Multi-Path Transport for RDMA in Datacenters

Yuanwei Lu, et. al.
Microsoft Research
USENIX NSDI `18

Presenter: Junghwan Song

Outline

- Introduction
- Backgrounds
  - Remote Direct Memory Access (RDMA)
  - RDMA over Converged Ethernet (RoCE) v2
- MP-RDMA
- Evaluation
- Conclusion

Introduction

- RDMA provides ultra-low latency (~1μs) and high throughput (40/100Gbps) with little CPU overhead
- Recently, RDMA has been deployed in datacenters at scale with RDMA over Converged Ethernet (RoCE) v2
- RDMA is a single path transport
  - Prone to path failures
  - Cannot utilize the rich parallel paths in modern datacenters

Main idea

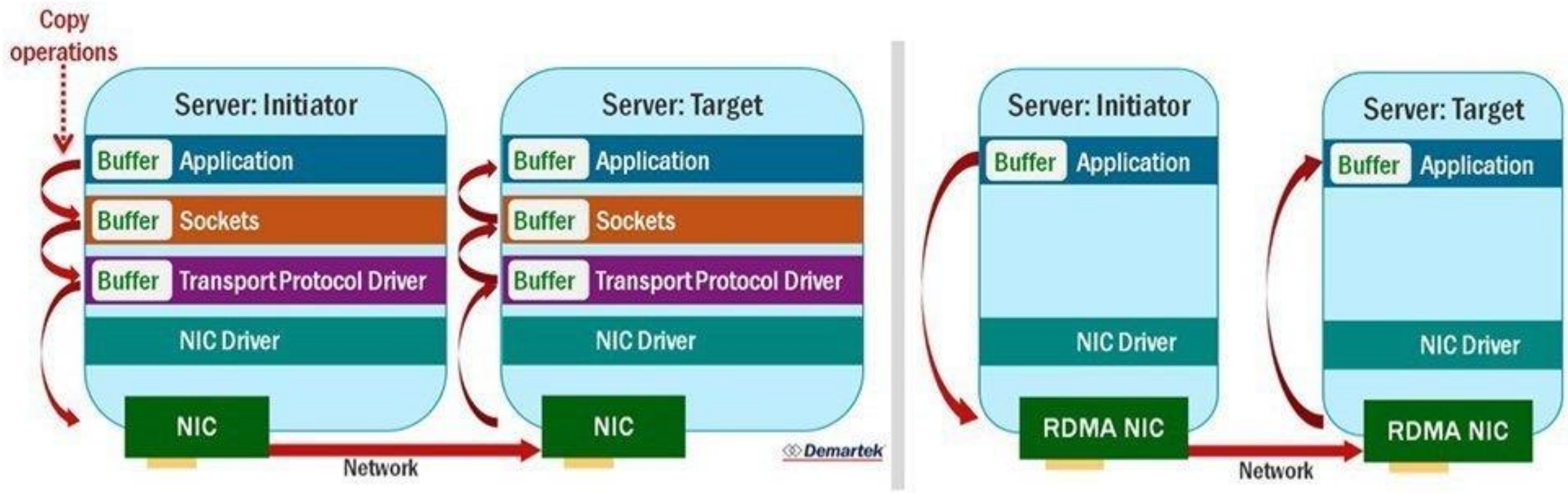
- Design RDMA transport supporting multiple paths
- Constraints
  - RDMA is completely implemented in NIC hardware
    - Limited computing resource
    - Small on-chip memory
- Key concept: Minimize memory footprint

Key challenges

- 1. Tracking path condition
  - Per-path condition is basis of congestion control
- 2. Metadata overhead
  - Out-Of-Order (OOO) packets should be tracked (whether a packet has arrived or not)
- 3. Out-of-order memory update
  - OOO packets cause OOO memory updates, leading to application failures

