High Performance Wide-area Overlay using Deadlock-free Routing

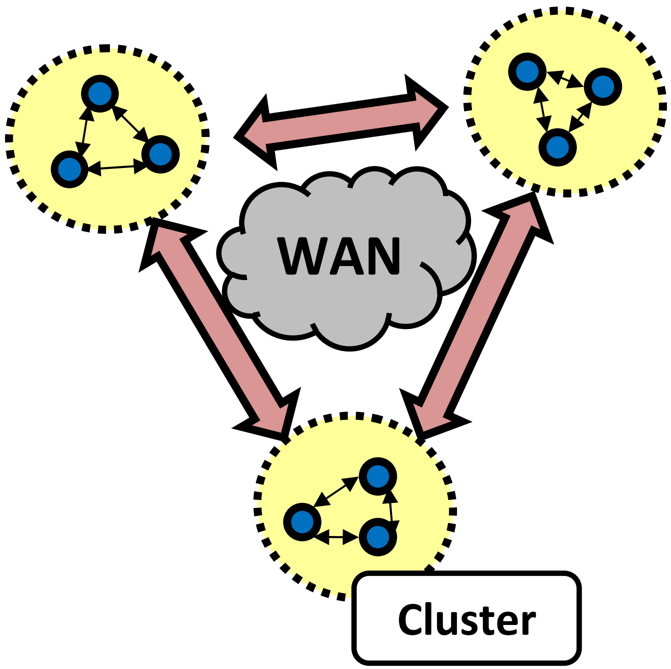
Ken Hironaka, Hideo Saito,
Kenjiro Taura
The University of Tokyo
June 12th, 2009

Parallel and distributed computing in WANs

- Grid environments have become popular platforms
  - Grid5000, DAS-3, InTrigger
  - Helped by greater WAN bandwidth

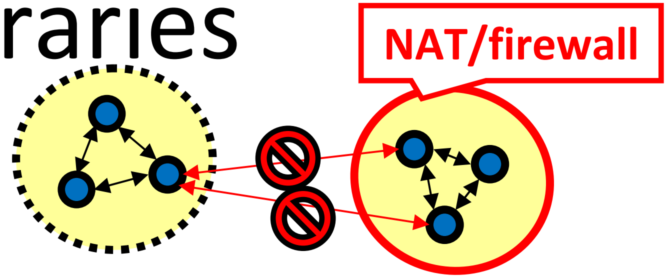
- Communication is getting increasingly important
  - CPU-intensive applications
    - More communication intensive
    - e.g.: Model-checking
  - Data-intensive applications

- The design/implementation of communication libraries is crucial

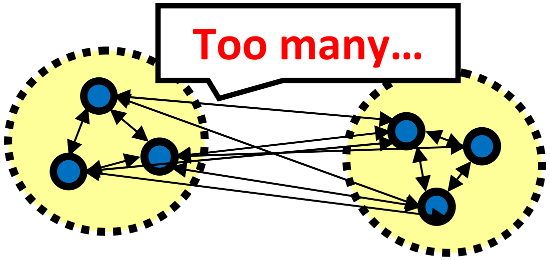


Application overlays for communication libraries

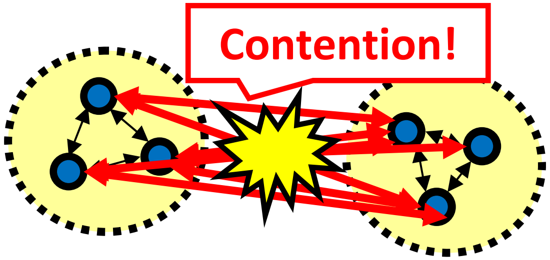
- **WAN connectivity**
  - NAT, Firewall
  - SmartSockets [Maassen et al. '07]



- **Scalability**
  - Few connections as possible
  - Host main memory constraints
  - Stateful firewall session constraints



- **High performance**
  - Avoid network contention at bottlenecks



WAN overlay Requirements for parallel and distributed computing

- Low Transfer/routing overhead
  - Overlay performance ***does*** matter
  - Not only latency, but also bandwidth

- **“safe”** overlays
  - ***NO*** Memory overflow
  - ***NO*** Communication deadlocks

Our contribution

- Overlay for effective parallel and distributed computing on WANs
  - Low transfer/routing overhead
  - No memory overflow/deadlocks
  - Efficient deadlock-free routing for heterogeneous networks
- Experiment on a large scale WAN environment
  - 4 to 7 clusters：up to 290 nodes
  - Higher performance for collective communication

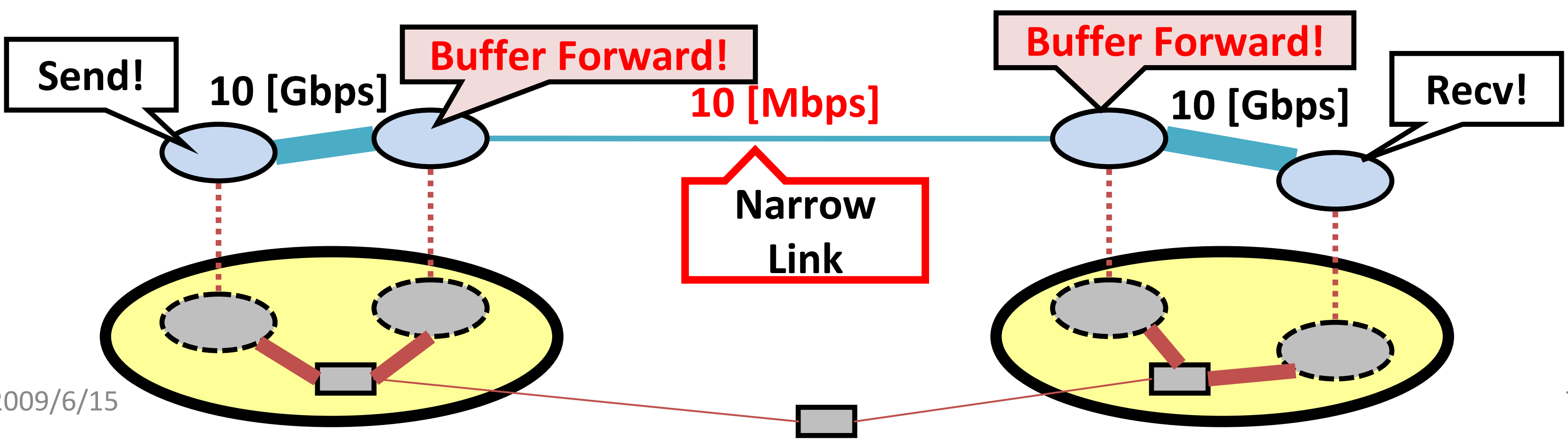
Problem Setting

- Introduction
- Problem Setting
- Related Work
- Proposal
- Evaluation
- Conclusion

Description of our overlay setting

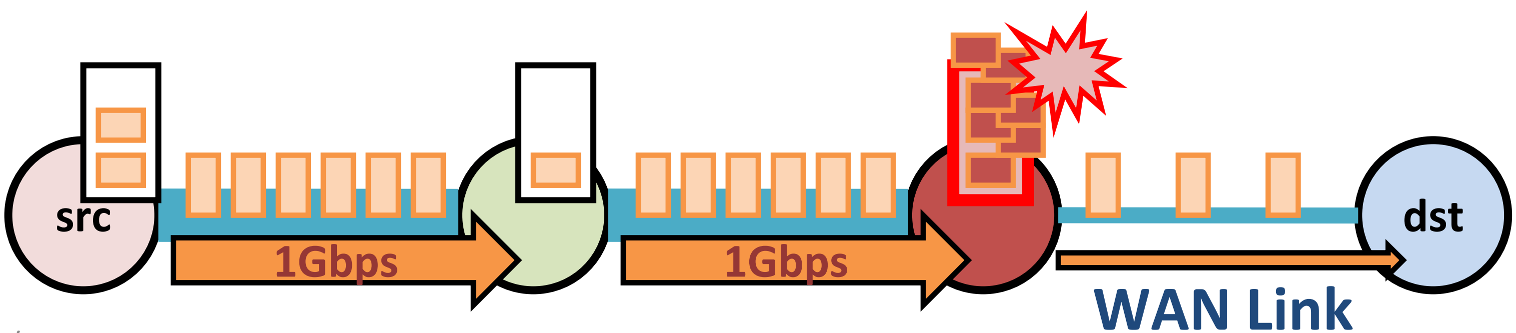
- Multi-cluster environment (LAN + WAN)
- Latency and Bandwidth are ***heterogeneous***
  - 100[us] ~ 100[ms]
  - 10 [Mbps] ~ 10[Gbps]

- Heavy stress on forwarding nodes that buffer packets
- A naïve implementation will have 2 outcomes
  - **Memory overflow** in intermediate nodes
  - **communication deadlock** among nodes



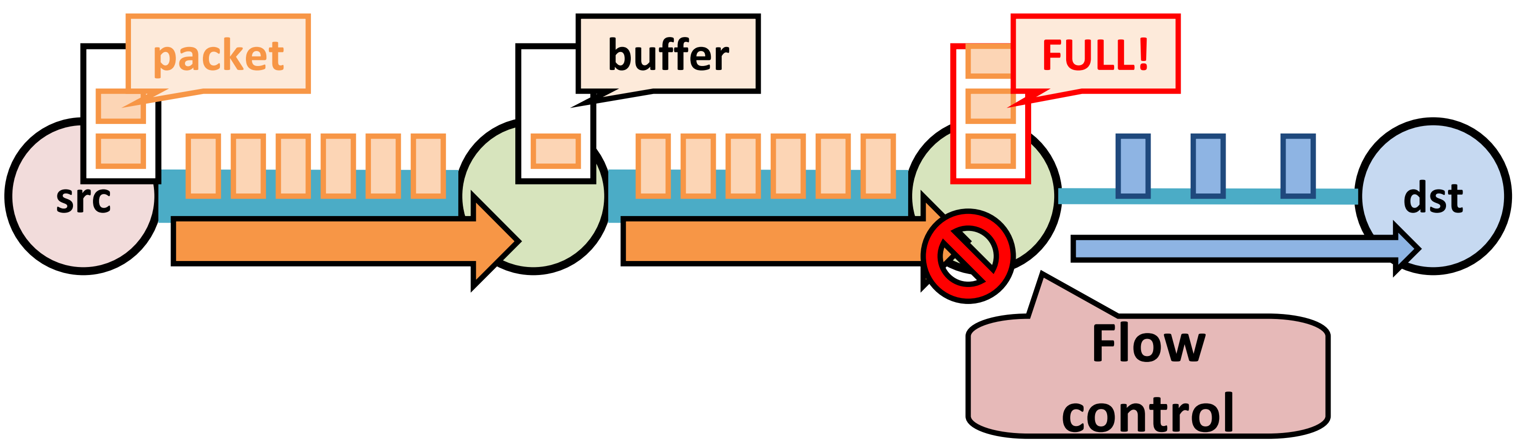
No. 1: Memory overflow

- When a node buffers packets *without regards* to its free buffer size
- Thus, all nodes must use ***fixed size*** buffers



