Evaluation of inter-cluster data transfer on grid environment

Shoji Ogura[*1], Satoshi Matsuoka[*1,*2], Hidemoto Nakada[*3,*1]

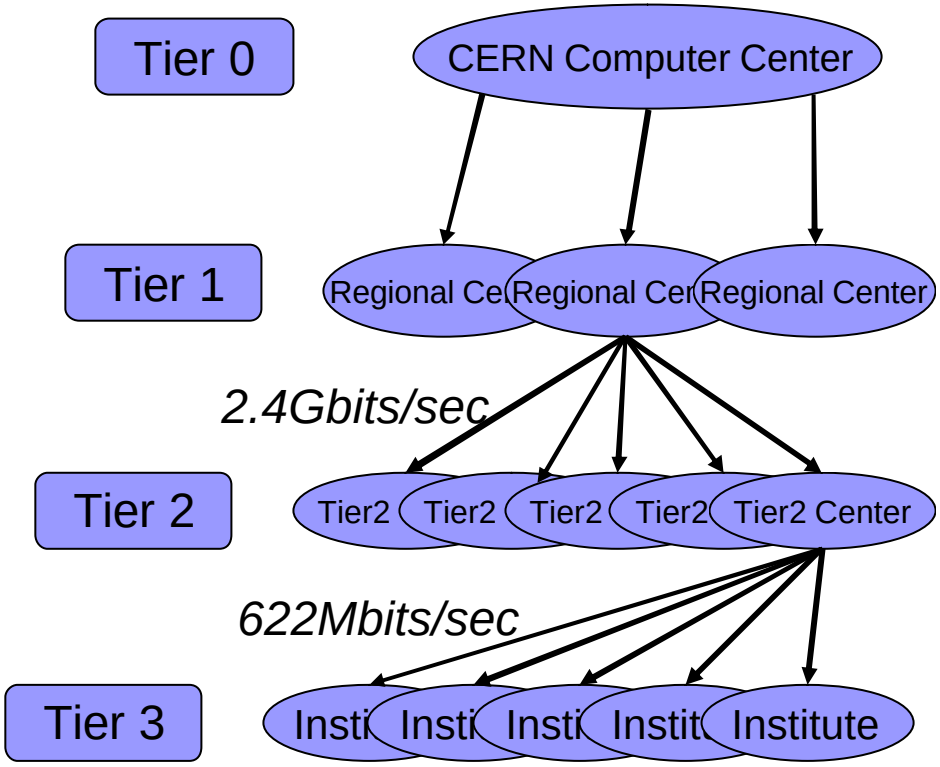
[*1]:Tokyo Institute of Technology

[*2]:National Institute of Informatics

[*3]:National Institute of Advanced Industrial Science and Technology

Background

- Large-scale cluster nodes are becoming mainstream platform for Grid computing
- CERN DataGrid Project
  - Construct a data processing fabric
  - Processing petascale data
  - Large scale transfer of massive data
- MPI jobs running across multiple clusters on the Grid
- Demands on inter-cluster data transfer



Inter-cluster data transfer

- No tools that optimize inter-cluster data transfer (or we don’t know)
- Can adapt the peer-to-peer method of optimizing data transfer to inter-cluster data transfer