Backgrounds

RDMA

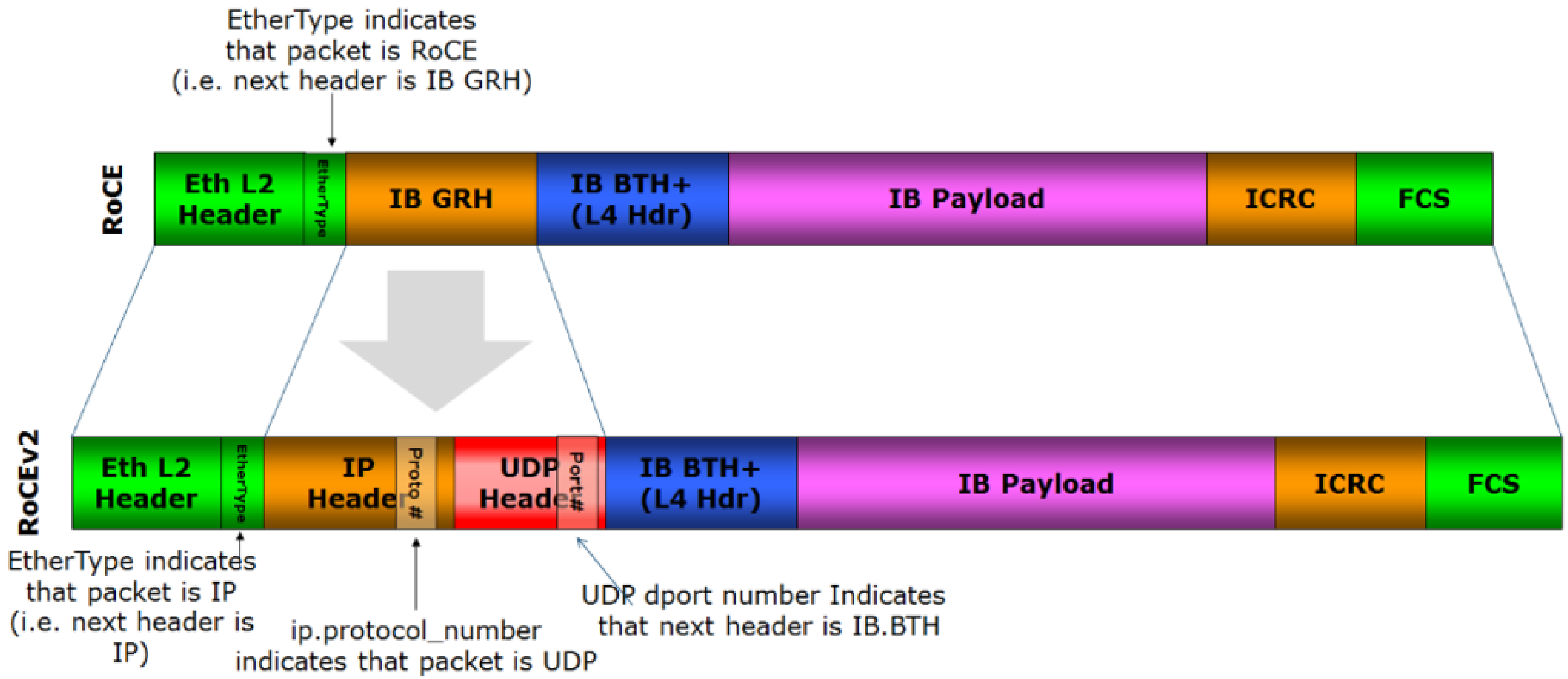


- RDMA enables direct memory access to remote system
  - Low latency and high throughput with little CPU involvement
- Transport should be entirely implemented on Network Interface Card (NIC)
- RDMA needs lossless network
  - e.g., Priority-based Flow Control (PFC)

RDMA operation

- RDMA connection is identified by an Queue Pair (QP)
  - Send Queue (SQ) and Receive Queue (RQ) on NIC
- Applications initiate RDMA operation with post a Work Queue Element (WQE) to SQ or RQ
- To close connection, Completion Queue Element (CQE) is sent to Completion Queue (CQ) by applications

RoCE v2



- RoCE v2 introduces UDP/IP/Ethernet encapsulation to be run over generic IP networks
  - Ethertype 0x8915 indicates RoCE
  - UDP destination port number 479 is reserved for RoCE v2