No. 2: Deadlocks with naïve flow control

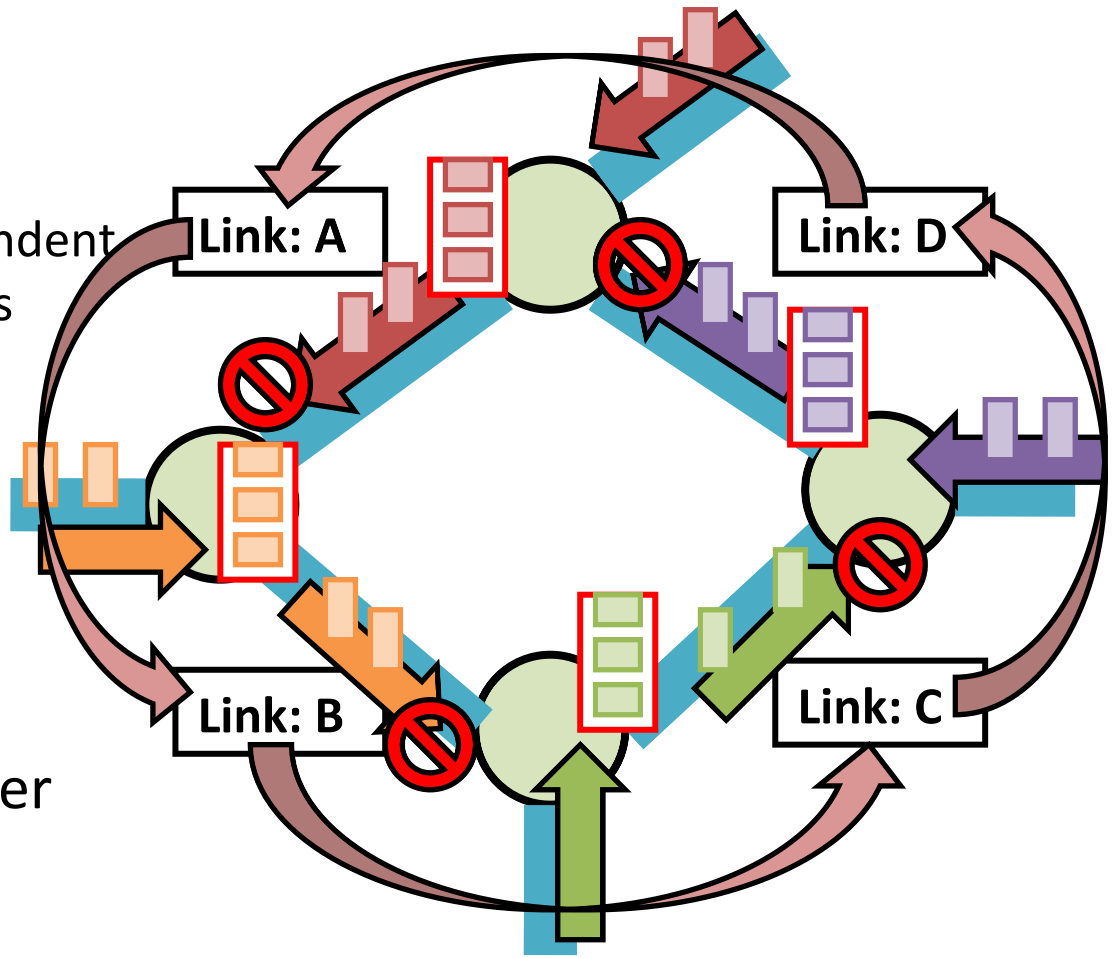
- Simple flow control solution
  - stop receiving once your buffer is full
  - Feeds back to the sender to tune the send rate
  - Possibility of a deadlock



A Deadlock Example

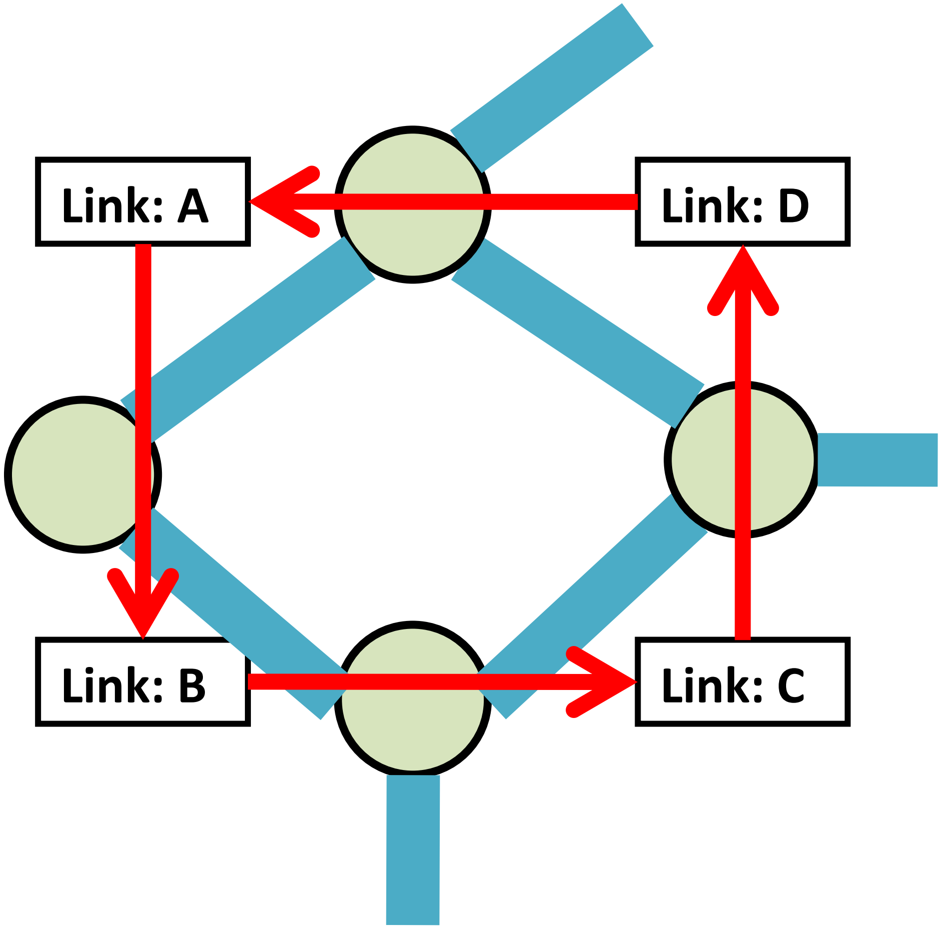
- When multiple transfers coexist ：
  - Transfers will become dependent each other to make progress
- 4 transfer example
  - Link A → Link B
  - Link B → Link C
  - Link C → Link D
  - Link D → Link A
- Transfers wait on each other

  ⇒**deadlock**



The source of the deadlock

- Cycle in the link dependency graph
- **Deadlock free routing is necessary**
  - ***Restrict*** routing paths so that deadlocks cannot occur
- They ***Do*** Happen!
  - 40-node LAN, 1 M buffers
  - 0.1 connection density graph
  - 5MB all-all operation



Additionally…

- Existing deadlock-free routing algorithms ***do not*** account for the underlying network
- In WANs, we must use underlying network information for efficient routing

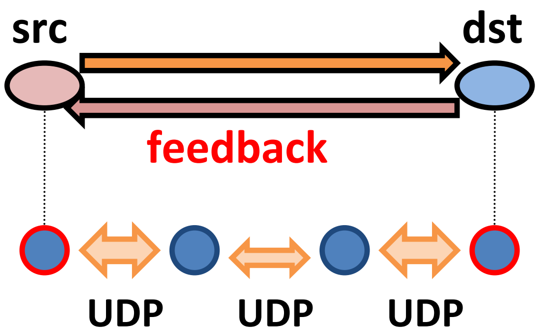
Related Work

- Introduction
- Problem Setting
- Related Work
- Proposal
- Evaluation
- Conclusion

Existing WAN overlays

- Do not consider problems like buffer overflow and deadlocks
- RON (Resilient Overlay Network) [Andersen et al. '01]
  - UDP overlay network among all nodes
  - UDP interface to the user
    - **Communication reliability and flow control are left to the user**
- DiskRouter [Kola et al. '03]
  - File transfer overlay
  - When buffer usage reach a threshold, stop receiving
    - **Possibility of deadlocks**

Flow Control for Overlays



- **UDP Overlay + End-End Flow-control**
- [Kar et al. ’01]
  - ACK on every packet sent
  - ACKs piggyback link congestion information
- Spines : [Amir et al. ‘02]
  - Link congestion information is shared periodically
  - Sender tunes its rate based on congestion of its path
- **Pros:**
  - **Eliminate burden on forwarding nodes**
- **Cons:**
  - **Isn’t this re-implementing TCP?**
  - **A lot of parameter tuning**
  - **hard to yield maximum bandwidth on path: (30 % utilization)**