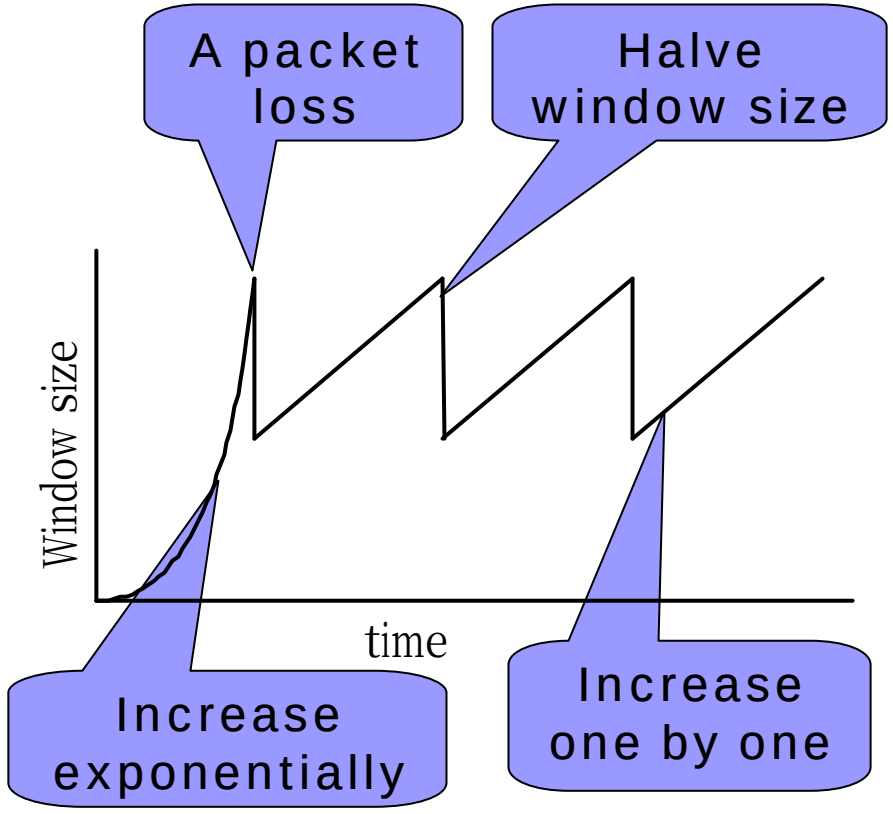
Overview

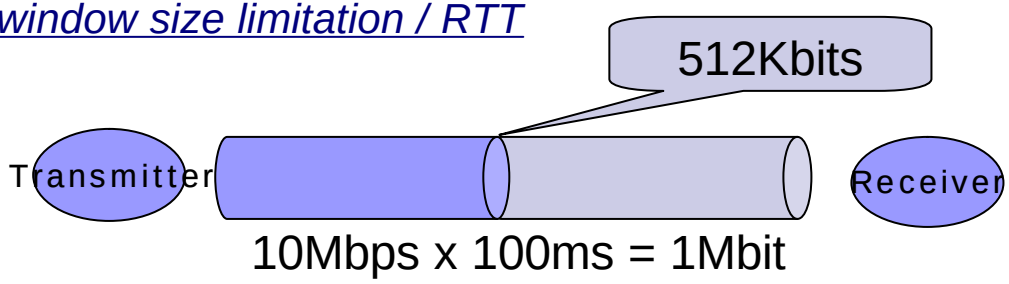
- Our goal
  - Propose an efficient transfer scheme when the peers are clusters
- Overview of this paper
  - Propose a method of coordinated transfer that determines parameters statically by simple model
  - Validate the model by simulating inter-cluster data transfer
    - The model is improved insufficient
  - Propose an efficient data transfer method that adjusts parameters dynamically

TCP's problems on Grid environment

- Behavior of TCP window size
  - Slow start phase
    - When window size is minimum value
    - Increase of TCP window size exponentially until a packet loss occurs
  - Congestion avoidance phase
    - Increase of TCP window size one by one
    - TCP window size is halved when a packet loss occurs
- Upper limit of TCP window size
  - Limited to a constant value (in the early TCP, the value is 64KB)

bandwidth ＜ the window size limitation / RTT



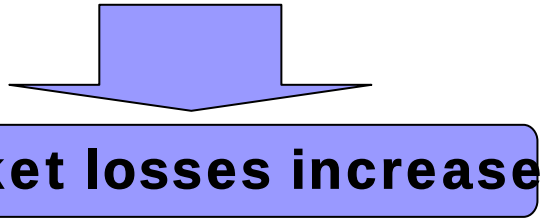


Existing methods of exploiting TCP bandwidth of peer-to-peer transfer

- Window size tuning
  - Change the upper limit of TCP's window size
  - For extremely high-bandwidth networks, the TCP window size becomes substantial
  - To recover the appropriate window size will take considerable amount of time because of packet losses