MP-RDMA Design

Reminder of key challenges

- 1. Tracking path condition
- 2. Metadata overhead
- 3. OOO memory update

Reminder of key challenges

- 1. Tracking path condition
- 2. Metadata overhead
- 3. OOO memory update

ACK-clocking congestion control

Compress header with bitmap

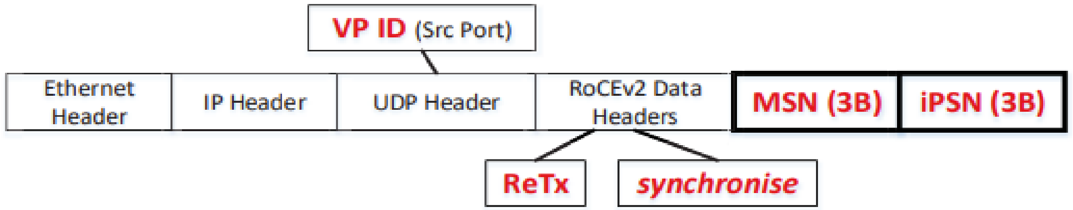
OOO aware path selection
Synchronise operation

Mechanisms overview

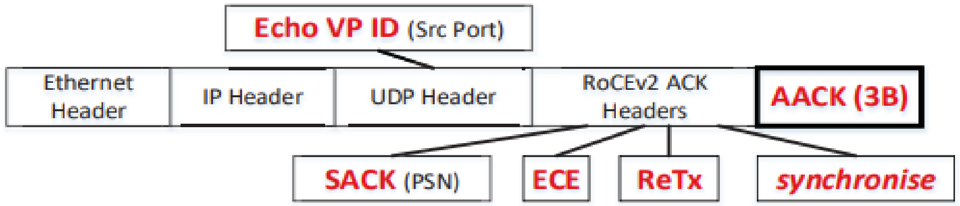
- ACK-clocking and congestion control mechanism
  - Congestion-aware load distribution without maintaining per-path states
- OOO aware path selection mechanism
  - Control the OOO degree among sending paths, thus minimizes the metadata size required for tracking OOO packets
- Synchronise mechanism for applications
  - Ensure in-order host memory update without sacrificing throughput

ACK-clocking and congestion control mechanism

- Use Virtual Path (VP) ID
  - VP is in UDP src port
  - Send a packet through VP that an ack came from



(a) MP-RDMA data packet header



- Use one congestion window for all paths

For each received ACK:

$$cwnd \leftarrow \begin{cases} cwnd + 1/cwnd & \text{if } ECN = 0 \\ cwnd - 1/2 & \text{if } ECN = 1 \end{cases}$$