Our work: Use TCP + flow-control at forwarding nodes + deadlock free routing

Deadlock-free Routing

- Restrict routing paths to prevent deadlocks
  - **Not suitable for WANs**
- Algorithms for parallel computer interconnects:
  - Assume “regular” topologies
  - [Antonio et al. ’94]
  - [Dally et al. ’87, ‘93]
- Algorithms for general graphs ：
  - Do not account for underlying network
  - Constructed paths are suboptimal
  - Up/Down Routing [Schroeder et al. ‘91]
  - Ordered-link Routing [Chiu et al. ‘02]
  - L-Turn Routing [Koibuchi et al. ‘01]

Proposal

- Introduction
- Problem Setting
- Related Work
- Proposal
- Evaluation
- Conclusion

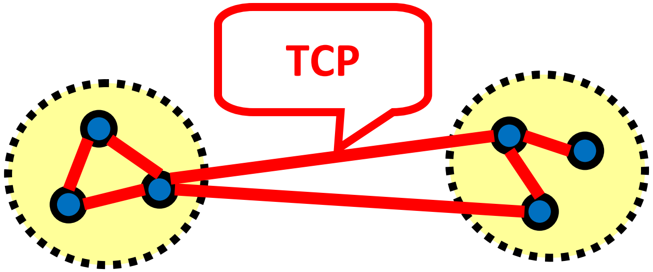
Proposal Overview

- Basic Proposal
  1. Construct a TCP overlay
  2. Apply deadlock-free routing constraints
  3. Calculate routing paths

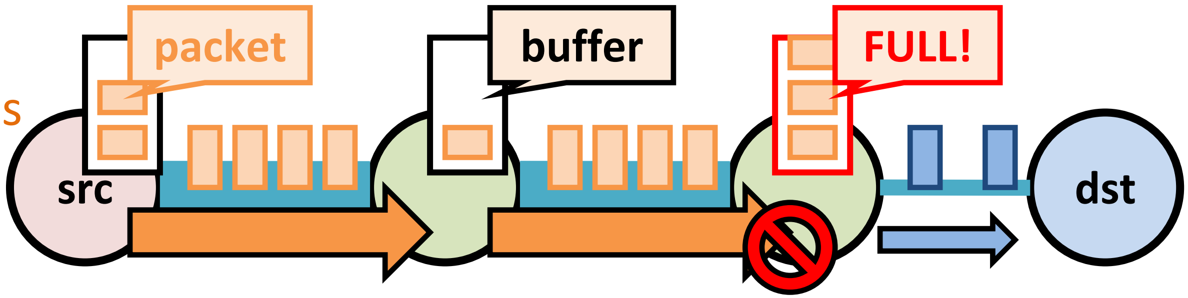
- Optimizations using network information

Basic Overlay Overview

- ***Only*** requires a connected overlay network using TCP connections
  - e.g.: random graph overlay construction

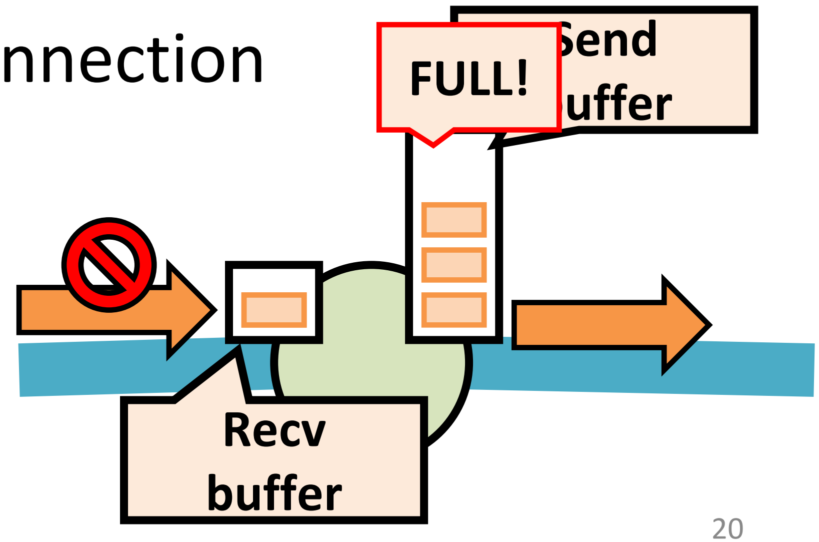


- Send in packets：
  - Predefined packet sizes
- End-End reliable communication：
  - FIFO transfer
  - Do NOT drop packets



Forwarding Procedure(1/2)

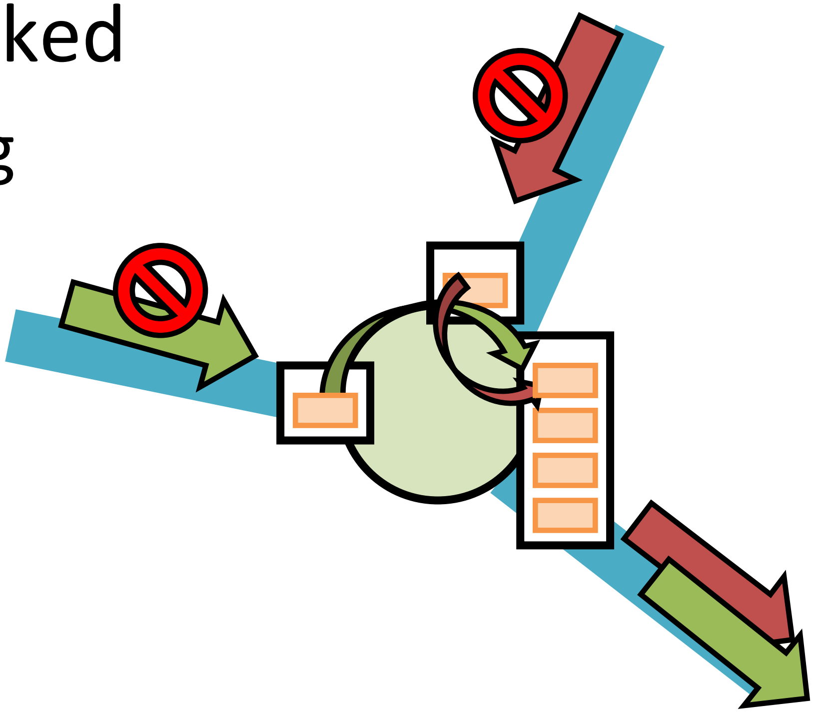
- Define the following per TCP connection
  - Fixed send buffer
  - 1-packet receive buffer
- Transfer procedure
  - receive packet on receive buffer
  - Move to send buffer of connection to be forwarded
  - If send buffer is full, stop receiving on it



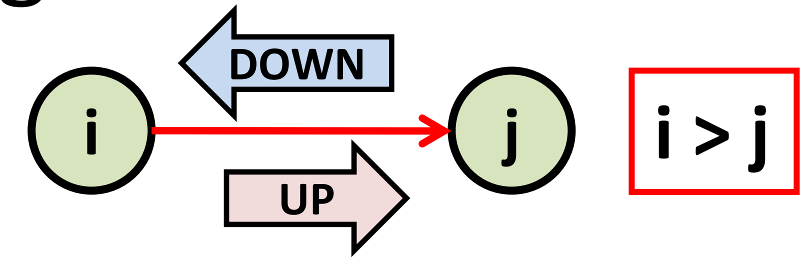
Forwarding Procedure(2/2)

- When multiple transfers contest for single link
  - They will make progress in round-robin fashion
- Transfers will be blocked
  - Deadlock-free routing

    ⇒ ***No*** deadlocks



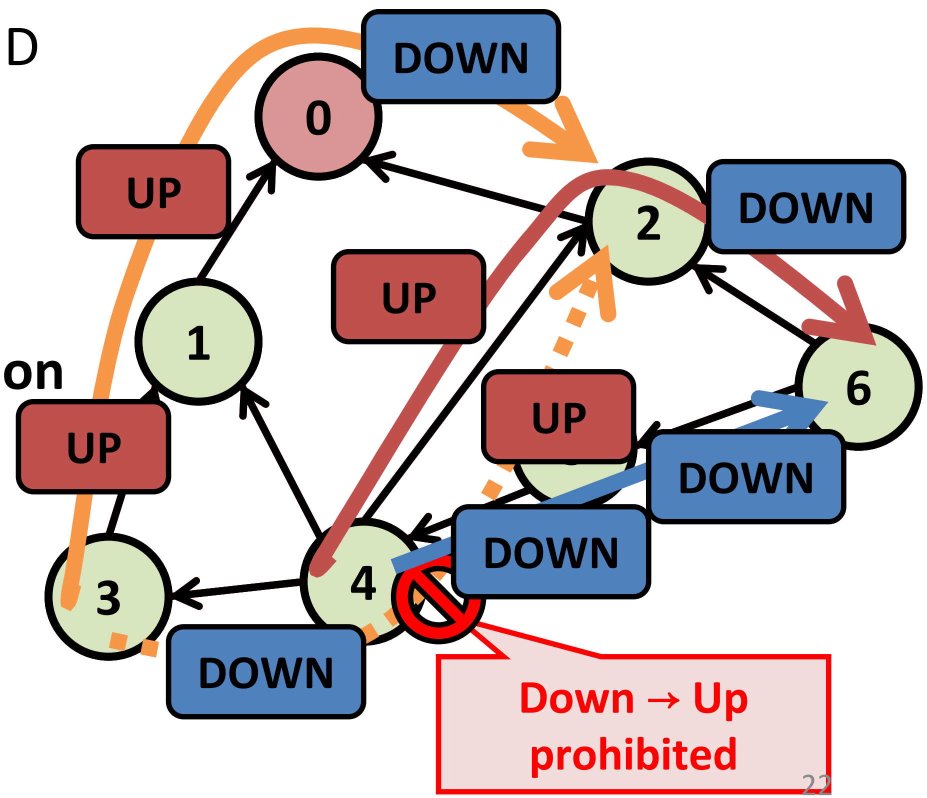
Deadlock-free Routing
Up/Down Routing [Schroeder et al. '91]



