

Network Performance Model for TCP/IP Based Cluster Computing

Akihiro Nomura
Hiroya Matsuba
Yutaka Ishikawa

The University of Tokyo

Motivation



- Predicting performance of collective communication routine in MPI
 - $\log P$, $\log GP$, $P \log P$, ...
 - Effective in Infiniband and Myrinet
- Cluster uses Ethernet TCP/IP Network
 - Previous models are NOT effective
 1. Bottle-neck links in network switches
 2. Effect from retransmission control
- New performance model for commodity network is needed

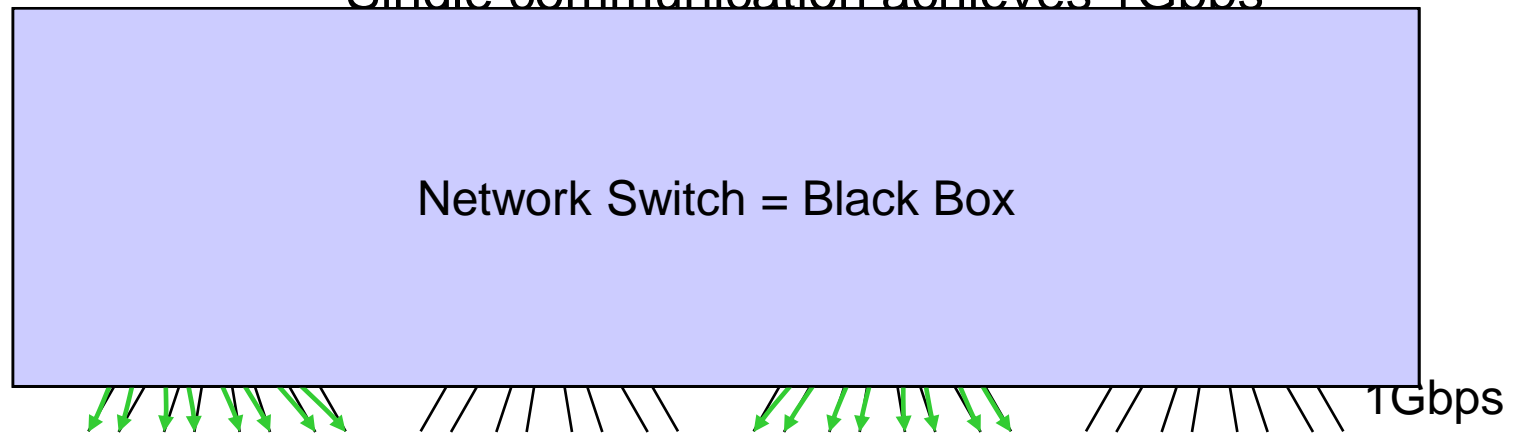
Issue 1

Bottleneck links in Network Switches

- Throughput is affected by other communications
 - Some ethernet switches does NOT guarantee full-bisection bandwidth

Ex:

