Collective Communication Operations

• Reading
  – Kumar et al., Basic Communication Operations
Objectives

• Examine network-specific implementations of collective communication operations
  – derive analytic costs for three representative networks
    » Ring
    » Torus
    » Hypercube

  – and two routing models
    » Store-and-Forward
    » Cut-through

• Implications for the BSP model
Networks considered

- **Ring**
  - Diameter $p/2$
  - Bisection width 2

- **2-D torus**
  - Diameter $2(\sqrt{p} / 2 - 1) \approx \sqrt{p}$
  - Bisection width $2\sqrt{p} \approx \sqrt{p}$

- **Hypercube**
  - Diameter $(\lg p)$
  - Bisection width $p/2 \approx p$
Network assumptions

• Communication cost model
  – Message size $m$ bits
  – Number of hops to travel $h$
  – Channel width $W$ in bits and channel cycle time $t_c$
    » per-bit transfer time $t_w = t_c / W$
    » transit time for message to cross channel $t_w m$
  – Startup time $t_s$
  – Node latency or per-hop time $t_h$
    » time taken by message header to cross one link and be switched to the next link

• Network model
  – Bi-directional communication links
  – Single-port communication model for source and destination
    » each processor can perform at most one send and one receive simultaneously
  – Multiport switches
    » each switch can permute inputs to outputs
    » contention for outputs causes serialization
Flow control strategy: SF and CT

- **Store and Forward (SF)**
  - packet buffered at each node
  \[ t_{SF} = t_S + (t_W m)h \]

- **Cut-through (CT)**
  - packet spread through network
  \[ t_{CT} = t_S + t_W m + t_h h \]
Simple message transfer

- Single sender, single receiver, single message size $m$, worst case time
  - diameter $d$ of network provides upper bound
    - SF: $t_{SF} = t_S + (t_W m)d$
      - ring: $t_{SF} = t_S + (t_W m)(p/2)$
      - 2-D torus: $t_{SF} = t_S + (t_W m)p^{1/2}$
      - Hypercube: $t_{SF} = t_S + (t_W m)(\lg p)$
    - CT: $t_{CT} = t_S + t_W m + t_h d$
      - ring: $t_{CT} = t_S + t_W m + t_h (p/2)$
      - 2-D torus: $t_{CT} = t_S + t_W m + t_h p^{1/2}$
      - Hypercube: $t_{CT} = t_S + t_W m + t_h \lg p$

With CT and $m$ large, all networks achieve approximately same performance

$$t_{CT} = t_S + t_W m + t_h d \approx t_W m$$
One-to-all broadcast (m)

\[ R_i = A_i \oplus B_i \oplus C_i \oplus D_i \]
One-to-all broadcast: (Ring, SF)

- Single sender, one common message, multiple receivers

Step 1

Step 2

Step 3

Step 4
One-to-all broadcast: (Torus, SF)

- Extend (Ring, SF) solution to each dimension in turn
- For 2-dimensional torus:
  (a) One-to-all broadcast from source along row, then
  (b) One-to-all broadcast in each column simultaneously
One-to-all broadcast (Hypercube, SF)

- Hypercube is extreme case of k-ary d-cube, with d = \( \lg P \) dimensions of k = 2 processors each
  - broadcast in each dimension requires a single step

```
0 1 0
0 1 1
0 0 0
0 0 1
1 1 0
1 1 1
1 0 0
1 0 1
```

```
0 1 0
0 1 1
0 0 0
0 0 1
1 1 0
1 1 1
1 0 0
1 0 1
```

```
0 1 0
0 1 1
0 0 0
0 0 1
1 1 0
1 1 1
1 0 0
1 0 1
```

```
0 1 0
0 1 1
0 0 0
0 0 1
1 1 0
1 1 1
1 0 0
1 0 1
```
A lower bound for one-to-all bcast

• **Claim:** With single-port communication model, no topology can do better than (Hypercube, SF) for one-to-all broadcast
  – At each step, each processor with data sends to a processor that needs data
  – Communication happens between neighboring processors

• **This argument ignores**
  – Dependence of $t_w$ and $t_s$ on wire length
  – (Multiport communication)
One-to-all broadcast (Ring, CT)

- **Observation:** Distance term is relatively insignificant with CT
- **Key idea:** Adapt (HC, SF) algorithm
  - At step $i \in 1 : \lg P$, send to processor at (anticlockwise) distance $P/2^i$
One-to-all broadcast (Torus + HC, CT)

- **Torus**
  - one-to-all broadcasts using CT in each successive dimension

\[
t_s \lg p + 2t_h (\sqrt{p} - 1) + t_w m \lg p
\]