- BFS from root node
  - Assign IDs in ascending order
- Determine link **arrow**
  - Arrow points to the younger ID

- Define link **traversal**
  - **UP: in arrow direction**
  - **DOWN: against arrow direction**
- **Routing path restriction:**
  - **Cannot go UP after DOWN**

Determined independently from underlying network



Routing Table Calculation

- Modification to Dijkstra's Shortest Path
  - Routing Table Calculation: **O(*N*log*N*)** for ***N*** nodes

Proposal Overview

- Optimizations using network information

•Inter-node Latency Matrix
•Connection bandwidth information

- Basic Proposal
  - Construct a TCP overlay
  - Apply deadlock-free routing constraints
  - Calculate routing paths

Locality-aware construction

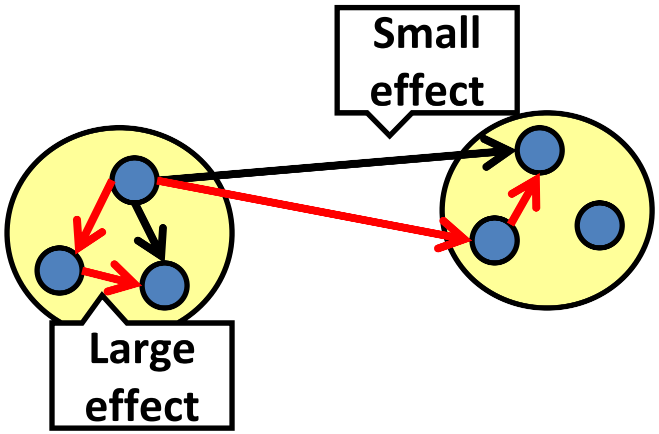
Locality-aware constraints

Throughput-aware Path calculation

Locality-aware overlay construction

[Saito et al. ‘07]

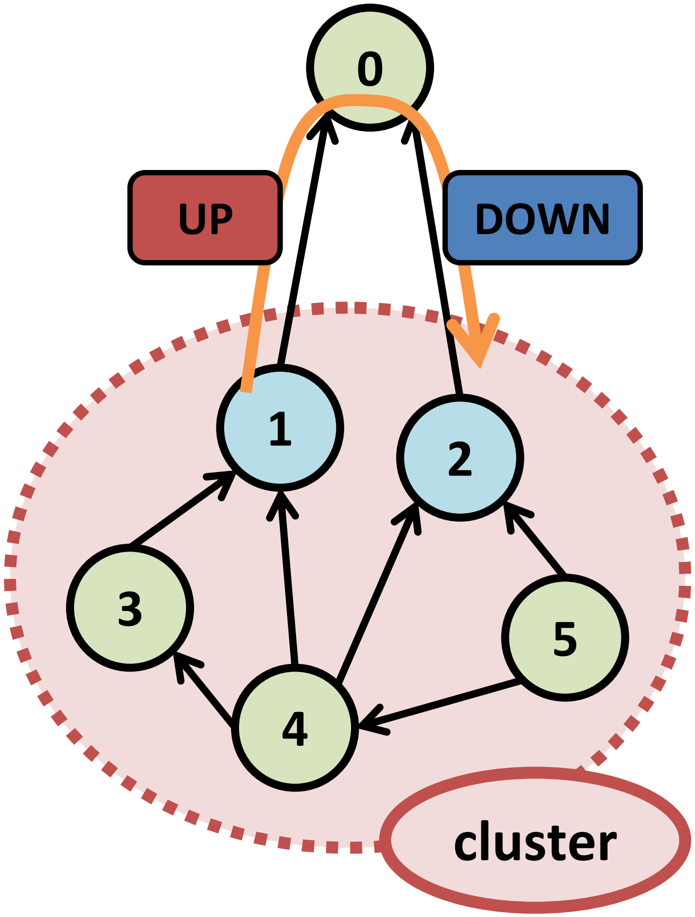
- “Routes to far nodes can afford to make detours”
- Connections choice
  - **Low** prob. With far-away nodes
  - **High** prob. With near nodes



- Reduce connections without performance impact

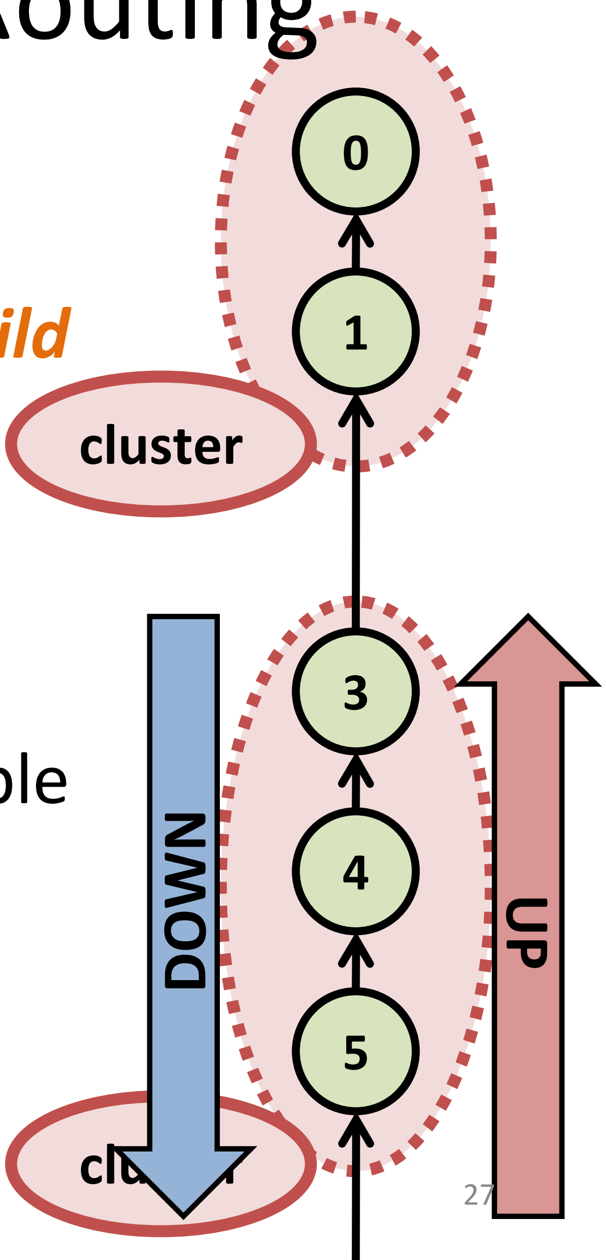
Up/Down Routing Optimizations

- BFS id assignment is problematic in multi-cluster settings
- Many nodes are reachable only with **UP** → **DOWN** paths
- Nodes with small IDs within cluster
  - UP direction traversal includes a high-latency WAN connection
  - They will use WAN links to reach intra-cluster nodes



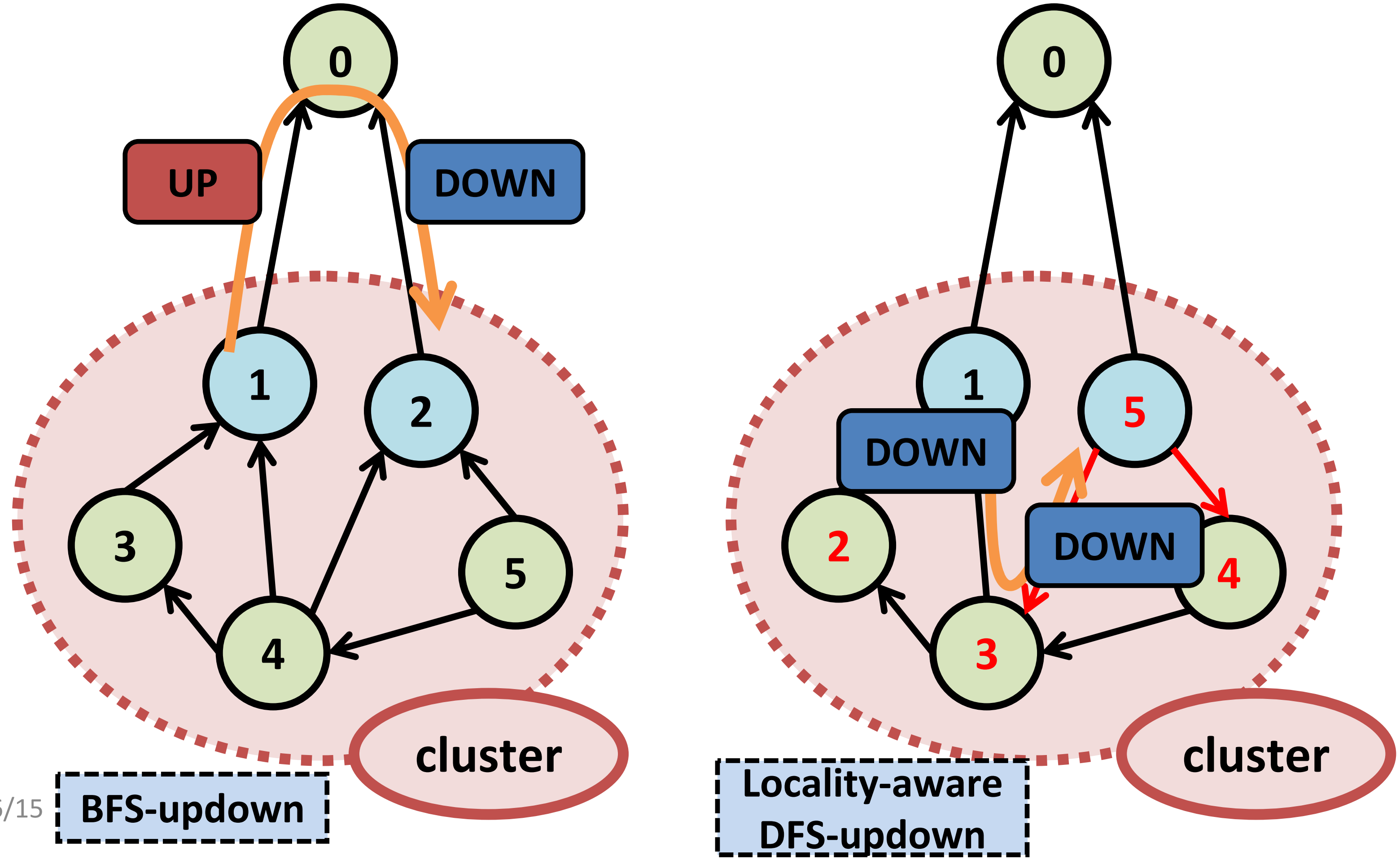
Proposed Up/Down Routing

- **DFS** ID assignment
  - traverse priority to ***low latency child***
- Rationale
  - Reduce UP→DOWN paths
  - Intra-cluster nodes can be reachable only using UP or DOWN traversals
  - **Reduce unnecessary WAN hops**