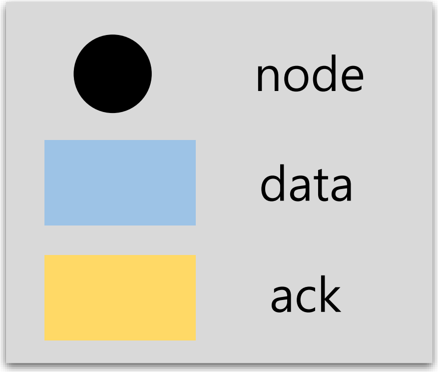
Illustration

Sender sends data
with arbitrary VP IDs

Illustration

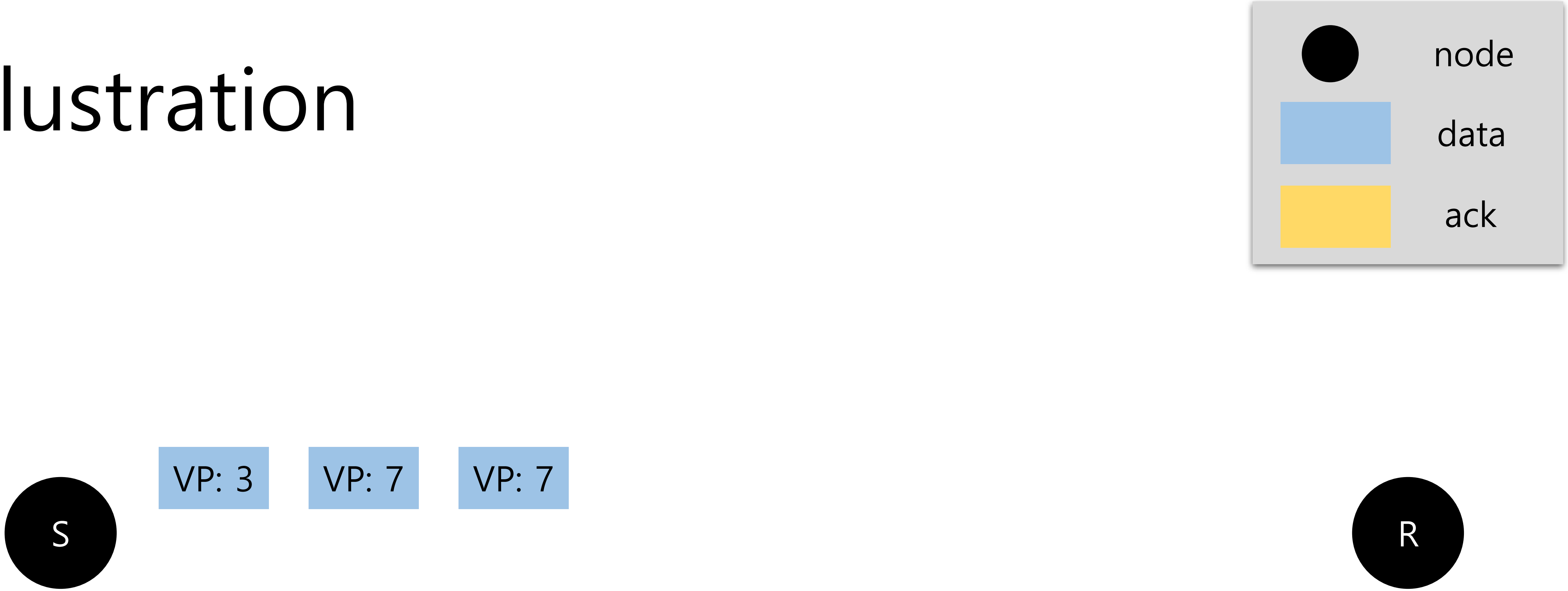
Receiver copies VP IDs in data
Receiver sends acks with them

Illustration



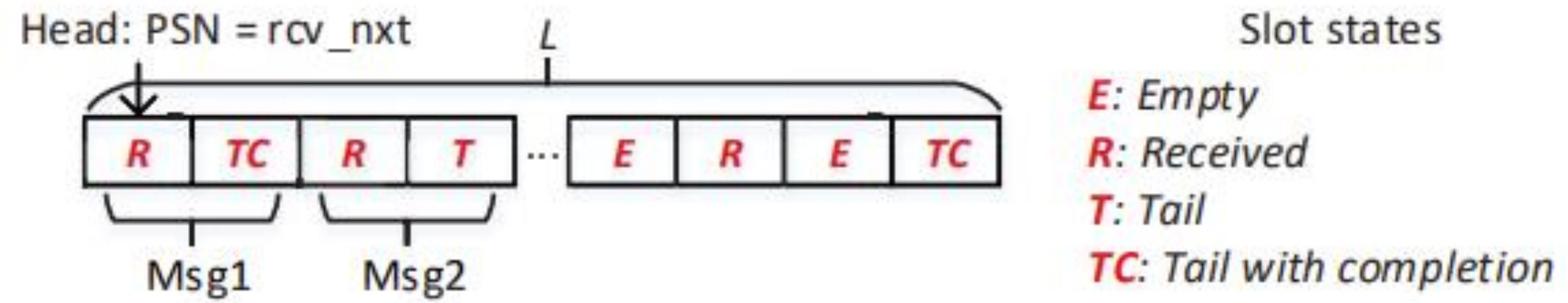
Sender adjusts cwnd
Sender sends data based on adjusted cwnd