Single communication achieves 1Gbps

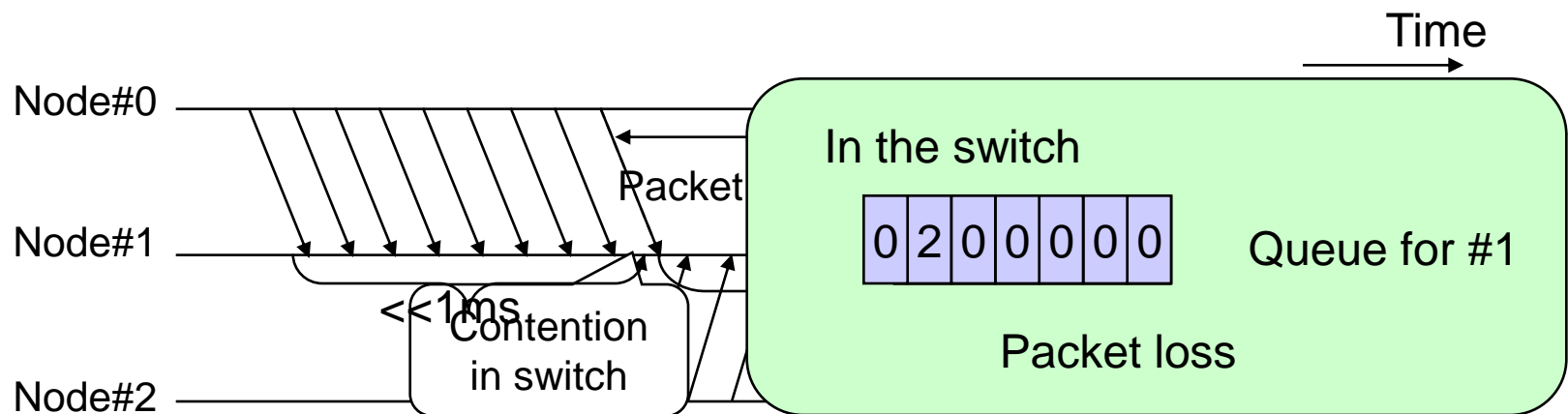


8 communications achieves only 2Gbps at once
→ 250Mbps per one communication

Issue 2

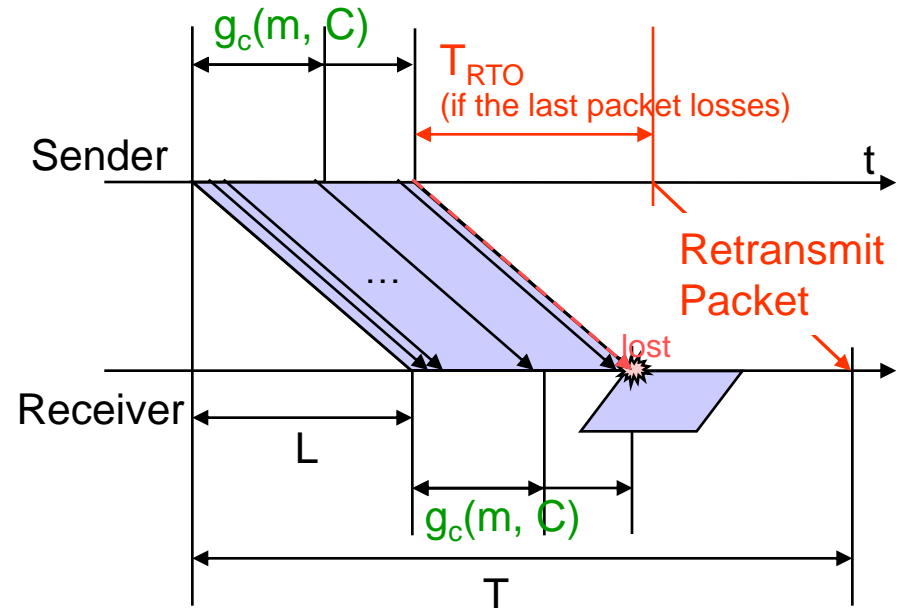
Retransmission Control

- TCP/IP uses acknowledge packets and retransmission to achieve reliability
- Fast retransmission [RFC 2001]
 - Not applicable for last packet of sequential communication
 - Requires next packets of same communication
 - Wait for expiration of RTO timer (200ms in Linux)
 - Last packet loss often happens in collective communication



PlogPT – Proposed Model

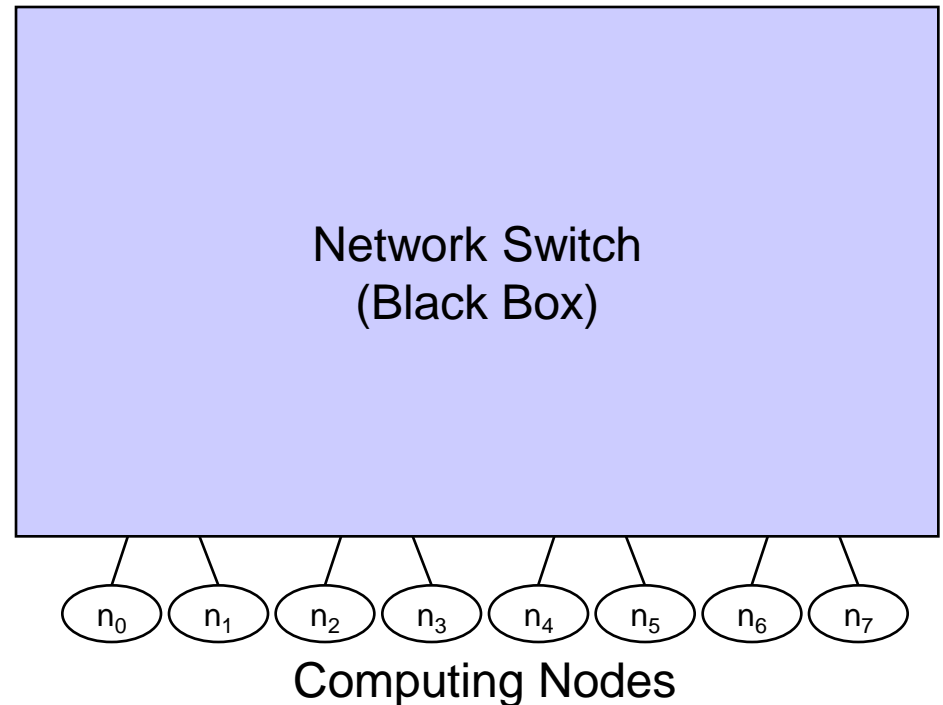
- Expresses
 1. Bottleneck links
 2. RTO Penalty
- Communication Time
 $= L + g_c(m, C) + T_{RTO}$
where
 - L: end-to-end Latency
 - $g_c(m, C)$: message Gap
 - T_{RTO} : RTO Penalty (If necessary)
- Topology aware gap
 - Message gap is affected by bottleneck links
- Treat RTO penalty
 - Condition for RTO is formulated