- Using UDP
  - Compensate for the lost packets later
  - Devastate most competing TCP transmissions
  - SABUL, FOBS, Tsunami, …
- Network Striping
  - Transfer data in parallel, using multiple sockets
  - PSockets, GridFTP, …

Adapting P2P data transfer methods to cluster-to-cluster data transfer

- All peers attempt to optimize the transfer individually
- TCP's window size may become excessively large
- The network pipe may be stuffed with excessive packets



- P2P data transfer optimization methods may be insufficient

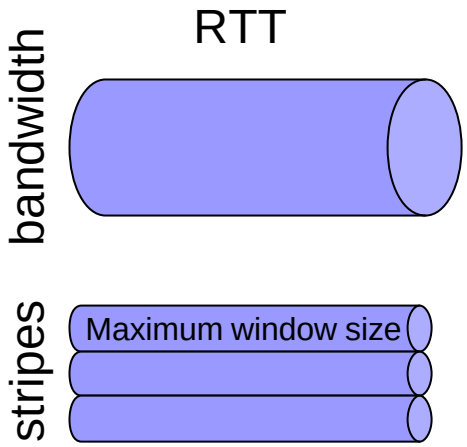
Coordinated striping for peer-to-peer transfer between clusters

- Coordinated transfers are needed
- Use Network Striping
  - Able to develop at the application level
  - Less greedy than Using UDP

- Propose the use of striping properly coordinated across the cluster nodes
- Method of determining the number of stripes

  - Determine the parameters dynamically

Simple modeling of optimal parameters

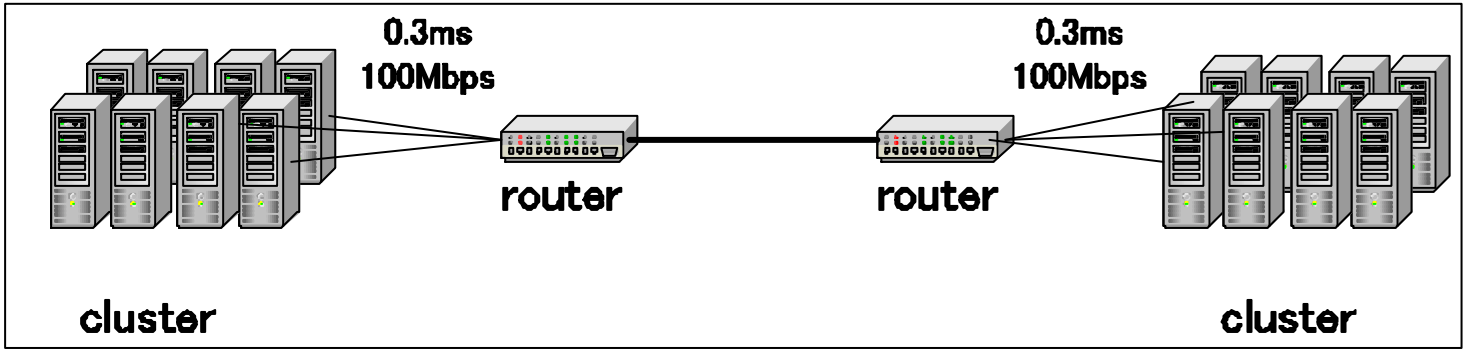


- Simple modeling
  - Available data on the link
    - RTT x Bandwidth
  - The amount of data transferred
    - The number of stripes x Maximum window size
  - Optimal number of stripes
    - (RTT x Bandwidth) / Maximum window size
- The formula denotes that the optimal number of stripes is proportional to RTT