Deadlock-free restriction comparison

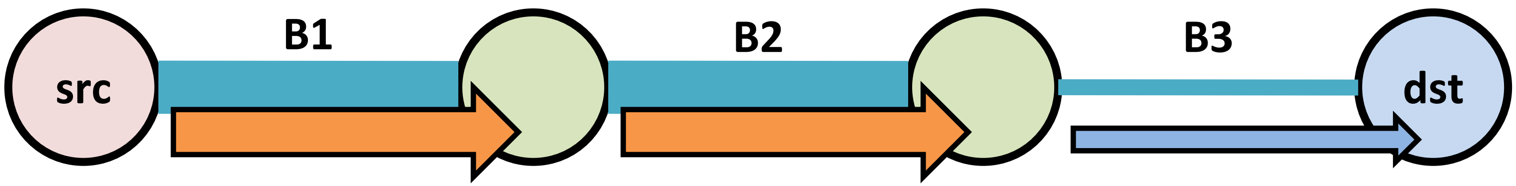
- Reduce restrictions banning intra-cluster links



Routing Metric

- Give weight to throughput of entire path
  - Sum of inverse of bandwidth of used links

$$Cost = \sum_{i=1}^{N} \frac{1}{B_i}$$

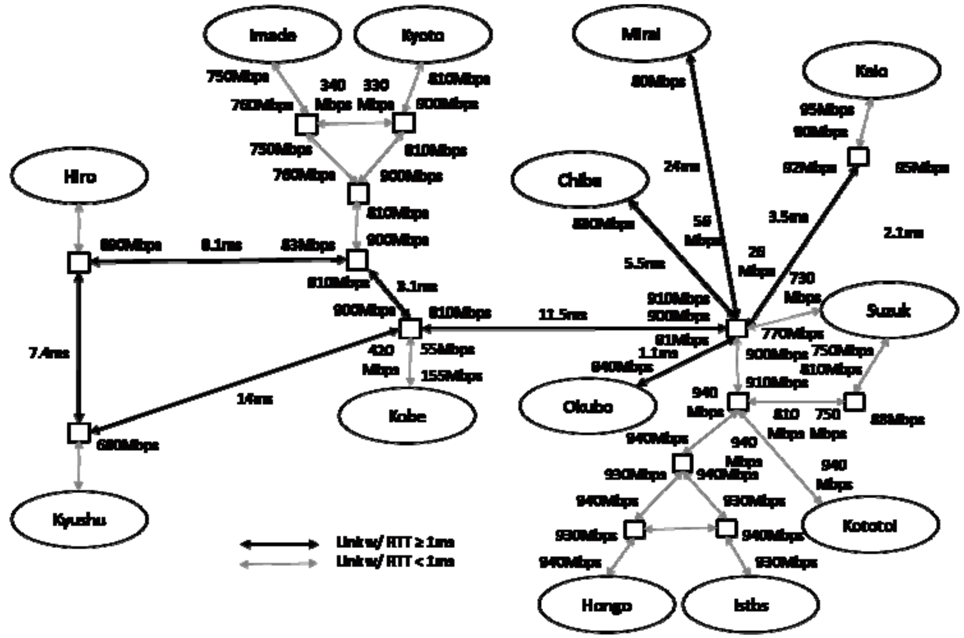


Evaluation

- Introduction
- Problem Setting
- Related Work
- Proposal
- Evaluation
- Conclusion

Deadlock-free Routing Overhead

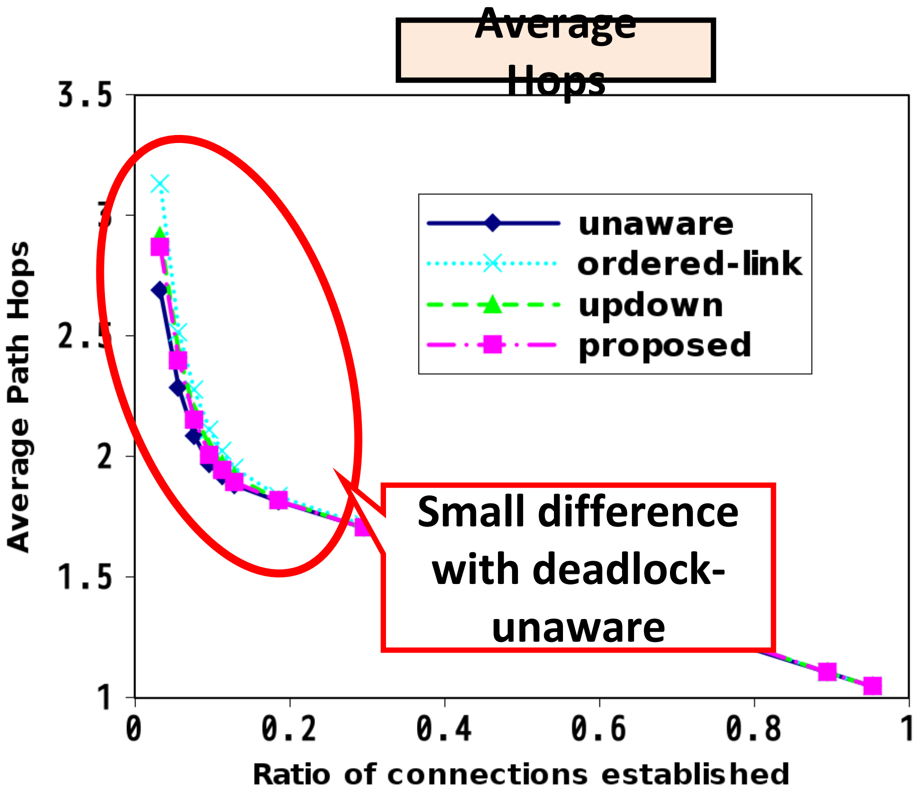
- Compare deadlock-***free*** vs. deadlock-***unaware*** routing
  - ordered-link
  - Up/Down
  - Proposed Up/Down
- Compared hops/bandwidth for all calculated paths

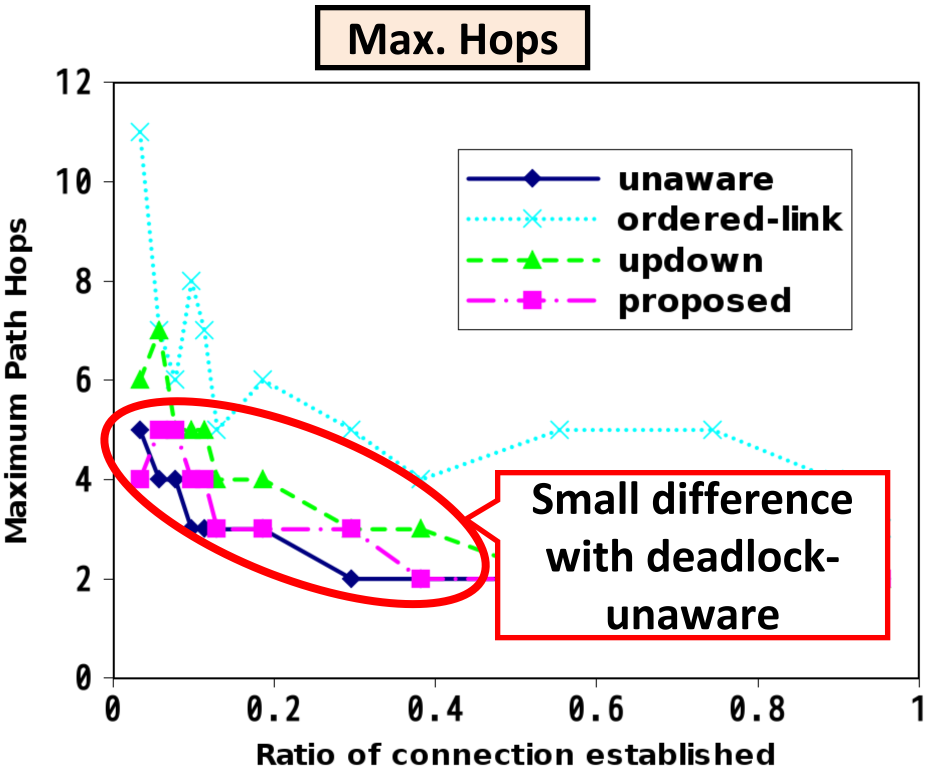


- Simulation
  - L2 topology information of the InTrigger Grid platform
    - 13 clusters (515 nodes)