If the communication shares links
with other communications
If the last packet is lost by contention
in the network switch

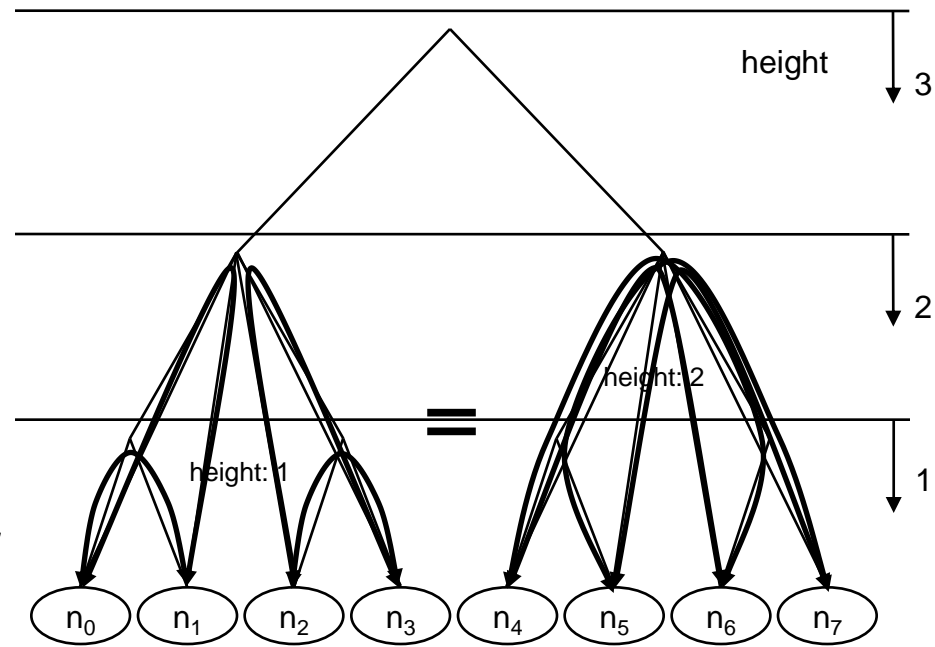
Binary Tree Network Model

- Network is mapped into symmetric binary tree
 - To calculate $g_c(m, C)$
 - To formularize the RTO condition
- Each edges have
 - b : bandwidth
 - d : delay
 - the same height edges have the same value
- Throughput is limited by bottleneck link
- Message gap $\propto (\text{throughput})^{-1}$



Binary Tree Network Model

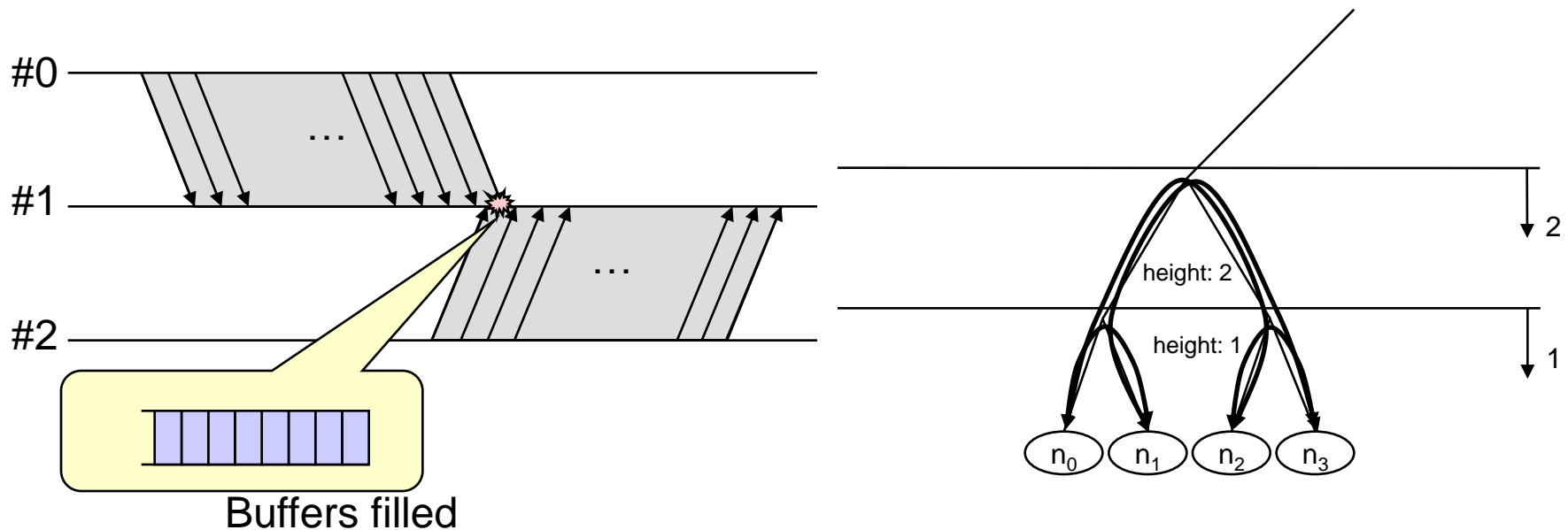
- The height i edge is negligible if
 - $b > 2b$ of height $i-1$ edge
 - $d = 0$
- Each communication has “height”
 - Minimum height of subtree contains the communication
- The height i communication is treated same as height $i-1$ communication if the height i edge is negligible
- The buffer size of switch is product of b and d



