large messages: $CT \approx t_s + m t_w$ (no dependence from $l$)
One-to-all broadcast

- A.k.a single-node broadcast
  - message of size $m$ on source processor
  - at the end of the operation message is replicated on all other procs

- Dual operation: single-node accumulation (a.k.a reduce operation)
  - initially every processor has message of size $m$
  - at the end, combination of all messages is on single destination proc
  - combination is through an associative operation (sum, product, max, min)
Broadcast over mesh: example

- Multiplication of 4 x 4 matrix with a 4 x 1 vector
Broadcast on ring (S&F)

- Number of steps: $\lceil p/2 \rceil$
- Latency of communication step: $t_s + mt_w$
- Total duration: $T_{one\_to\_all} = (t_s + mt_w) \lceil p/2 \rceil$
Broadcast on mesh (S&F)

- Row/column broadcast time:
  - \((t_s + m t_w) \lceil \sqrt{p/2} \rceil\)

- Total duration:
  - \(T_{\text{one to all}} = 2(t_s + m t_w) \lceil \sqrt{p/2} \rceil\)

- 3D mesh
  - \(T_{\text{one to all}} = 3(t_s + m t_w) \lceil p^{1/3}/2 \rceil\)
Broadcast on hypercube (S&F)

- Total duration: $T_{one\_to\_all} = (t_s + mt_w) \log p$
Broadcast on hypercube: algorithm

**Procedure** ONE_TO_ALL_BC\((d, \text{my\_id}, X)\)

begin

\[ mask := 2^d - 1 \quad /* Set all bits of mask to 1 */ \]

for \(i := d - 1\) downto 0 do /* Outer loop */

begin

\[ mask := mask \ XOR 2^i \quad /* Set bit \(i\) of mask to 0 */ \]

if \((\text{my\_id AND mask}) = 0\) then

/* the lower \(i\) bits of \text{my\_id} are 0 */

if \((\text{my\_id AND } 2^i) = 0\) then

begin

\[ \text{msg\_destination} := \text{my\_id XOR } 2^i \]

send \(X\) to \text{msg\_destination}

end

else

begin

\[ \text{msg\_source} := \text{my\_id XOR } 2^i \]

receive \(X\) from \text{msg\_source}

end

endfor

end ONE_TO_ALL_BC

Only nodes with last \(i\) bits equal to 0 participate in communication in \(i\)th iteration

If my \(i\)th bit is 0, I am a sender; otherwise I am a receiver
Dual of Broadcast: single-node Accumulation

**Procedure** ONE_TO_ALL_BC($d$, $my_id$, $X$)

begin

$mask := 2^d - 1$ /* Set all bits of $mask$ to 1 */

for $i := d - 1$ downto 0 do /* Outer loop */

begin

$mask := mask \text{ XOR } 2^i$ /* Set bit $i$ of $mask$ to 0 */

if ($my_id$ AND $mask$) = 0 then

/* the lower $i$ bits of $my_id$ are 0 */

if ($my_id$ AND $2^i$) = 0 then

begin

$msg\_destination := my_id \text{ XOR } 2^i$

send $X$ to $msg\_destination$

end

else

begin

$msg\_source := my_id \text{ XOR } 2^i$

receive $X$ from $msg\_source$

end

endfor

end ONE_TO_ALL_BC

**Procedure** SINGLE_NODE_ACC($d$, $my_id$, $m$, $X$, $sum$)

begin

for $j := 0$ to $m - 1$ do $sum[j] := X[j]$

$mask := 0$

for $i := 0$ to $d - 1$ do

begin /* select node whose lower $i$ bits are 0 */

if ($my_id$ AND $mask$) = 0 then

if ($my_id$ AND $2^i$) \neq 0 then

begin

$msg\_destination := my_id \text{ XOR } 2^i$

send $sum$ to $msg\_destination$

end

else

begin

$msg\_source := my_id \text{ XOR } 2^i$

receive $X$ from $msg\_source$

for $j := 0$ to $m - 1$ do $sum[j] := sum[j] + X[j]$

end

end

end

end SINGLE_NODE_ACC
Broadcast on ring (CT)

- Latency of communication at step $i$: $t_s + mt_w + t_h p/2^i$
- Total duration:
  
  \[
  T_{\text{one-to-all}} = \sum_{i=1}^{\log p} (t_s + mt_w + t_h p/2^i) = t_s \log p + mt_w \log p + t_h (p - 1)
  \]
Broadcast on mesh (CT)

- Row/column broadcast time:
  $$- (t_s + mt_w) \log \sqrt{p} + t_h (\sqrt{p} - 1)$$

- Total duration:
  $$- (t_s + mt_w) \log p + 2t_h (\sqrt{p} - 1)$$
Broadcast on binary tree (CT)

- Hypercube algorithm
  - there are different number of switches traversed along different paths

- Total duration:
  - $T_{one\_to\_all} = (t_s + mt_w + t_h (\log p + 1))\log p$
All-to-All Broadcast

- **A.k.a multinode broadcast**
  - message of size $m$ on each processor
  - at the end of the operation messages are replicated on all procs

- **Dual operation: multinode accumulation (a.k.a personalized reduction operation)**
  - each processor is the destination of a single-node accumulation
  - combination is through an associative operation (sum, product, max, min)
A2A Broadcast on Ring (S&F)

- Number of steps: $p - 1$
- Latency of each communication step: $t_s + mt_w$
- Total duration: $T_{all\_to\_all} = (t_s + mt_w) (p - 1)$
A2A Broadcast on mesh (S&F)

- Row broadcast time: \((t_s + mt_w) (\sqrt{p} - 1)\)
- Column broadcast time: \((t_s + \sqrt{p} \cdot mt_w) (\sqrt{p} - 1)\)
- Total duration: \(T_{all\_to\_all} = 2t_s(\sqrt{p} - 1) + mt_w(p - 1)\)
A2A Broadcast on hypercube (S&F)

- Duration of step $i$: $t_s + mt_w 2^{i-1}$
- Total duration:
  - $T_{all\_to\_all} = \sum_{i=1}^{\log p} (t_s + mt_w 2^{i-1}) = t_s \log p + mt_w (p - 1)$