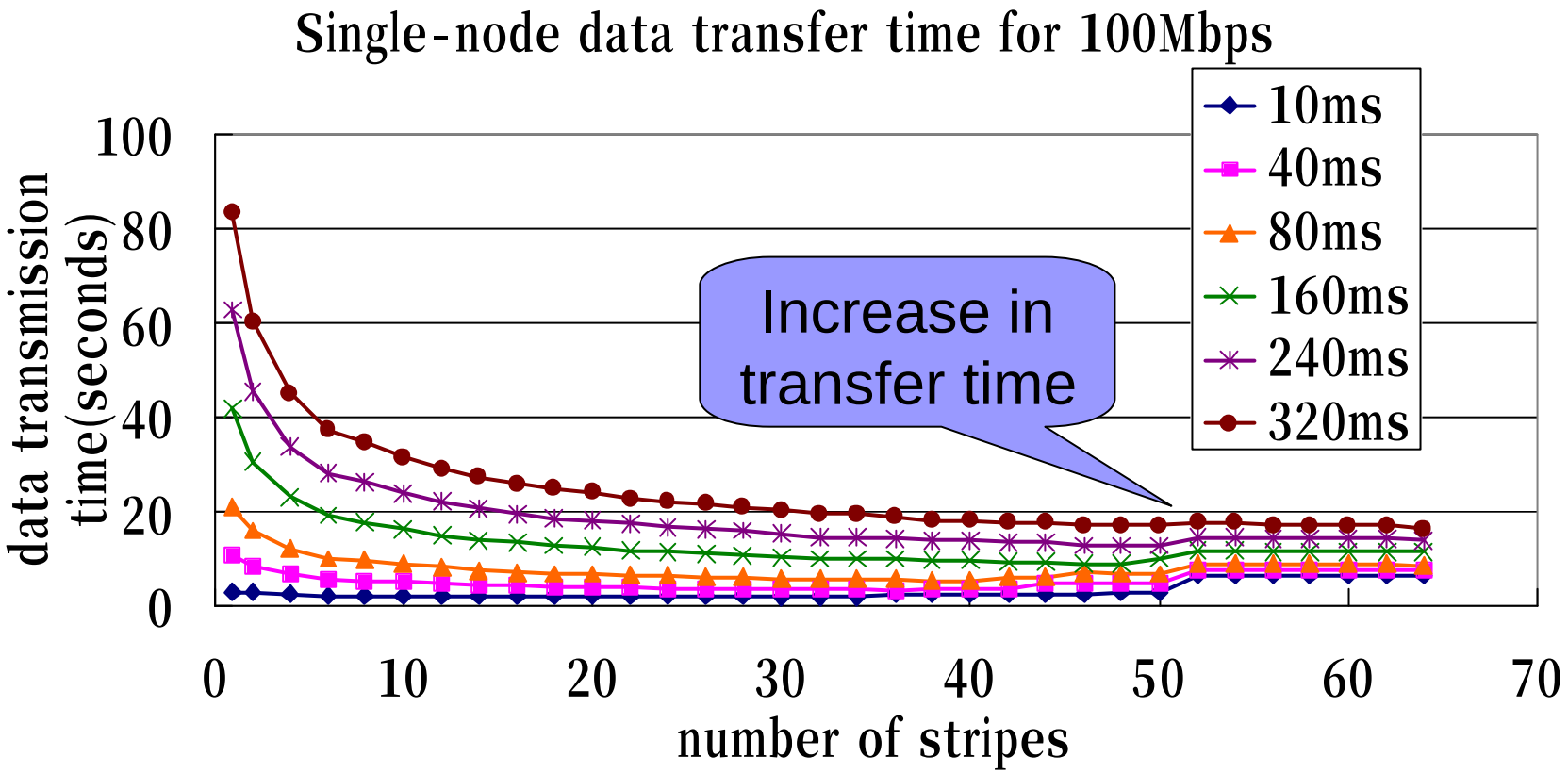
Confirm the validity of this simple modeling by simulations

Simulating data transfer between clusters on the Grid

- NS (Network Simulator)ver.2.1b8
  - A discrete event simulator dedicated to simulating various network behaviors
  - Providing various protocols for different networking layers
    - Low-level unicast/multicast protocols
    - Standard TCP protocols
    - HTTP,FTP,...
- Our settings
  - The sizes of transmitted data are identical among the nodes(8MBytes)
  - The data on each node is striped to be the same size
  - The number of nodes are varied 1～16, stripes 1～64
  - Between the node to router: latency 0.3ms, bandwidth 100Mbps
  - Between the routers: latency 10ms～320ms, bandwidth 1.5Mbps～1Gbps
    - When the bandwidths of the network link between routers are 100Mbps, 1Gbps, the bottleneck link is between the node to router