Each edges have
 b : Bandwidth
 d : delay

Estimation of RTO Occurrence

- RTO conditions (conjunction)
 1. A node receives the last packet of a message and the first packet of other message at the same time
 2. Message size > Buffer of switch
 3. Height of Communication Changes

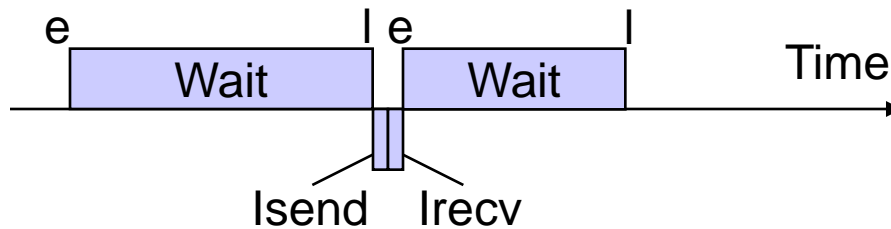


Estimation of Execution Time

- MPI routine is described by a sequence of simple primitives
 - `Isend()`, `Irecv()`, `Wait()` (not rendezvous, eager)
- Following values are defined for each primitives
 - e : the time when enters operation (start time)
 - l : the time when leaves operation (finish time)
 - f : the time when finishes communication
 - f_{noRTO} : the finish time if the RTO does not occur
 - These values are calculated by simulation
- Execution time of MPI routine is $\text{Max}(l)$

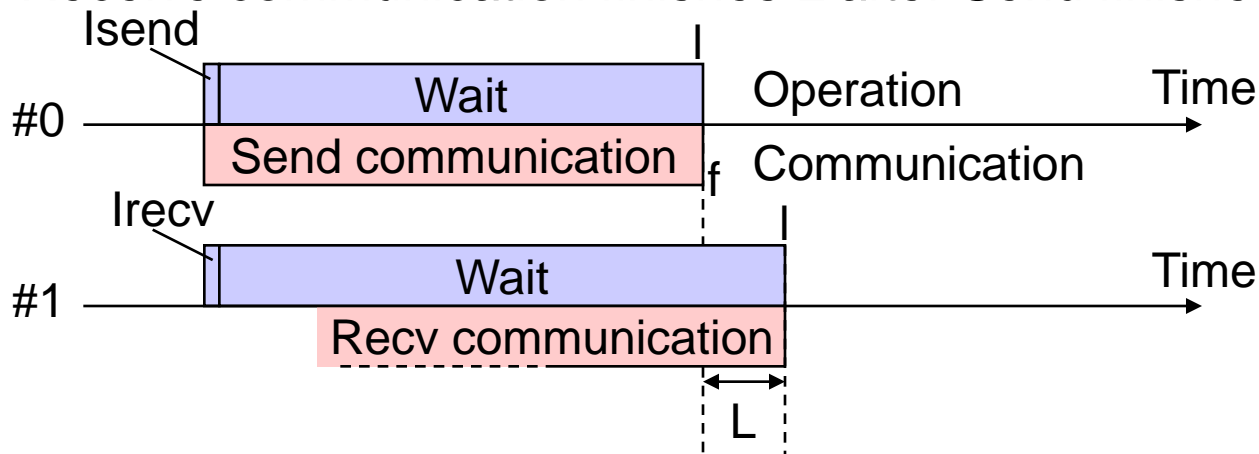
Constraints

- Enter time (e) is leave time (l) of previous operation
- Leave time (l) of Isend/Irecv is its enter time (e)



