Lecture 26: Performance Models for Distributed Memory Parallel Computing

William Gropp

www.cs.illinois.edu/~wgropp
Overview

- Simple model of communication – $s+rn$
- LogP – adding overhead
- LogGP – adding long messages
- Hop Count – approximating contention (among other things)
Simple Model Of Communication – Two Parties

• $T = s + rn$ model
  ♦ $T = \text{latency} + \frac{\text{length}}{\text{bandwidth}}$
  ♦ $s = \text{latency}$
  ♦ $r = \frac{1}{\text{bandwidth}}$

• On modern HPC systems, latency is 1-10usec and bandwidths are 0.1 to 10 GB/sec
What Does s Contain?

- All costs for a short message to be sent from user program to user program
  - Including data that describes message
    - \( s = s_0 + r n_e \), \( n_e \) = size of message “envelope”
- Can have separate parameter values for different cases:
  - Programming models (e.g., due to semantics of operations, such as required copies)
  - Implementations (quality of implementation)
  - Networks within a single machine
    - Intrachip, intranode, internode
What Does $r$ Contain?

• $r$ is $1$/minimum of rate along path
  - That is, the achieved rate is limited by the slowest part of the path from one process to another

• $r$ includes contributions from
  - Software to move data at each end, e.g., the rate at which software can feed the hardware
  - Hardware along each link, e.g., the rate that data moves along the wires or fibers
Contributions to r

- Example path of data from one node to another
Improving the Model: LogP

- Represent time as separate components:
  - Latency (hardware)
  - overhead (software)
  - gap (inverse of bandwidth; seconds per message)
  - p (processors (nodes))
  - For analysis, measured in terms of processor cycles

- All maximum times
  - Used for analysis – like our performance expectation; not intended for prediction
Visualizing LogP

P0

P1

O

L
Working with LogP

• Short messages (single message packet):
  ♦ 2o+L

• Finite capacity of network
  ♦ \( \text{Ceil}(L/g) \) messages in transit between any pair of nodes

• Long messages
  ♦ Pipeline of depth L with rate g and overhead o (at each end)
    • Depth L because it takes L units of time for message to travel through network and one message every g units of time. You’d like g = 1, but it might not.
Why Separate Latency and Overhead?

- Latency is Hardware – including time for data to traverse network
  - Question: What is the difference in distance (measured in clock cycles) between close and far nodes in large machine like BW?
  - Some facts:
    - Speed of light is about 30cm/nanosecond
    - Large systems are $O(10,000)$ sq ft
One Answer

• Nearby nodes are less than 15cm apart
  ♦ For 2GHz clock, that is 1 clock cycle
• Far away nodes may be
  \[2 \times \sqrt{10,000 \text{ft}^2} = 2 \times 100 \text{ft} = 2 \times 100 \times 30 \text{cm} = 6000 \text{cm}\]
• \[6000 \text{cm} / 15 \text{cm} / \text{clock} = 400 \text{ clock cycles}\]
  ♦ Only 0.2 usec
• Note speed of signal in wire < speed of light; distance is minimum possible rather than typical
Why Separate Latency and Overhead?

• Overhead is involvement of CPU
• Significant difference between message passing (matching) and put/get (e.g., PGAS)
  ♦ Message passing: receiver must find matching receive in a queue of posted but unmatched receives or save information on the message in a queue of unexpected messages
  ♦ Overhead typically scales linearly with the number of messages in the queue
    • Linear algorithms fastest when queues nearly empty
Why no Topology in LogP?

- Question for class:
  - Average distance in graph for 3D mesh and a hypercube
    - \( P = 1024 \) (time LogP paper written)
    - \( P = 32,768 \) (slightly larger than Blue Waters)
    - \( P = 98304 \) (LLNL Sequoia)

- The authors of LogP contend that contention should be fixed in the network hardware (see Section 5.6 in the paper)
## Average Number of Hops

<table>
<thead>
<tr>
<th>Network</th>
<th>Average Distance</th>
<th>$P=1024$</th>
<th>$P=32,768$</th>
<th>$P=98,304$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypercube</td>
<td>$\frac{1}{2} \log p$</td>
<td>5</td>
<td>7.5</td>
<td>8.29</td>
</tr>
<tr>
<td>Butterfly</td>
<td>$\log p$</td>
<td>10</td>
<td>15</td>
<td>16.6</td>
</tr>
<tr>
<td>4th degree Fat Tree</td>
<td>$2\log_4 p - 2/3$</td>
<td>9.33</td>
<td>14.3</td>
<td>15.9</td>
</tr>
<tr>
<td>3D Torus</td>
<td>$\frac{3}{4} p^{1/3}$</td>
<td>7.5</td>
<td>24</td>
<td>34.6</td>
</tr>
<tr>
<td>3D Mesh</td>
<td>$p^{1/3}$</td>
<td>10</td>
<td>32</td>
<td>46.2</td>
</tr>
<tr>
<td>2D Torus</td>
<td>$\frac{1}{2} p^{1/2}$</td>
<td>16</td>
<td>90.5</td>
<td>157</td>
</tr>
<tr>
<td>2D Mesh</td>
<td>$\frac{2}{3} p^{1/2}$</td>
<td>21</td>
<td>121</td>
<td>209</td>
</tr>
</tbody>
</table>
Contributions to r Revisited

- Example path of data from one node to another: Using remote direct memory access
More on Long Messages: LogGP

• The LogP model targets short messages, or messages made up of a sequence of short messages (the “g” term)

• Features such as RDMA mean that long messages may have a different rate.