1. Vary connection density
2. Computed routing tables
3. Evaluate result using topology information

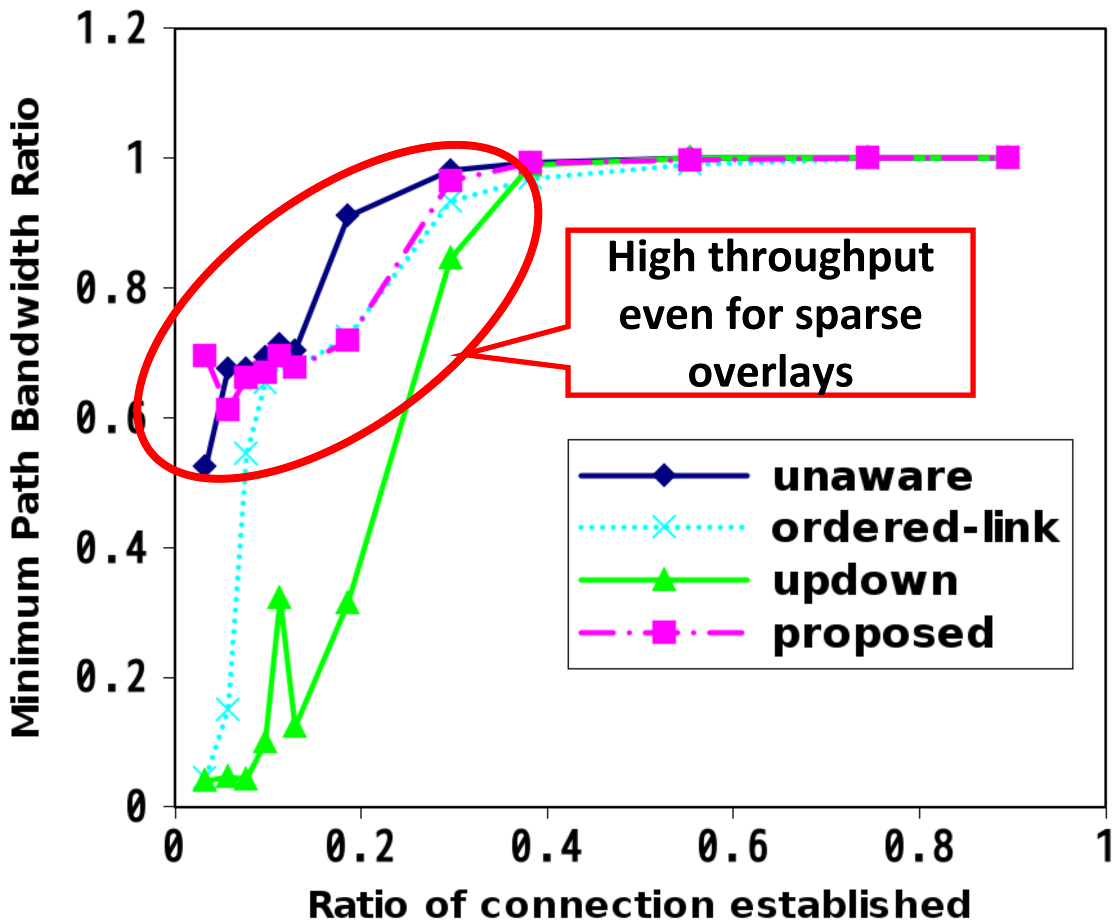
Num. of hops for all paths





- Very small difference for average hop count
- Proposed Up/Down has comparable max. hop count

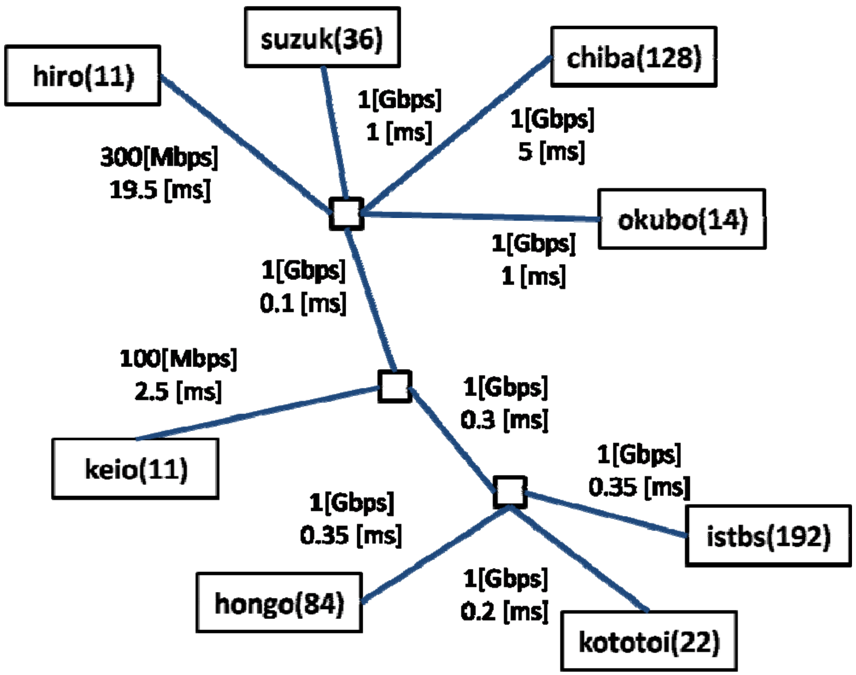
Minimum Path bandwidth Ratio



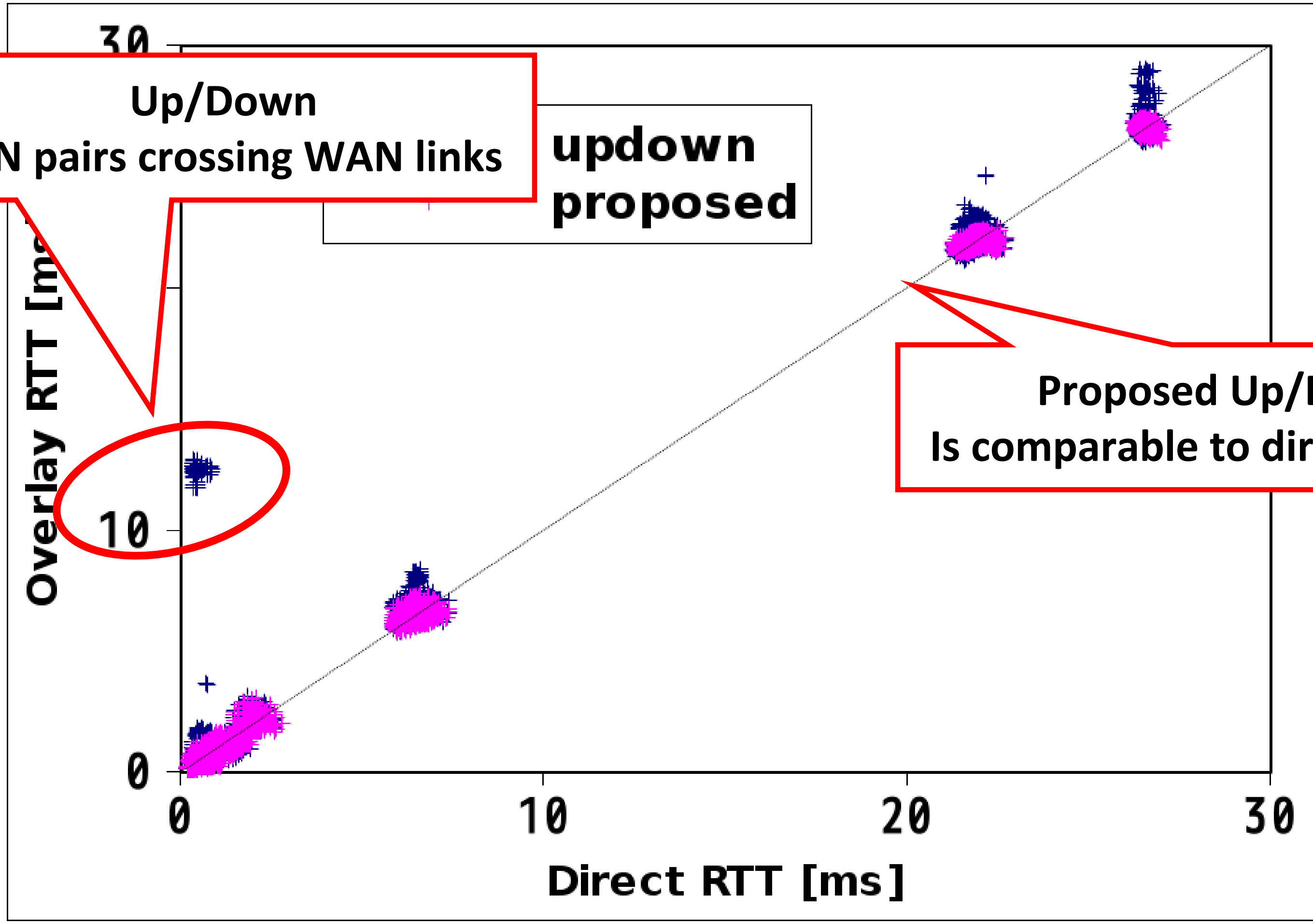
- **Other deadlock-free algorithms take unnecessary WAN-hops**
- **Proposed optimization avoids taking WAN-hops**

Deadlock-free routing effect on path restriction and latency

- Comparison to direct sockets
- 7 Real clusters on InTrigger (170 nodes)
  - Connection density: 9%
- Routing Metric: Latency over path