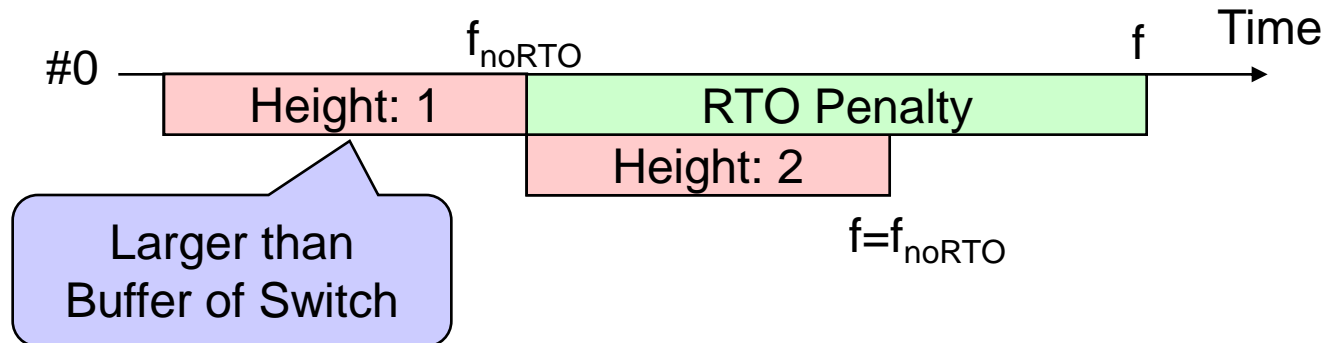
Execution time of whole MPI routine is $\text{Max}(l)$

- A node receives the last packet of a message and the first packet of other message at the same time
- Leave time (l) of Wait is corresponding Irecv's finish time (f)
- Receive communication finishes L after Send finishes



Constraints (contd.)

- f_{noRTO} is calculated by $e + (\text{message size} / \text{throughput})$
 - The message fills the network between e and f_{noRTO}
- RTO conditions
 - Communication finishes (f_{noRTO}) the same time the other communication starts (e) at the same destination
 - Message size > Buffer of switch
 - Height of Communication Changes



If these conditions are satisfied, $f = f_{noRTO} + T_{RTO}$
otherwise, $f = f_{noRTO}$

Evaluation of PlogPT model

- Compared two algorithms for MPI_Alltoall in both PlogPT and PlogP [Kielmann '00] models
- Evaluation Environment
 - AMD Opteron 2GHz Dual / 32 nodes
 - Ethernet 1Gbps (Broadcom Tigon 3)
 - BayStack5510-48T
 - Linux 2.6.11 SMP Kernel
 - YAMPI 1.0

Alltoall Algorithms

```

Alltoall_A#2(iP, P, t, int, m) {
  for (j = 0 to P - 1) {
    Irecv(1, j, x, i, m);
    Wait(Irecv);
    Irecv(2, 0, m);
  }
  Irecv(2, 0, m);
  Wait(Irecv);
  Wait(Irecv);
  Irecv(2, 1, m);
  Irecv(2, 1, m);
  Wait(Irecv);
  Wait(Irecv);
  ...
}