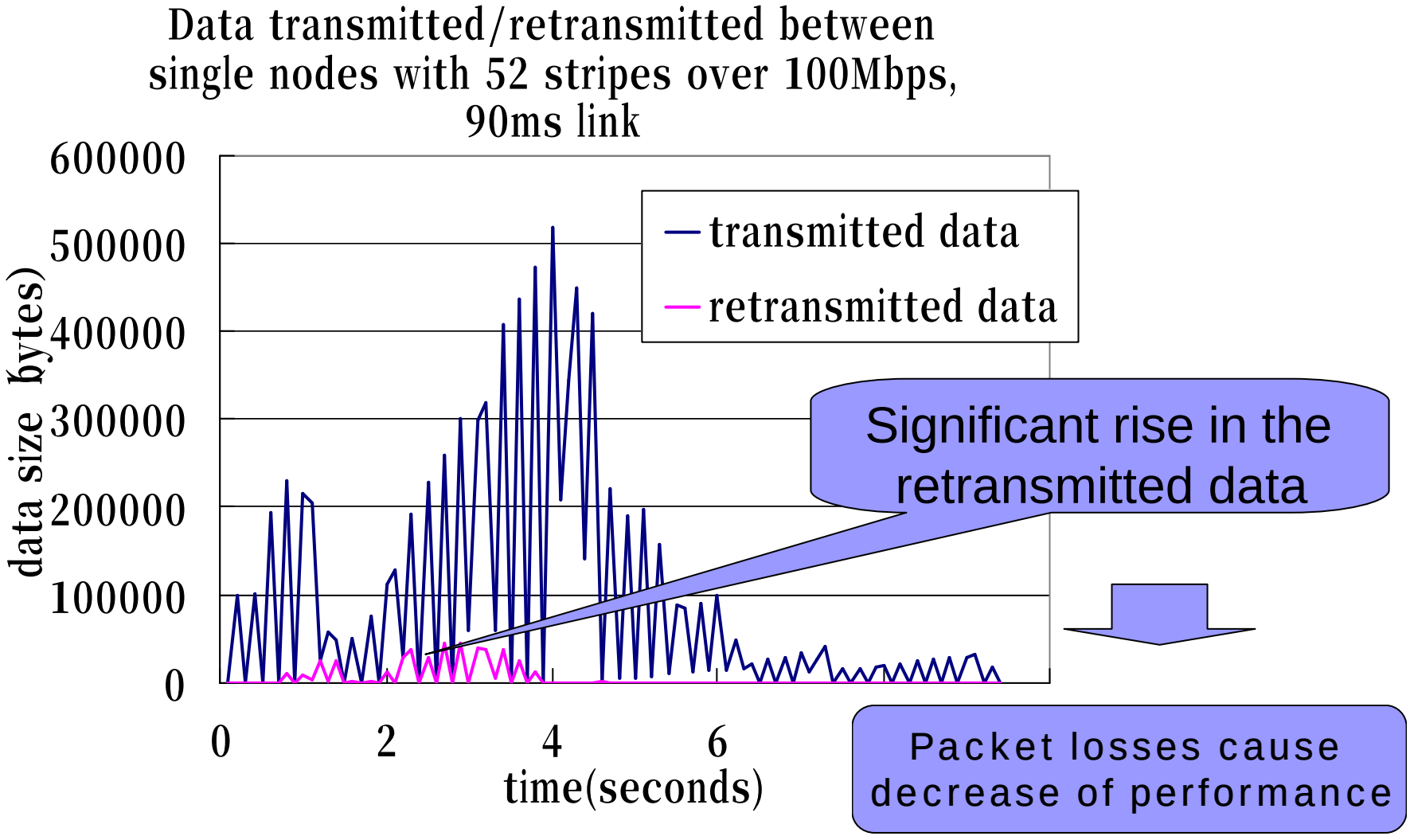


Single-Node Transfers

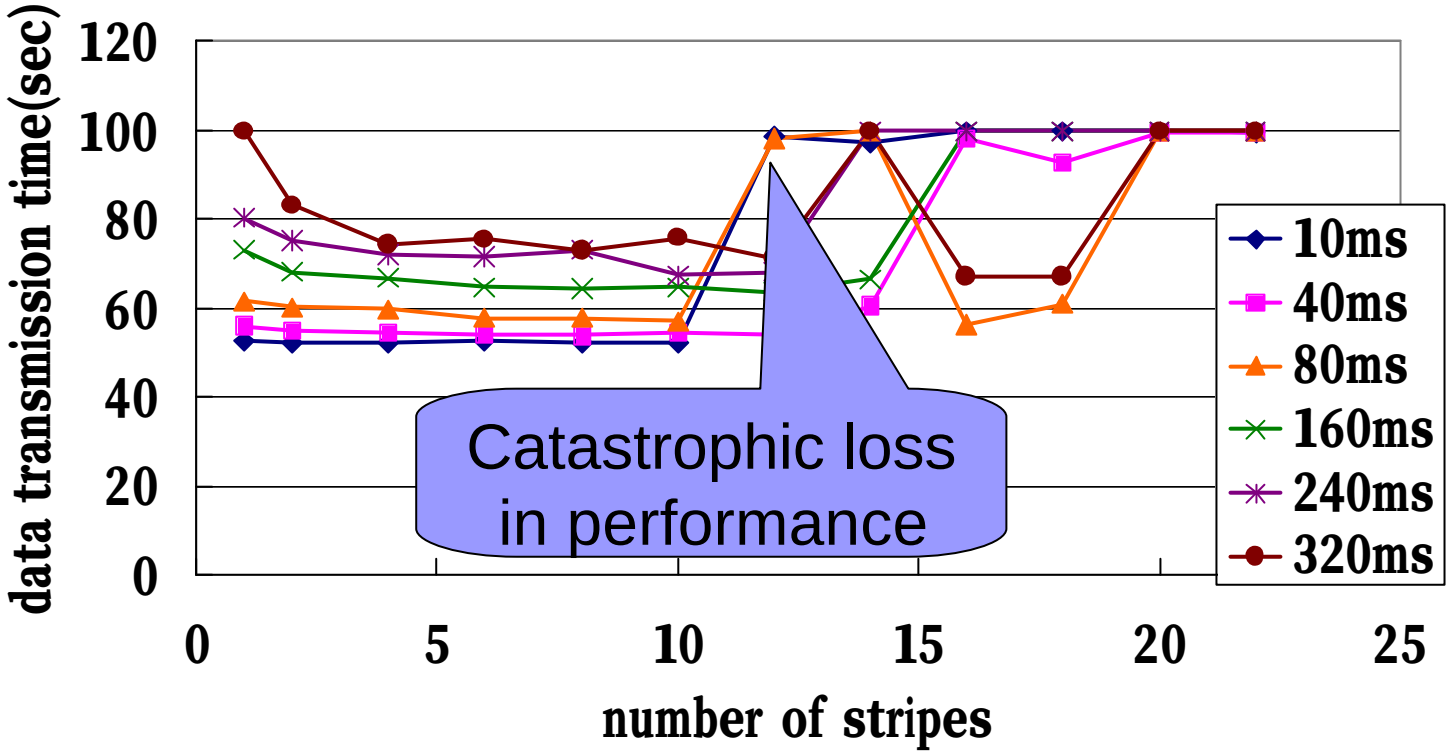


- The transfer time initially decreases as we increase the number of stripes
- Increase in transfer time at some point (approximately 50 stripes)
- But these changes are slight