- **Hypercube**
  - no advantage for CT, since all communications are single-step.
**SUMMARY: One-to-all broadcast**

- **communication size**

<table>
<thead>
<tr>
<th>source</th>
<th>network</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>$m$</td>
<td>$m$</td>
</tr>
</tbody>
</table>

- **communication time**

<table>
<thead>
<tr>
<th>Network</th>
<th>SF</th>
<th>CT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring</td>
<td>$(t_s + t_w m) \left\lceil \frac{p}{2} \right\rceil$</td>
<td>$t_s \lg p + t_h (p - 1) + t_w m (\lg p)$</td>
</tr>
<tr>
<td>2-D Torus</td>
<td>$2(t_s + t_w m) \left\lceil \frac{\sqrt{p}}{2} \right\rceil$</td>
<td>$t_s \lg p + 2t_h (\sqrt{p} - 1) + t_w m (\lg p)$</td>
</tr>
<tr>
<td>Hypercube</td>
<td>$(t_s + t_w m) \lg p$</td>
<td>$(t_s + t_w m) \lg p$</td>
</tr>
</tbody>
</table>
All-to-all broadcast

all-to-all broadcast (m)

\[ A_0 \oplus B_0 \oplus C_0 \oplus D_0 \]

all-to-all sum (m)

\[ R_i = A_i \oplus B_i \oplus C_i \oplus D_i \]
All-to-all broadcast

- Each processor has information that it sends to all other processors
  - $p$ senders
  - $p$ messages
  - $p-1$ receivers of each message

- Example
  - distribution of vector in BSP Matrix * Vector Algorithm

- Naive solution: perform $p$ independent one-to-all broadcasts
  - Costs $p$ times more than single one-to-all broadcast

- Better solution: pipeline the broadcasts
Collective Communication

All-to-all broadcast (Ring, SF)

Ex: \( p = 6 \)

\[
t_{SF}^{ring} = \sum_{i=1}^{p-1} (t_s + t_{wm}) = (p-1)t_s + (p-1)t_{wm}
\]
All-to-all broadcast (2-D Torus, SF)

- Use ring algorithm once in each dimension
- In the second dimension, the size of the message to be broadcast increases by a factor of $p^{1/2}$

$$t_{SF}^{torus} = (\sqrt{p} - 1)t_s + (\sqrt{p} - 1)t_w m + (\sqrt{p} - 1)t_s + (\sqrt{p} - 1)t_w (m\sqrt{p})$$

$$= 2(\sqrt{p} - 1)t_s + (p - 1)t_w m$$
All-to-all broadcast (Hypercube, SF)

- Use ring algorithm consecutively in each dimension. The size of the message doubles with each consecutive dimension.

\[
t_{hyps}^{SF} = \sum_{i=1}^{\lfloor \log p \rfloor} t_s + t_w 2^{i-1} m = (\log p) t_s + (p - 1) t_w m
\]
All-to-all broadcast (CT)

• CT doesn’t help
  – Hypercube
    » all communication is distance 1
  – Ring & Torus
    » mapping HC algorithm to ring causes link congestion
    » can’t do much better anyway: \((p-1)mt_w\) is a lower bound, since each processor must receive \((p-1)m\) data
SUMMARY: All-to-all broadcast

- **communication size**

<table>
<thead>
<tr>
<th>source</th>
<th>network</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>(m)</td>
<td>(pm)</td>
<td>(pm)</td>
</tr>
</tbody>
</table>

- **communication time**

<table>
<thead>
<tr>
<th>Network</th>
<th>SF</th>
<th>CT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring</td>
<td>((t_s + t_w m)(p - 1))</td>
<td>(same)</td>
</tr>
<tr>
<td>2 - D Torus</td>
<td>(2t_s(\sqrt{p} - 1) + t_w m(p - 1))</td>
<td>(same)</td>
</tr>
<tr>
<td>Hypercube</td>
<td>(t_s \log p + t_w m(p - 1))</td>
<td>(same)</td>
</tr>
</tbody>
</table>
One-to-all personalized communication

- One-to-all personalized communication (m)
  - a.k.a. single-node scatter

- All-to-one personalized communication (m)
  - a.k.a. single-node gather
One-to-all personalized communication (Scatter, Ring, SF)

\[ t_{SF}^{\text{ring}} = \sum_{i=1}^{p-1} (t_s + t_{wm}) = (p-1)t_s + (p-1)t_{wm} \]
One-to-all personalized communication (Torus, SF)