```

```

Alltoall_B#2(iP, P, t, int, m) {
  for (j = 0 to P - 1) {
    Irecv(2, j, x, i, m);
    Irecv(2, 3, m);
  }
  for (i = 1 to P - 1) {
    Irecv(2, j, x, i, m);
    Wait(Irecv);
    Irecv(2, 0, m);
  }
  Irecv(2, 0, m);
  Wait(Irecv);
  ...
  Wait(Irecv);
  ...
}

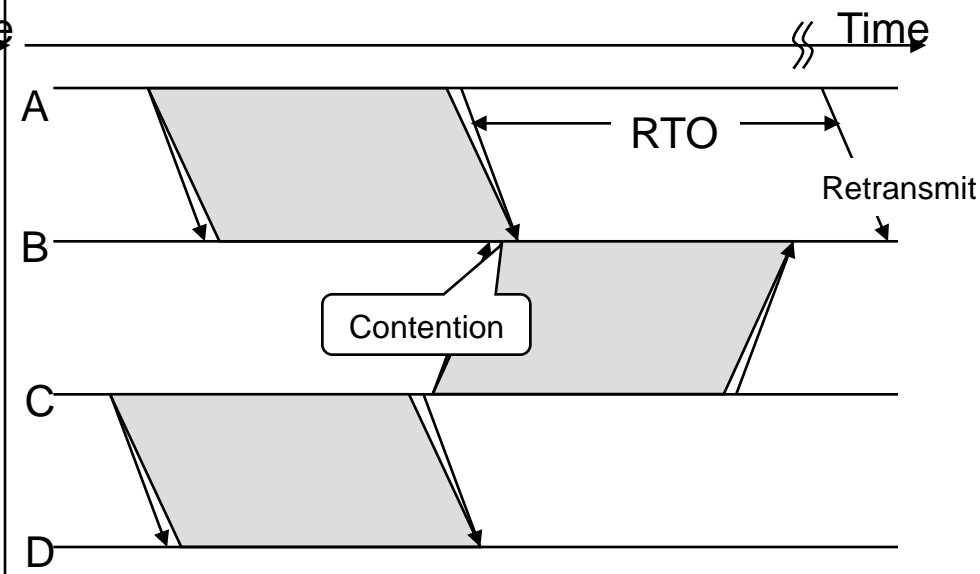
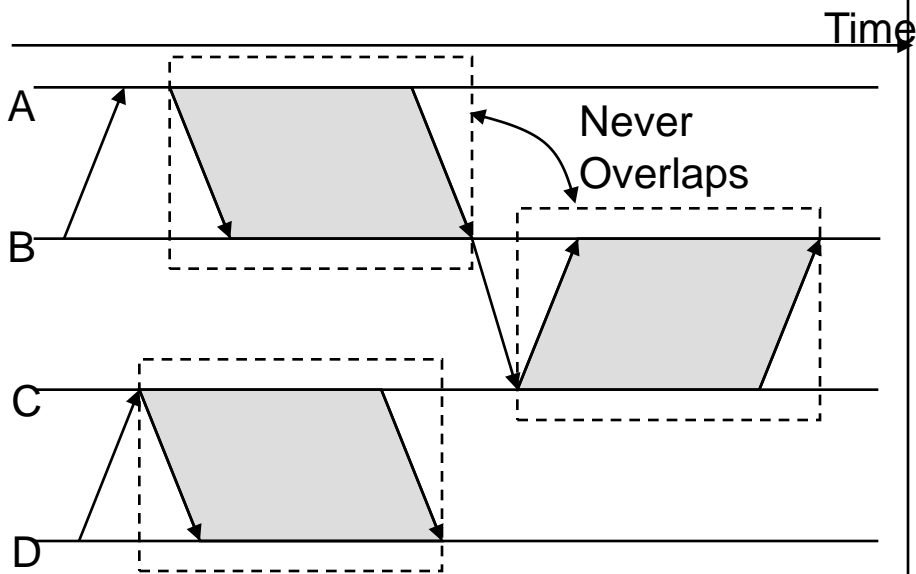
```