Direct vs. Overlay Latency



Up/Down
LAN pairs crossing WAN links

Proposed Up/Down
Is comparable to direct sockets

- Up/Down uses WAN links even for LAN communication

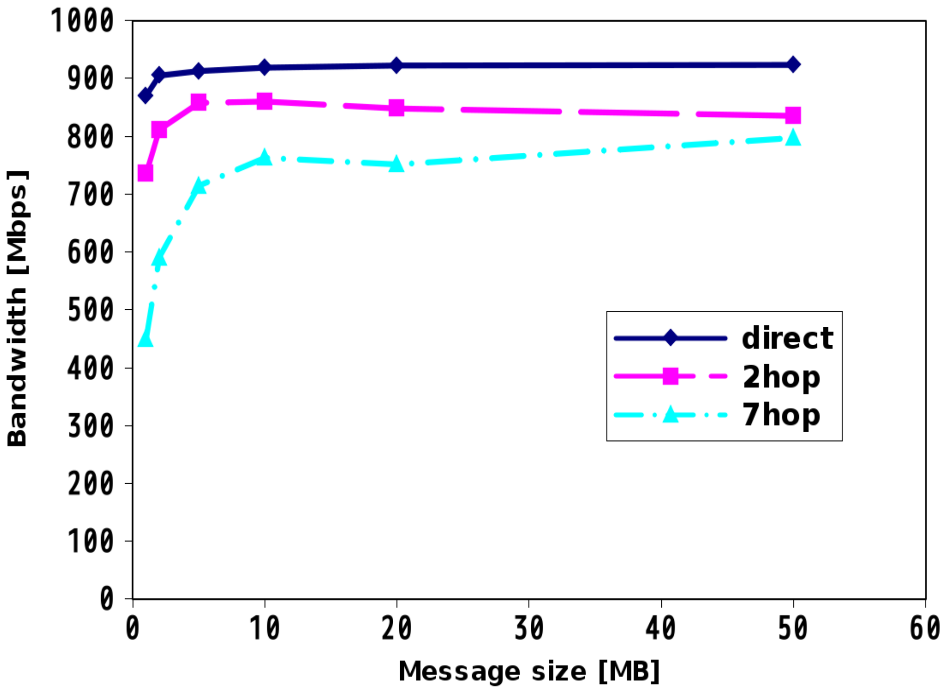
Overlay Throughput Performance

- A wide range of environments
  - 1 Gigabit Ethernet LAN (940 [Mbps])
  - Myrinet 10G LAN (7 [Gbps])

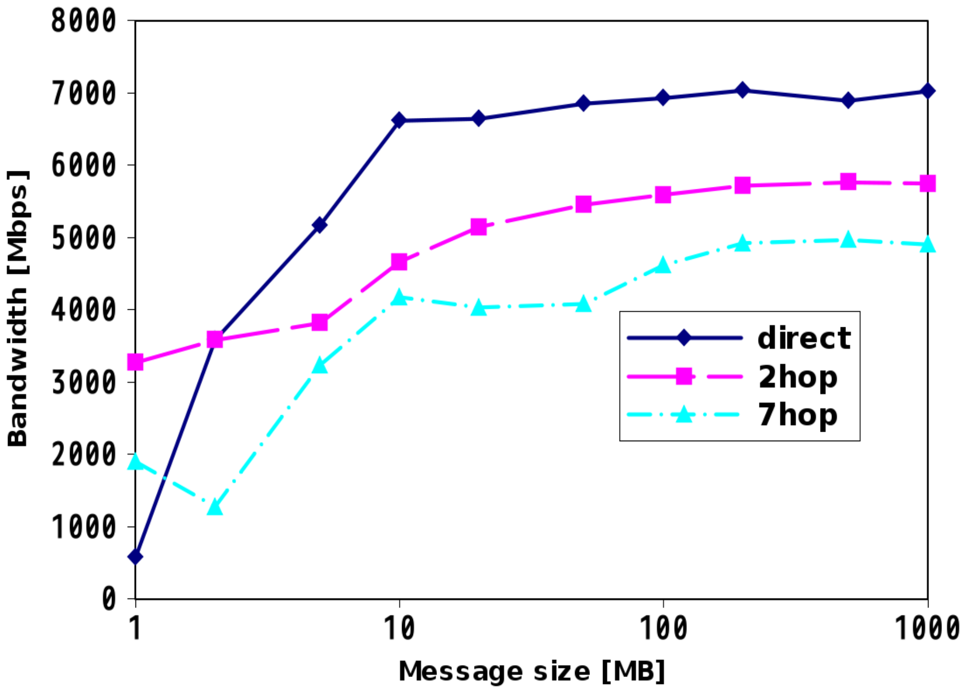
- With varying number of intermediate nodes

Direct vs. overlay throughput

GbE cluster (940[Mbps])



Myrinet cluster (7[Gbps])



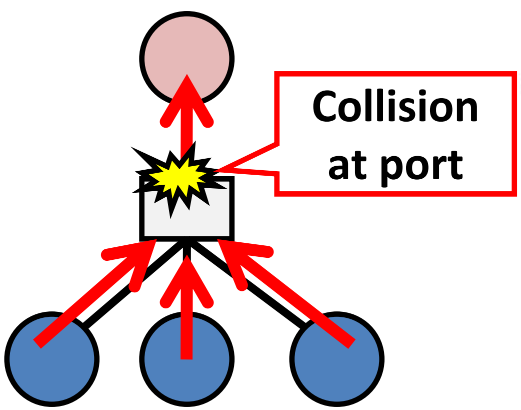
- **Able to attain close to direct socket throughput**

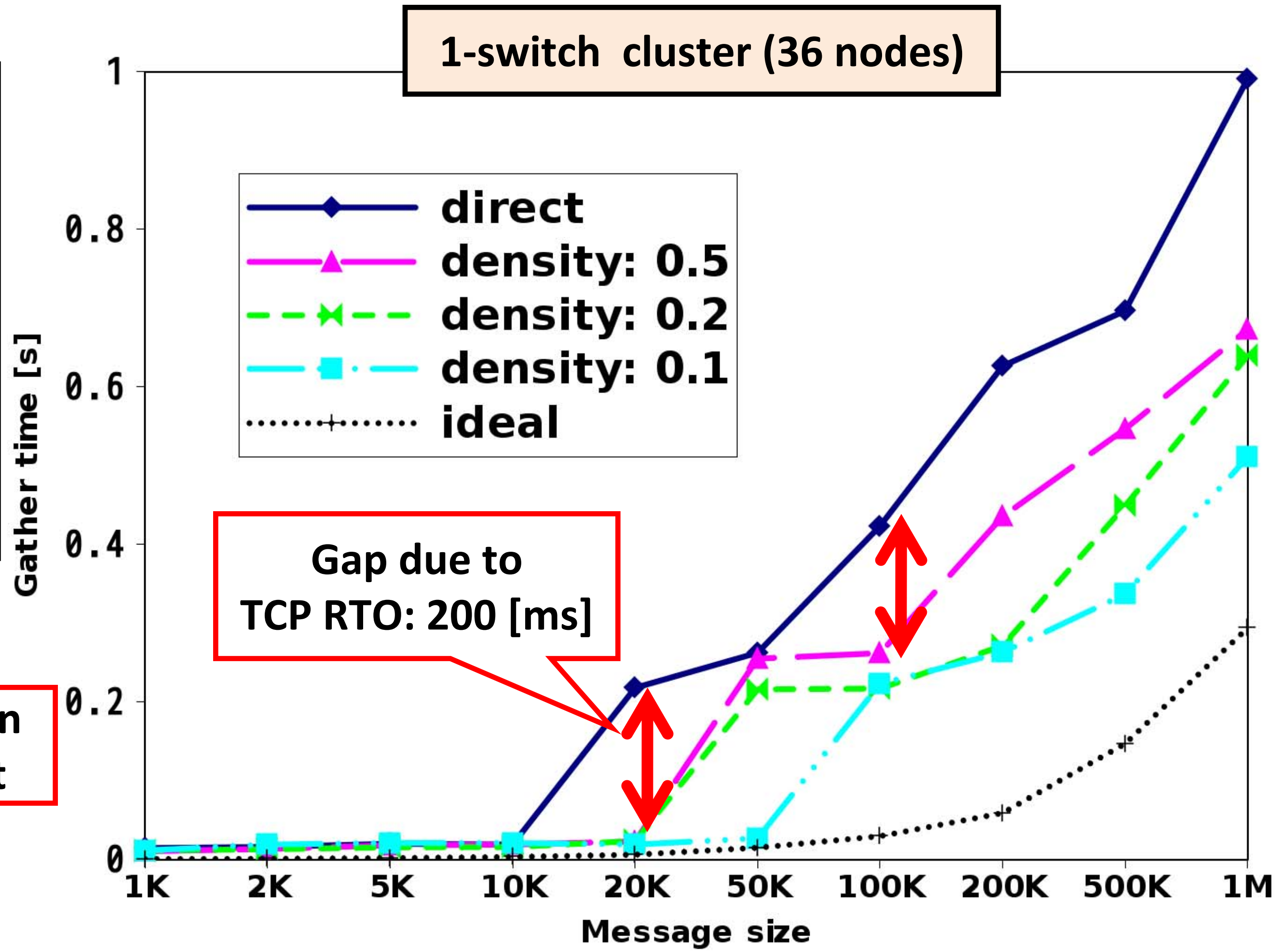
Collective Communication

- Our overlay ***outperforms*** direct sockets even with deadlock-free constraints
- Evaluation
  - Gather, All-to-All
  - Varying message size, and connection density
- Environment
  - LAN: 1-switch (36 nodes), hierarchical (177 nodes)
  - WAN: 4 clusters (291 nodes)