Illustration



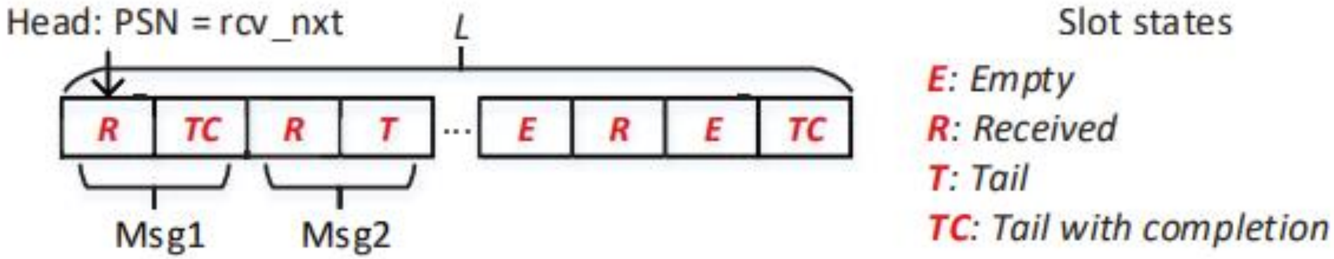
If cwnd is increased, sender sends one more data packet
VP ID of the data is same with the ack which increases cwnd

OOO tracking bitmap



- OOO is common in multiple path transmission
- For tracking OOO packets, data structure is needed
- To minimize NIC memory footprint, employ a simple bitmap at the receiver

Bitmap operations



- When a packet arrives, receiver
  - checks PSN in the packet header
  - finds the corresponding slot in the bitmap
  - fill the bitmap with R, T, or TC states

- Receiver continuously check the bitmap
  - A continuous block of slots are marked as Received with the last slot being either Tail or Tail with completion → clears these slots to Empty and moves the head point after this message

OOO aware path selection

- If an out-of-order packet holds a PSN larger than (rcv_nxt + L), the receiver has to drop this packet
  - L is size of the bitmap
- Core idea is to actively prune the slow paths and select only fast paths with similar delay
- Decrease cwnd by 1 if received ack's PSN is lower than (the highest sequence number – Δ)
  - Δ is parameter, <= L

Synchronise motivation

- To buffer OOO packets, host memory should be used
  - NIC does not have enough memory space
- When whole packets are received, re-ordered packets are copied to right location
  - Cause significant overhead: twice traverse of PCIe bus
- MP-RDMA chooses directly place OOO packets into app memory
  - Minimize overheads

Synchronise operation

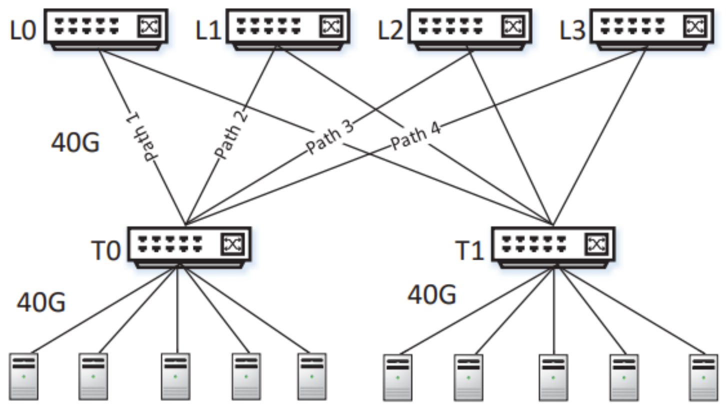
- Direct app memory placing might be not suitable with order-sensitive applications
  - e.g., Key-value store using RDMA write operation