We will calculate e, l, f of each primitive using the constraints

Communication Timing

- A: Send w/ Synchronize
 - Slower
 - Low potential of contention

- B: Send w/o Synchronize
 - Faster
 - High possibility of contention



Outline of the Prediction

- In PlogP

- Not considering RTO

- B is always faster than A for synchronization loss

- In PlogPT

- Considering RTO

- B is faster if RTO does not occur
- B is slower than A if RTO occurs

- Considering slowdown by bottleneck links

- Slower than the PlogP model expects

RTO Conditions

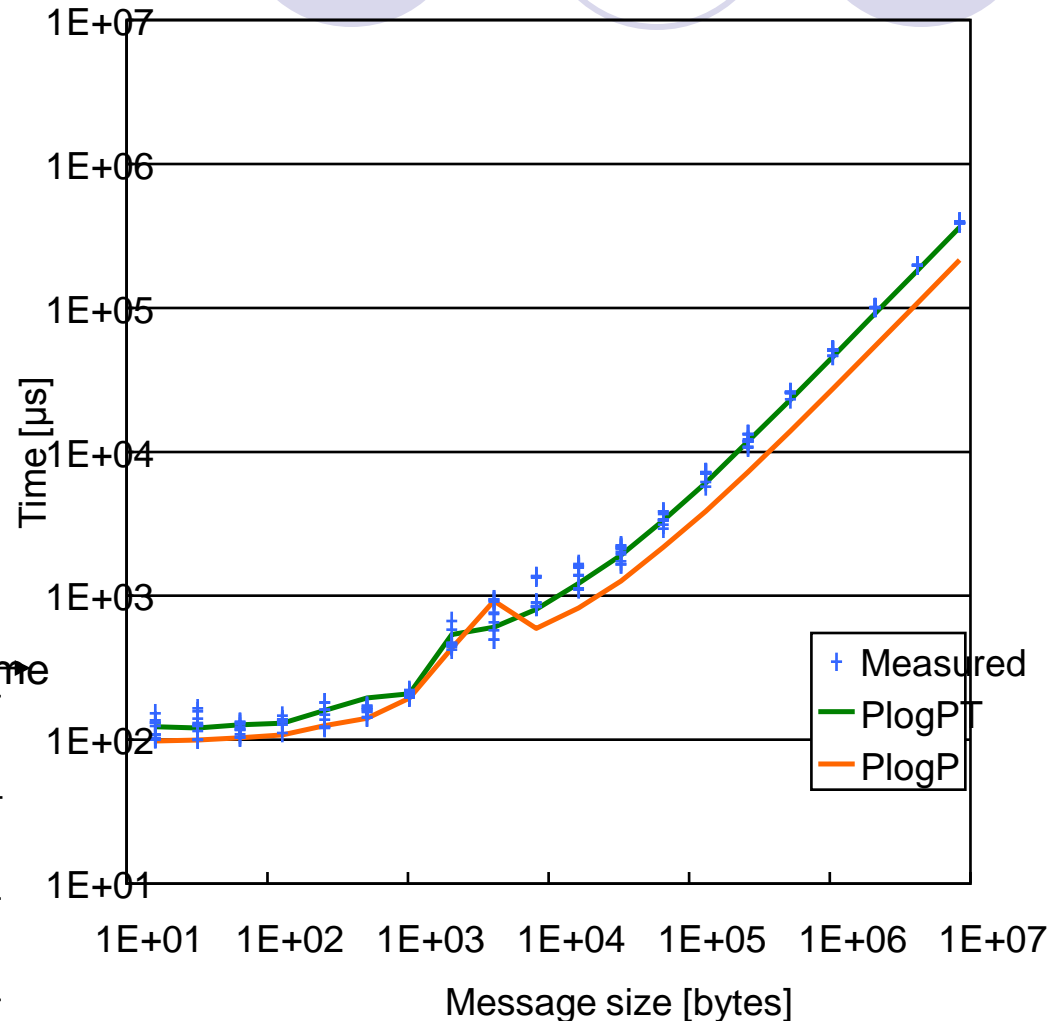
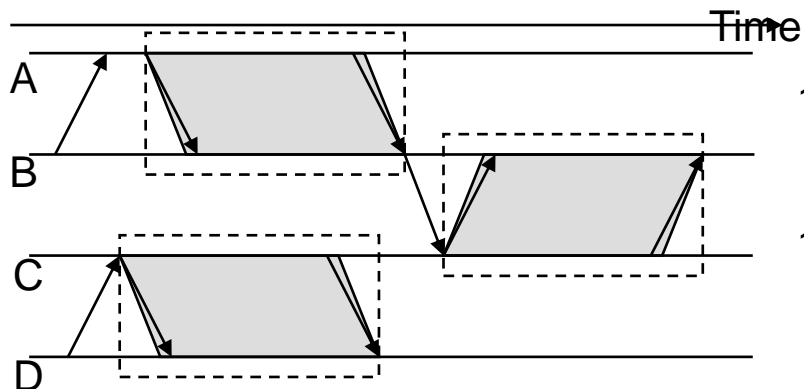


1. A node receives the last packet of a message and the first packet of other message at the same time
 - A: not satisfied because of synchronization
 - B: satisfied
2. Message size > Buffer of switch
 - If message size is > 4550 (in this case)
3. Height of Communication Changes
 - satisfied once

Prediction and Result - Algorithm A

```
Alltoall_A(int P, int r, int m) {  
  for (i = 0 to P - 1) {  
    MPI_Irecv(r, r xor i, m);  
    MPI_Send(r, r xor i, m);  
    Wait(Irecv);  
  }  
}
```

