Communication Cost in Parallel Computing

Ned Nedialkov

McMaster University
Canada

SE/CS 4F03
January 2016

Outline

Message passing cost

The time for passing a message includes

- startup time, t_s
- pre-hop time, t_h
- pre-word time, t_w

We consider these times in order
A message consists of a header and the actual data

Startup time t_s

Startup time t_s includes time

- to prepare a message
  - adding header, trailer, error correction information
- to execute a routing algorithm
- to establish interface between the local node and the router

Startup time occurs once per message

Pre-hop time t_h

- This is the time for a header to travel between two directly connected nodes (one link)
- Also called node latency

Pre-word transfer time t_w

- This is the time for a word to traverse a link
- Assume a channel bandwidth is r words/sec
  Then

$$t_w = \frac{1}{r}$$

Store-and-forward routing

Each node

- receives a message
- stores it completely
- forwards it

Assume a message of size m and l links
The time for sending the message over l links is

$$t_{\text{comm}} = t_s + t_h l + m t_w l = t_s + l(t_h + t_w m)$$

$$t_{\text{comm}} \approx t_s + m l t_w$$

t_h is usually very small

Packet routing

- A message is broken into smaller messages, packets
- Packets can take different paths
- Better error correction (smaller packets)
- But, each packet must carry routing, error correction, and sequencing information
- Suitable for long-haul communication networks (error rates can be higher)

We omit the cost calculation

Cut-through routing

- A message is broken into fixed-size units
  - flow-control digits or flits
  - 4 bits to 32 bytes
- A traces is sent to establish a connection
- Flits are sent one by one in order
- As soon as a flit is received it is forwarded to the next node
- All take the same route
  - no routing information with each flit
  - no sequencing information
  - less overhead for error detection
- Most parallel machines use this routing

Cost in cut-through networks

- Suppose a message of size m is broken into k pieces, each of size m/k
- Assume l links
- What is t_{comm}?

Cost in cut-through routing

- t_s startup time
- Header takes lt_h time
- It arrives after $t_s + lt_h$ time
- 1st flit arrives in time $t_w m/k$ after header
- 2nd flit arrives in time $t_w m/k$ after first
- ...
- kth flit arrives in time $t_w m/k$ after $(k-1)$st
- Total

$$t_{\text{comm}} = t_s + lt_h + t_w m$$

Implications

- $t_{\text{comm}} = t_s + l t_h + t_w m$
- t_s, t_w, and t_h are determined by hardware, software layers, and messaging semantics
- As programmers, we do not have control over them
- t_s is much larger than t_h and t_w
- Send larger messages versus many small ones
- Minimize the volume of data: better algorithms
- Minimize l: not much control by the user
  With MPI: little control over mapping processes to processors

Simplified model

- lt_h is usually dominated by t_s for small messages or $t_w m$ for large messages
- Maximum number of hops l is usually small in parallel machines
- We can simplify

$$t_{\text{comm}} = t_s + lt_h + t_w m$$

to

$$t_{\text{comm}} = t_s + t_w m$$

$$t_{\text{comm}} = t_s + t_w m$$

- This implies the same amount of time to communicate between any two nodes
- We can develop algorithms without worrying about the number of links