• The LogGP model introduces an additional parameter G used for long messages
More on Topology and Contention

- Vendors often insist that topology no longer matters
- Evidence (and logic) say otherwise
- See Bhatele (Ph.D. thesis and numerous papers); introduced hop count metric

This example from IBM BG/P using messages between equidistant pairs; from “Quantifying Network Contention on Large Parallel Machines”, Bhatele and Kale
Hop Count

- L becomes L(h) and roughly h*L(1)
- Use of hop count and hop bytes
  - Communication time increases with increasing hop count, thus
  - Performance decreases as average hop count increases
- Thus arrange
  - Algorithm to have low hop count
  - Mapping of processes to core/chip/node to (approximately) minimize hop count
Hop Count and LogP

- LogP rejected topology – why consider hop count?
  - Machines larger, gap and overhead smaller. Thus variation in latency is significant (more than an order of magnitude)
    - Just a constant term \( c \) can be ignored in theoretical analysis
    - A big constant term \( c \) cannot be ignored in performance expectations
  - LogP assumes networks/programming systems will have low contention on network links
    - Not true, even for fast, high-radix switched networks
      - Avoiding Hot-Spots on two-level direct networks, Bhide, Jain, Gropp, Kale, SC2011
- Recall ring example (lecture 20, slide 35)
  - Effective bandwidth = \( (1/k) \times \text{peak bandwidth} \)
  - \( K = \text{hop count} \)
Including Contention in the Performance Model

- Hard. Made harder by innovation in the network hardware that tries to reduce the impact of contention
  - Adaptive routing
    - Rather than a fixed route, each switch picks route to avoid very busy links while still moving toward destination
    - Local decisions can still lead to contention
  - Timing critical
    - Finite resources at each switch may be exceeded in bursts but ok if paced properly (though that’s almost impossible to accomplish)
Simulation

• Use the computer to simulate the network, using simplified rules for message transit through the network
  ♦ Injection
  ♦ Switching
• Many tools, both open source and proprietary
• A few examples:
  ♦ Bigsim [http://charm.cs.uiuc.edu/research/bigsim](http://charm.cs.uiuc.edu/research/bigsim)
  ♦ ORCS [http://htor.inf.ethz.ch/research/orcs/](http://htor.inf.ethz.ch/research/orcs/)
  ♦ LogGOPSim [http://htor.inf.ethz.ch/research/LogGOPSim/](http://htor.inf.ethz.ch/research/LogGOPSim/)
Emulation

- Like simulation, but much more detailed and accurate modeling of network
  - Needs many details (some trade secrets) of the hardware
  - Very likely to be much slower than simulation
- Because more accurate, can expose foibles of the specific design, such as buffer exhaustion and problems with adaptive routing method
Worst Case Analysis

• Pick a routing strategy and network, then essentially do what simulation would do, but use worst case at each time/location to simplify the analysis
  ♦ Pro: parameterized; one analysis applies to many cases
  ♦ Con: big simplification, can significantly overestimate communication time
Assume that adaptive routing is perfect. Then one limit to network performance is the total capacity of the network – the number of bytes (or message packets) in transit at any time:

- 1-D mesh: \( p - 1 \) links
- 2-D mesh: \( 2(p - p^{1/2}) \) links
- 3-D mesh: \( 3(p - p^{2/3}) \) links

Another limit is the ability of the nodes to fill the network:

- This is the injection rate limit
- Determined by the rate at which nodes can inject data into the network
Relationship Between Capacity and Hop Count

• Higher average hop count increases the amount of data *in* the network at any one time, assuming either long messages or large numbers of small messages.
Nonblocking and Asynchronous

- Nonblocking in MPI only describes whether a routine blocks the process during an operation.
  - Not whether the communication and computation can take place concurrently
    - Sometimes called asynchronous communication

- Performance models must distinguish these cases
  - MPI implementations may offer different modes, each of which has different tradeoffs
  - E.g., MPICH_ASYNC_PROGRESS
    - Establishes separate communication thread
    - Now requires thread safe implementation, which increases overhead $o$ (and may increase the gap $g$)
Readings

  ♦ http://dl.acm.org/citation.cfm?doid=240455.240477

Questions for Discussion

- Express $s + r_n$ using the parameters of:
  - $\log p$
  - $\log G_p$
Some Solutions

- For LogP:
  - $s = 2^o + L$
  - Could add a term for the message envelope
  - $r = 1/(gw)$, where $w$ is the length of the message sent

- For LogGP
  - $s = 2^o + L$
  - $r = 1/G$
  - Since $s + rn$ typically uses $r$ for the asymptotically large message time