Synchronized Communication

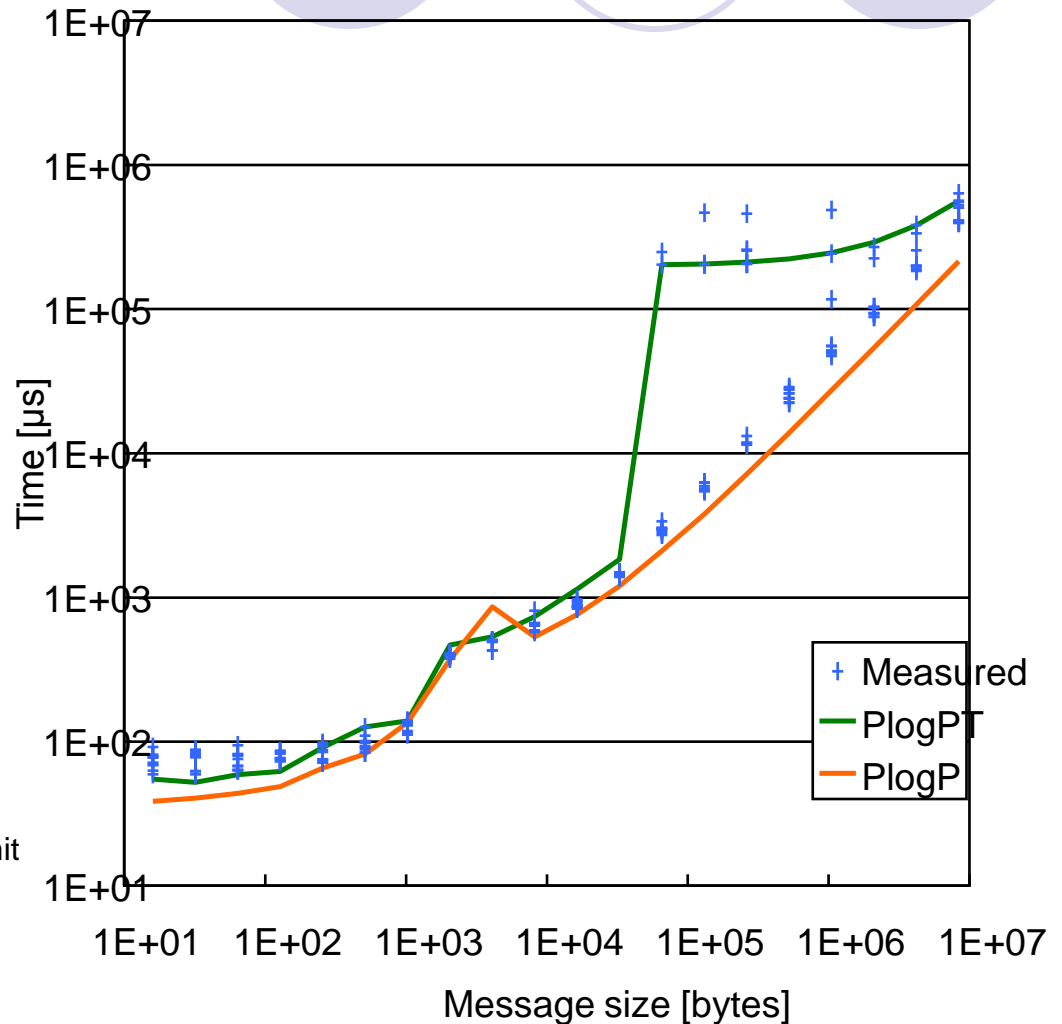
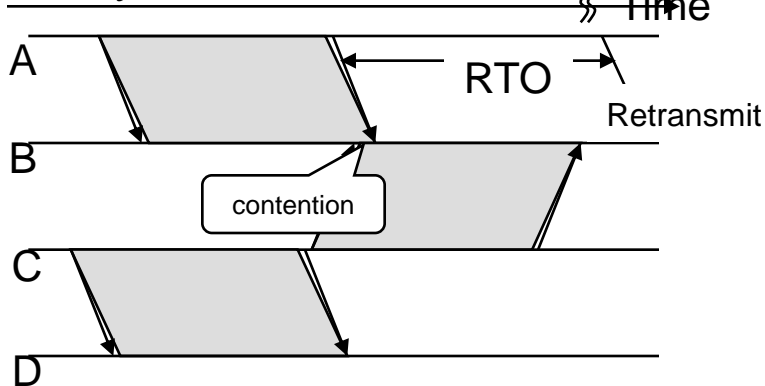


Prediction and Result - Algorithm B

```

Alltoall_B(int P, int r, int m) {
  for (i = 0 to P - 1) {
    if (i != r) MPI_Irecv(r, i, m);
  }
  for (i = 1 to P - 1) {
    MPI_Send(r, r xor i, m);
  }
  for (i = 0 to P - 1) {
    MPI_Wait(Irecv);
  }
}
    
```

Not Synchronized Communication



Related Work

A decorative graphic at the top of the slide consists of two rows of circles. The top row has three circles: a solid light purple circle on the left, a white circle with a light purple outline in the middle, and a solid light purple circle on the right. The bottom row has three circles: a solid light purple circle on the left, a white circle with a light purple outline in the middle, and a solid light purple circle on the right.

- logP [Culler '93]
 - Latency + Gap/Overhead per message
- logGP [Alexandrov '95]
 - Gap \propto Message size
- logPQ [Tohyama '98]
 - Divides the communication into multiple communication between buffers
- PlogP [Kielmann '00]
 - Gap = f(Message size)
- loGPC [Moritz '01]
 - Use both logP and logGP as Message size changes



Conclusion

- We have introduced new performance model: PlogPT
 - Based on binary tree network model
 - Expresses non-full-bandwidth network
 - Overcomes existing model's difficulty
 - Expresses slowdown in contention, especially RTO
- Future Work
 - Detailed condition of RTO occurrence
 - Establish stabilized parameter estimation method