The amount of data transmitted versus amount of data retransmitted

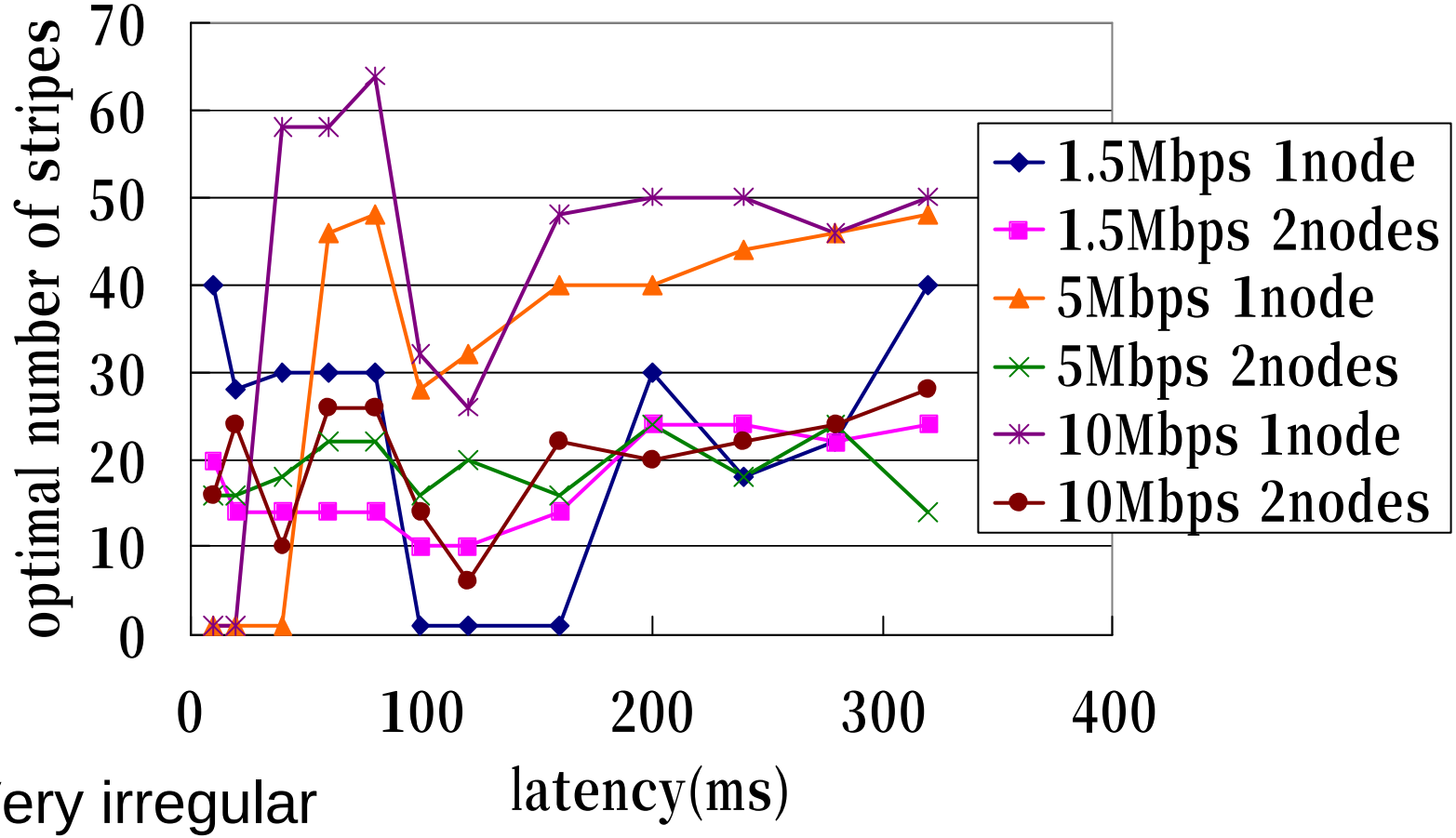


Multiple node transfers over low bandwidth link（10Mbps, 8nodes）



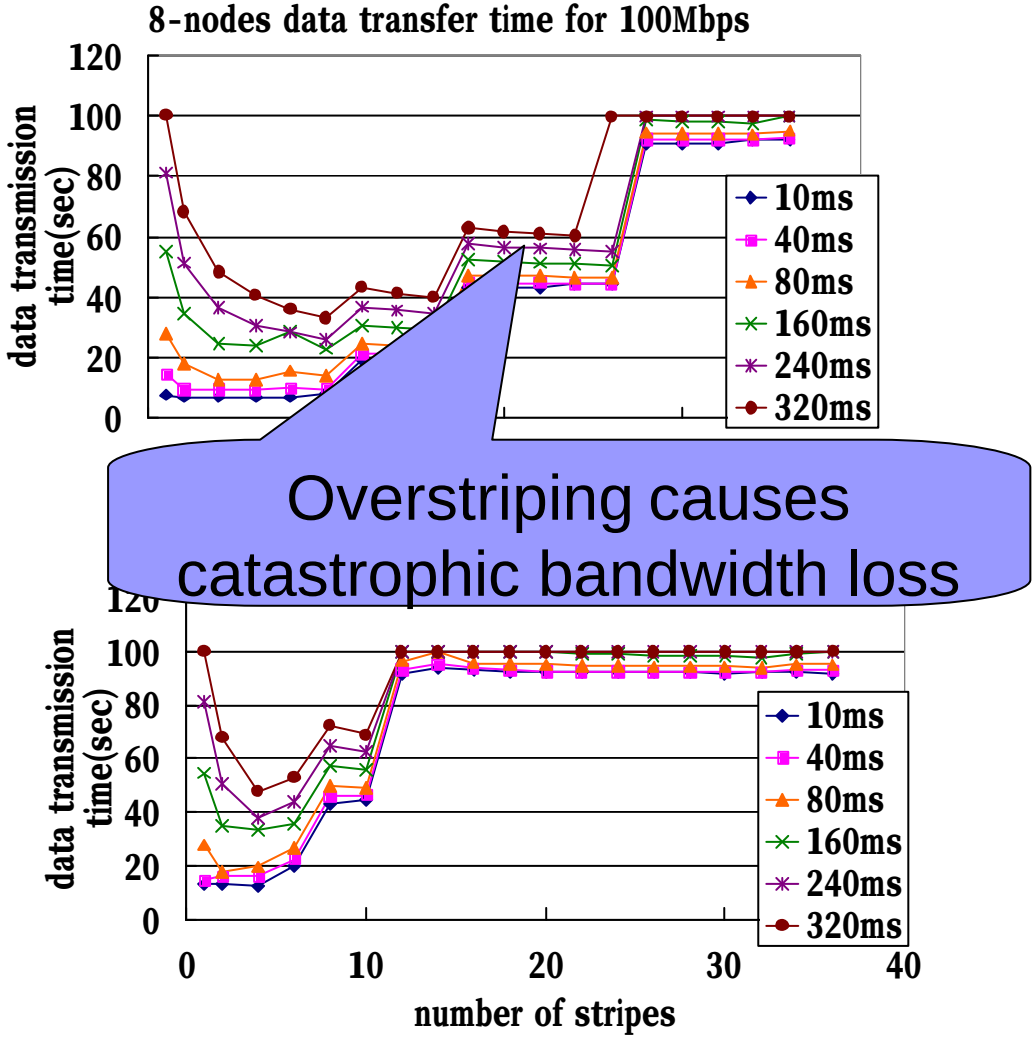
- Even with striping, uncoordinated transfer that would saturate the network will have unacceptable performance due to explosive increase in packet loss

Optimal number of stripes over low bandwidth link



- Very irregular
- Difficult to determine the trend given the network bandwidth/delay and number of nodes