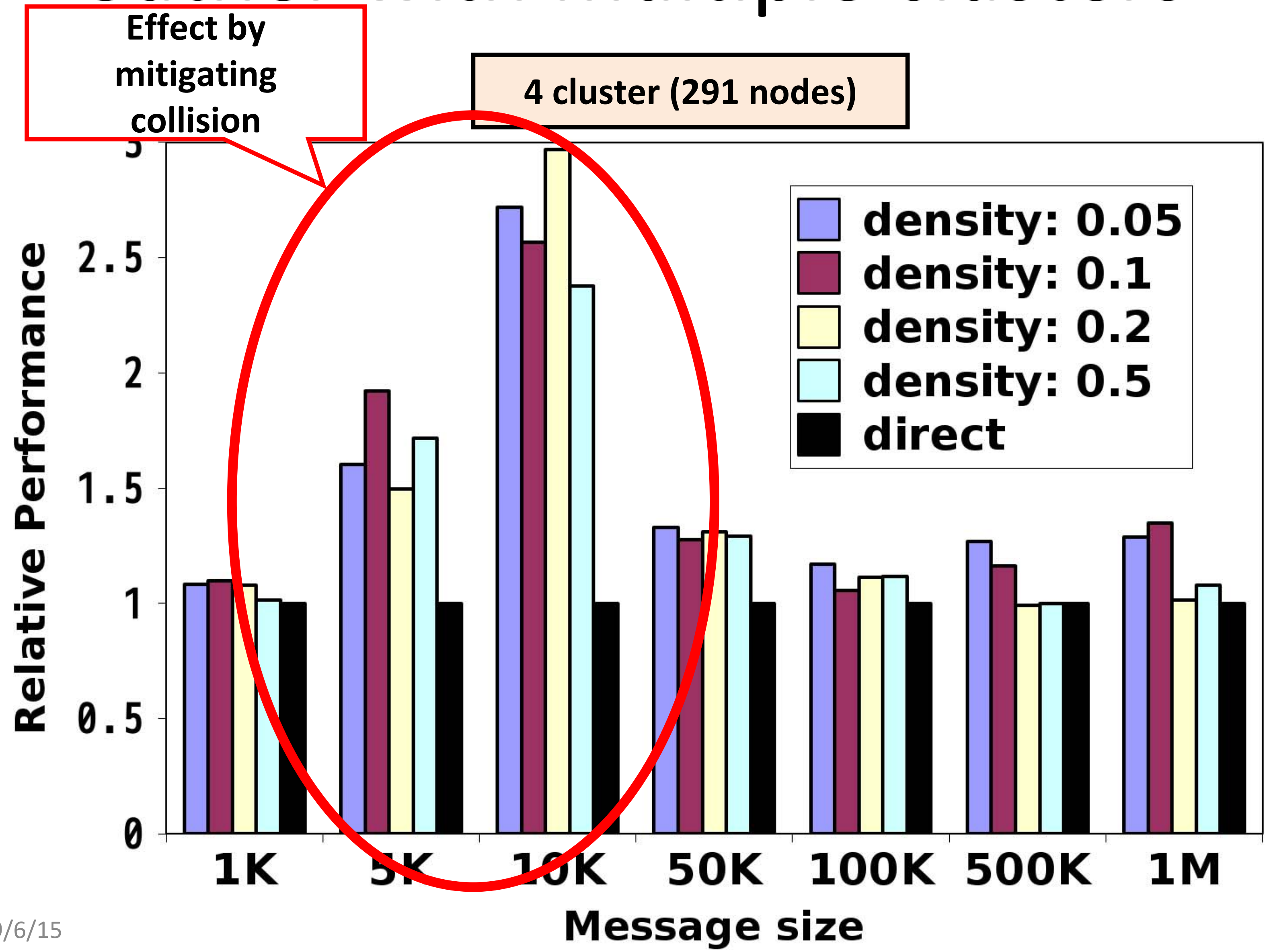
Gather time

- Collision at switch:
  - Packet-loss
  - TCP retrans.:
  - **200 [ms] loss**
- Sparse overlay:
  - **Mitigates collision**



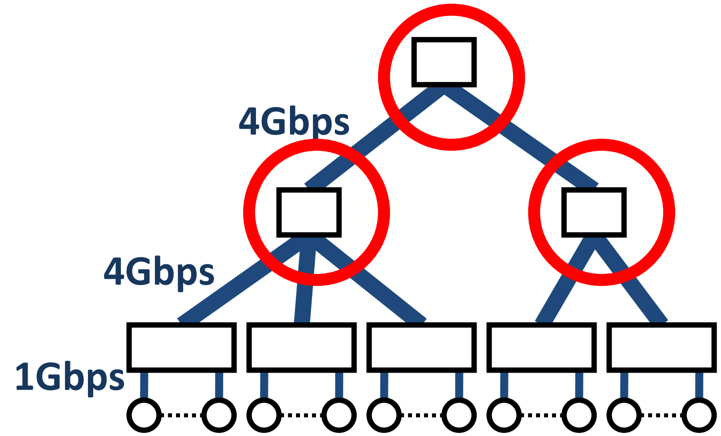


Gather with multiple clusters

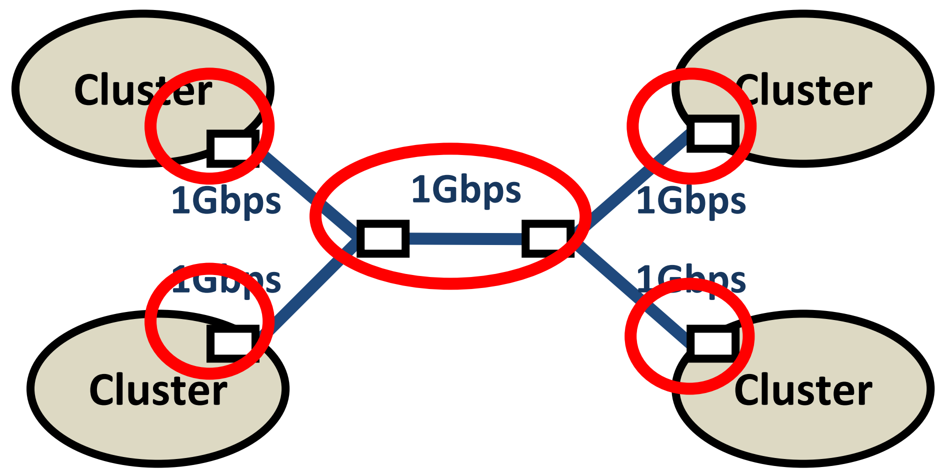


All-to-All

- Large-scale environments have bottlenecks
  - Hierarchical cluster
    - 177 nodes

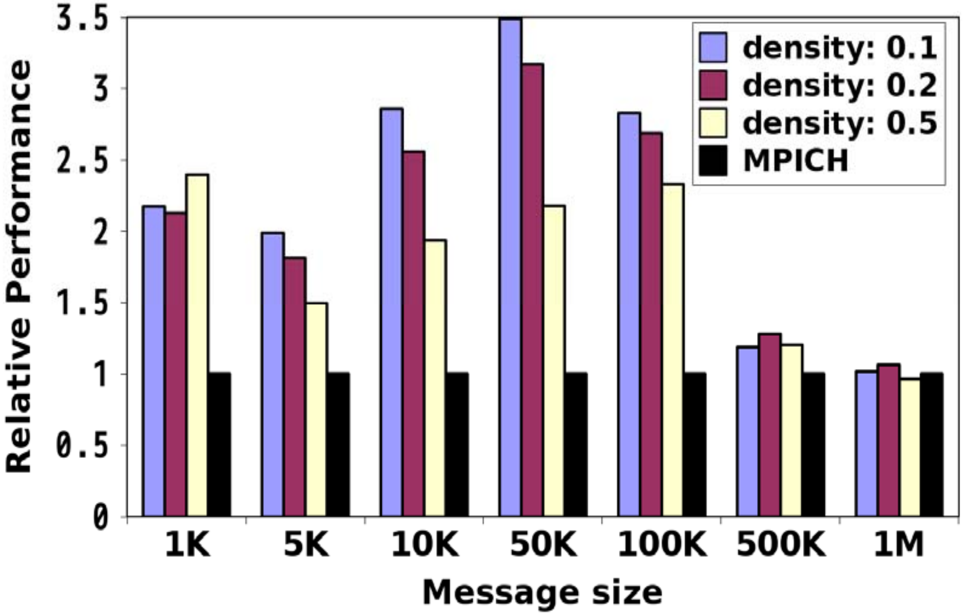


  - 4 clusters connected on WAN
    - 291 nodes

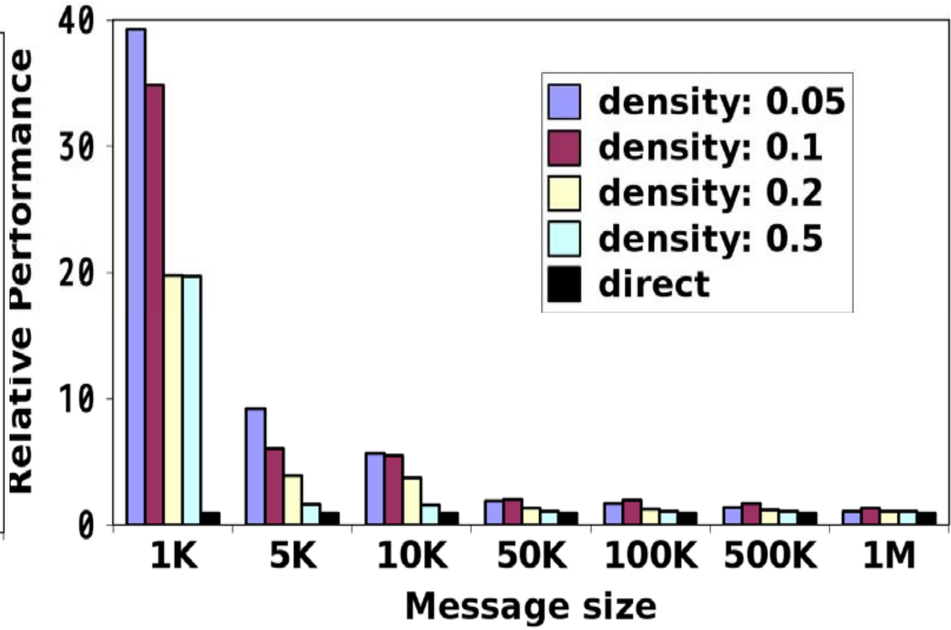


All-to-All performance

1 cluster (177 nodes)



4 cluster (291 nodes)



- **Sparse overlays perform better due to packet loss avoidance**
- **For hierarchical cluster, packet loss occurs at switches**
- **For multi-cluster setting, WAN becomes source of packet loss**

Conclusion

- Introduction
- Problem Setting
- Related Work
- Proposal
- Evaluation
- Conclusion

Conclusion

- Overlay for effective parallel and distributed computing on WANs
  - Low transfer overhead
  - No memory overflow/deadlocks
  - Use network information to mitigate routing overhead
- Evaluation on simulation/LANs/WANs
  - Low overhead relative to deadlock-unaware routing
  - Throughput/latency comparable to direct sockets
  - Outperforms direct sockets for collective communication
- Future Work
  - Allow dynamic changes in overlay topology and routing

Questions?

- Ken Hironaka

kenny@logos.ic.i.u-tokyo.ac.jp

- Taura Research Lab

www.logos.ic.i.u-tokyo.ac.jp