- **Stage 1**
  - one-to-all personalized communication in single row, data size \( mp^{1/2} \)

- **Stage 2**
  - one-to-all personalized communication in all columns, data size \( m \)

\[
t_{SF}^{torus} = (\sqrt{p - 1})(t_s + t_w m \sqrt{p}) + (\sqrt{p - 1})(t_s + t_w m) = 2(\sqrt{p - 1})t_s + (p - 1)t_w m
\]
One-to-all personalized communication (HC, SF)

\[ t_{SF}^{hypc} = \sum_{i=1}^{\log p} t_s + t_w m \frac{p}{2^i} = (\log p) t_s + (p - 1) t_w m \]
One-to-all personalized communication (Ring, CT)

- Adapt (HC, SF) algorithm
  - At step $i \in 1 : \lg P$, send to processor at (anticlockwise) distance $P/2^i$
SUMMARY: One-to-all personalized communication

- **CT is not much help**
  - source must send $m(p - 1)$ data, and SF implementations already at $m(p - 1)t_w$ bandwidth bound
  - possibly decrease in latency using SF Hypercube algorithm in ring with CT
    » improvement only if $t_s > t_h$

- **communication size**
  
<table>
<thead>
<tr>
<th>source</th>
<th>network</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>$pm$</td>
<td>$pm$</td>
<td>$m$</td>
</tr>
</tbody>
</table>

- **communication time**

<table>
<thead>
<tr>
<th></th>
<th>SF</th>
<th>CT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring</td>
<td>$t_s(p - 1) + t_wm(p - 1)$</td>
<td>$t_s\lg p + t_h(p - 1) + t_wm(p - 1)$</td>
</tr>
<tr>
<td>2-D Torus</td>
<td>$2t_s\left(\sqrt{p - 1}\right) + t_wm(p - 1)$</td>
<td>$t_s\lg p + 2t_h\left(\sqrt{p - 1}\right) + t_wm(p - 1)$</td>
</tr>
<tr>
<td>Hypercube</td>
<td>$t_s\lg p + t_wm(p - 1)$</td>
<td>(same)</td>
</tr>
</tbody>
</table>
All-to-all personalized communication

- all-to-all exchange (m)
  - a.k.a. total exchange (m)
All-to-all personalized communication (Ring, SF)

\[
t_{\text{SF}}^{\text{ring}} = \sum_{i=1}^{p-1} \left( t_s + t_w m(p-i) \right) = (p-1)t_s + (p-1)\frac{p}{2} t_w m
\]
All-to-all personalized communication (HC, SF)

- **Full exchange in each dimension**
  - ex: successive elements at processor 0 on left, values in destination proc on right

\[
t_{SF}^{hypc} = \sum_{i=1}^{\lg p} \left(t_s + t_w m \frac{p}{2}\right) = (\lg p) t_s + (\lg p) \frac{p}{2} t_w m
\]
All-to-all personalized communication (HC, CT)

- CT can improve performance
  - eliminate \((\lg p)\) intermediate destinations for each personalized message
  - replace with \(p-1\) communication phases
    - phase \(0 \leq i < p\)
      - pairwise direct exchange of personalized message of size \(m\)
      - proc \(j\) communicates with proc \((j \oplus i)\)
    - each phase of pairwise communications is contention-free
  - bandwidth term is optimal

\[
\tau_{CT}^{hypc} \leq \sum_{i=1}^{p-1} (t_s + t_h \lg p + t_w m) = (p - 1)t_s + (p - 1)(\lg p)t_h + pt_w m
\]
SUMMARY: All-to-all personalized communication

- **communication size**

<table>
<thead>
<tr>
<th>source</th>
<th>network</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>(pm)</td>
<td>(p^2m)</td>
<td>(pm)</td>
</tr>
</tbody>
</table>

- **communication time**

<table>
<thead>
<tr>
<th></th>
<th>SF</th>
<th>CT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ring</td>
<td>(t_s(p-1) + t_w m \frac{p}{2}(p-1))</td>
<td>(same)</td>
</tr>
<tr>
<td>2-D Torus</td>
<td>(2t_s(\sqrt{p}-1) + 2t_w m \frac{p}{2}(\sqrt{p}-1))</td>
<td>(same)</td>
</tr>
<tr>
<td>Hypercube</td>
<td>(t_s \lg p + t_w m \frac{p}{2}(\lg p))</td>
<td>(t_s(p-1) + t_h \frac{p}{2}(\lg p) + t_w m(p-1))</td>
</tr>
</tbody>
</table>

- **Low bisection-width networks (tori) really cannot match BSP costs in this case**