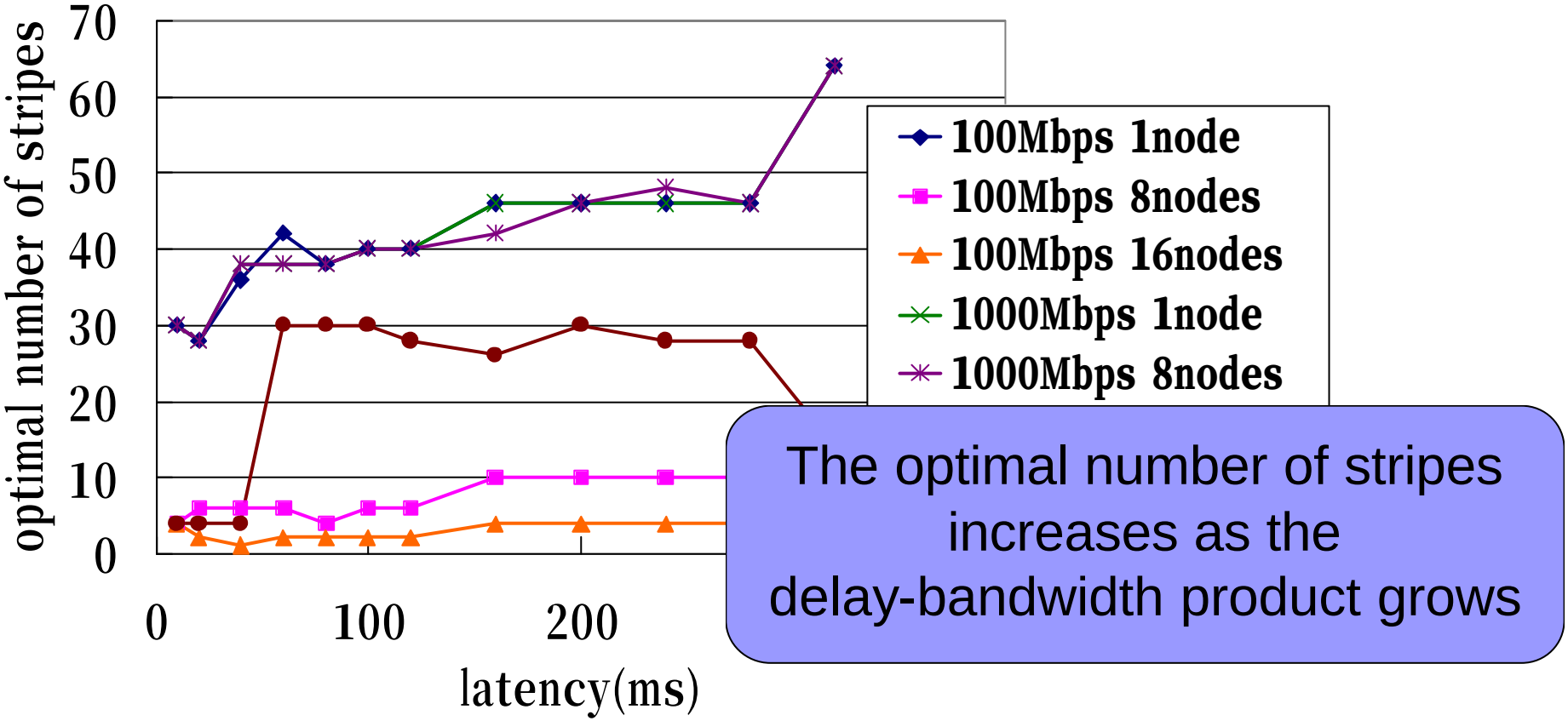
Multiple node transfers over high bandwidth link (100Mbps, 8/16nodes)



- Data transmission time at 16nodes data transfers is almost same to one at 8nodes, twice number of stripes

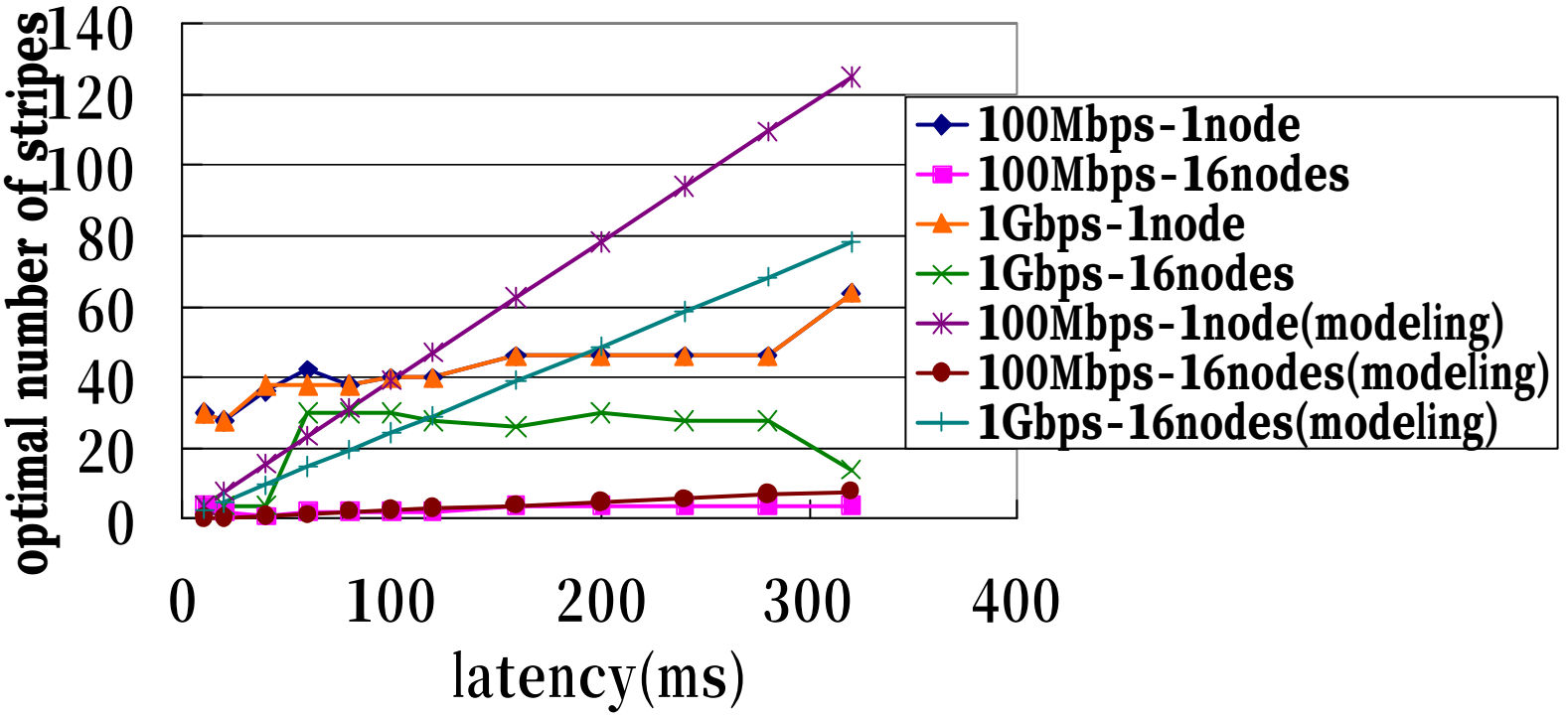
⇒When the bottleneck resides in the network between routers, the product of optimal number of stripes and nodes may be constant