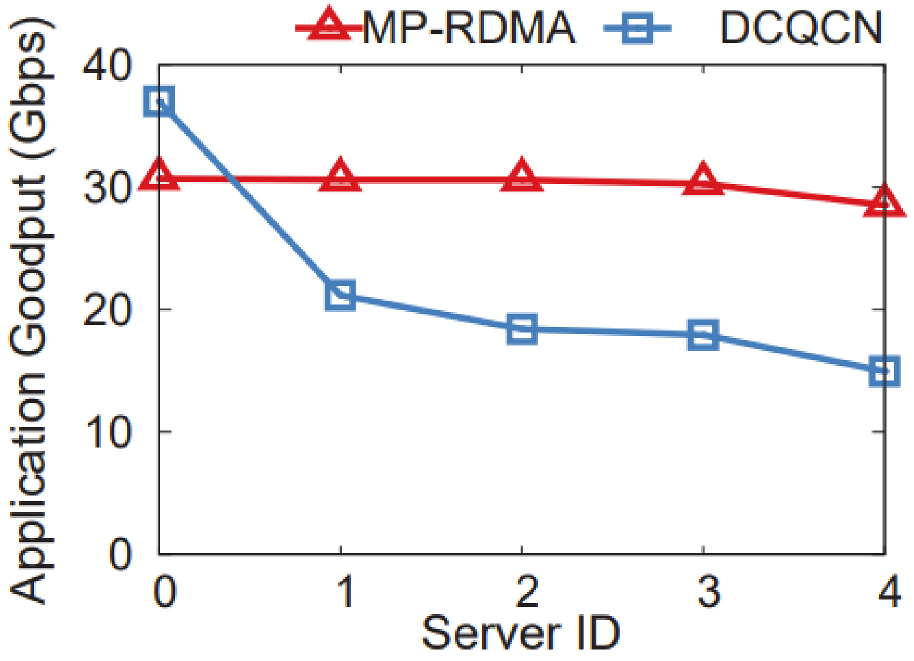
- MP-RDMA adds ‘synchronise’ flag

- Syn flagged packet is processed only after previous operations are completed

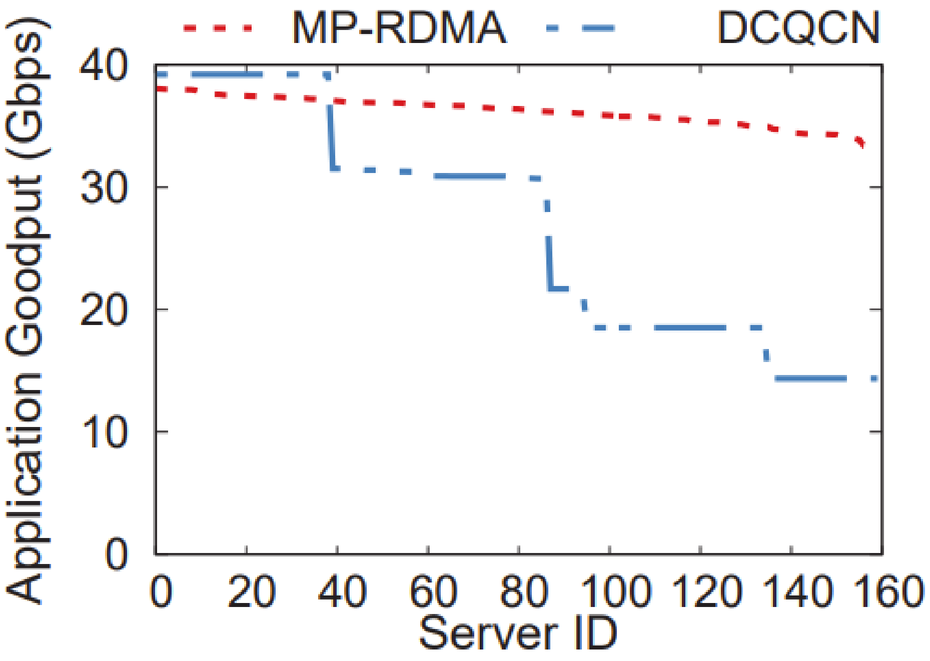
Evaluation & conclusion

Evaluation





(a) Small-scale testbed.



(b) Large-scale simulation.

Conclusion

- RDMA provides ultra-low latency and high throughput with little CPU overhead
- RDMA has been deployed in datacenters, however, multiple paths are not many considered
- Authors provide key challenges when RDMA support multiple paths and design MP-RDMA