Optimal number of stripes over high bandwidth links (100M/1Gbps, 1/8/16nodes)



- In 100Mbps-1node、1000Mbps-1node、1000Mbps-8node graphs, the bottleneck of the link is the same (between node – router links)

Comparison with simple modeling



- The formula denotes that the optimal number of stripes is proportional to latency, but is not so in the simulation

Proposal for dynamic parameter tuning

- Difficult to apply the simple model above to determining the optimal number of stripes
- Need to determine the parameters dynamically
  - Adjust the maximum stripe size dynamically
  - A coordinated fashion across the nodes

Dynamic parameter tuning

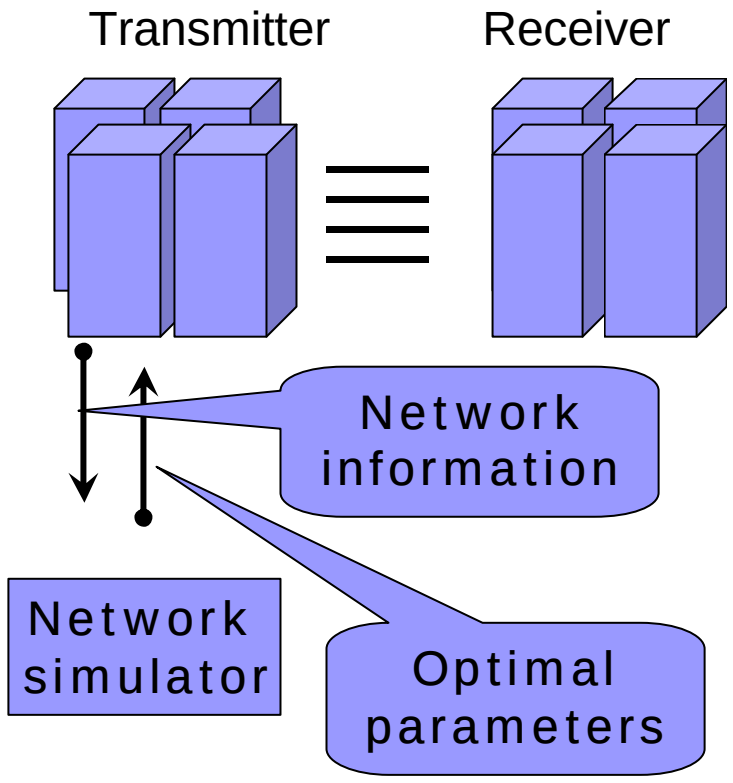
- Previously opening multiple sockets
- How to determine the parameters
  1. Using observable dynamic values
  2. By running simulations during transfer
  3. Using periodic observable dynamic values

1. Using observable dynamic values

- Tune the parameters such as the number of nodes or stripes dynamically during transfer, using observable dynamic values of the transfer (e.g., packet loss ratio)
- Apply an extended and corrected model
  - Using dynamic parameters

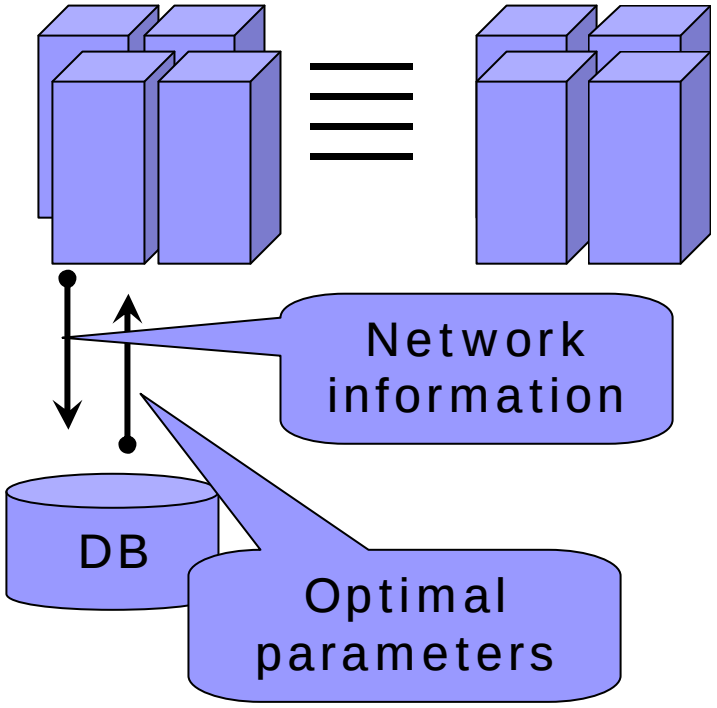
2. By running simulations during transfer

- Tune the parameters such as the number of nodes dynamically by running simulations
- More accurate and faster compliance with the optimal setting
- × difficult to manage



3. Using periodic observable dynamic values

- Determine the parameters
  - Using periodic observable dynamic values of the transfer
  - Searching the database that indicates the optimal values
- The database may be constructed from real observation or from simulation
- ○ accuracy and lightweight
- × difficult to construct the database



Conclusion

- Propose a method of coordinated transfer that determines parameters statically by simple model
- Validate the model by simulating inter-cluster data transfer
  - The model is improved insufficient
- Propose an efficient data transfer method that adjusts parameters dynamically

Future work

- Conduct experiments on real environment to verify the validity of our simulation
- Develop a coordinated data transfer tool that tunes a parameters dynamically

end