Host Network Stack Overheads

ECE/CS598HPN

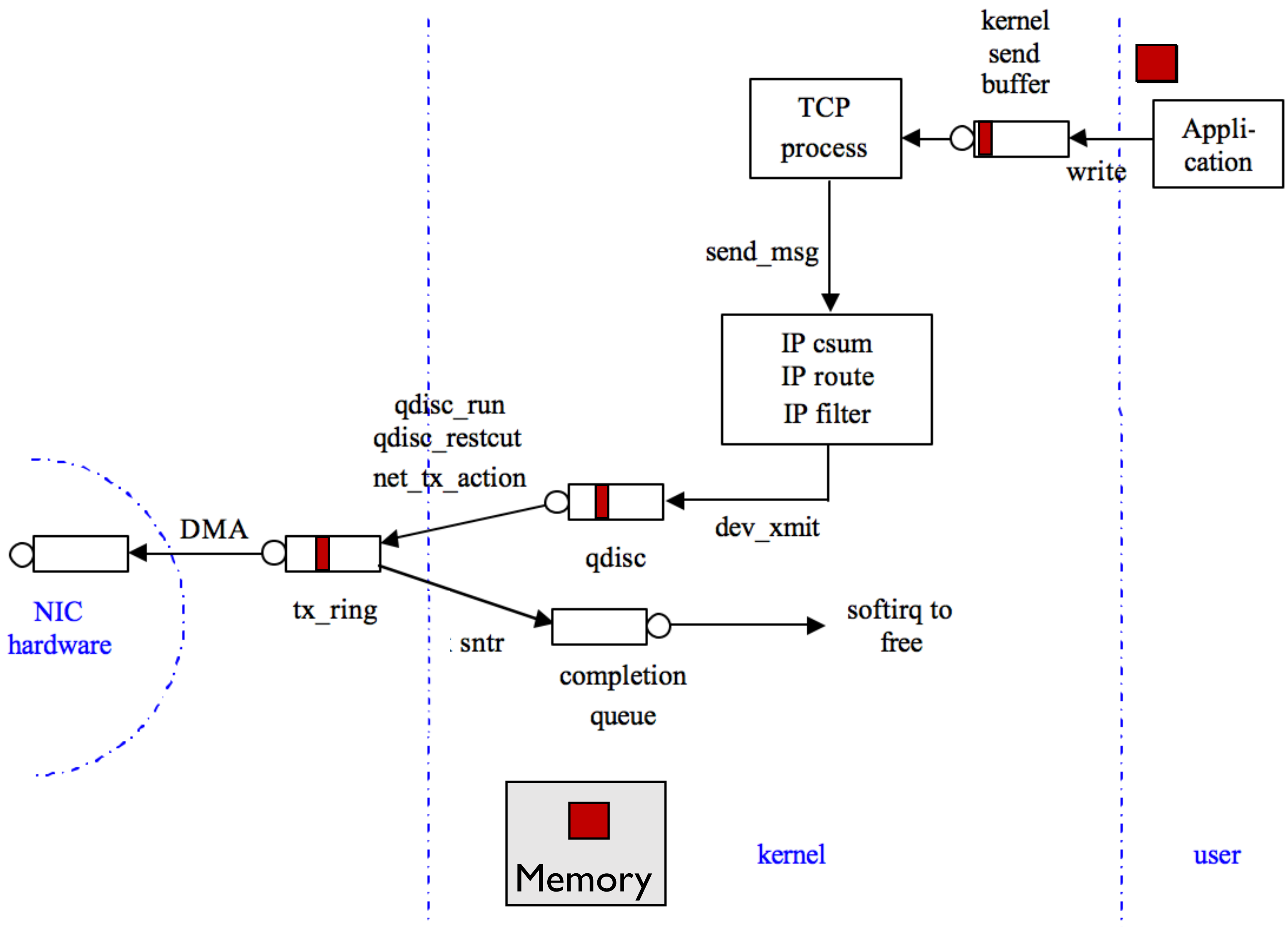
Radhika Mittal

Today’s reading

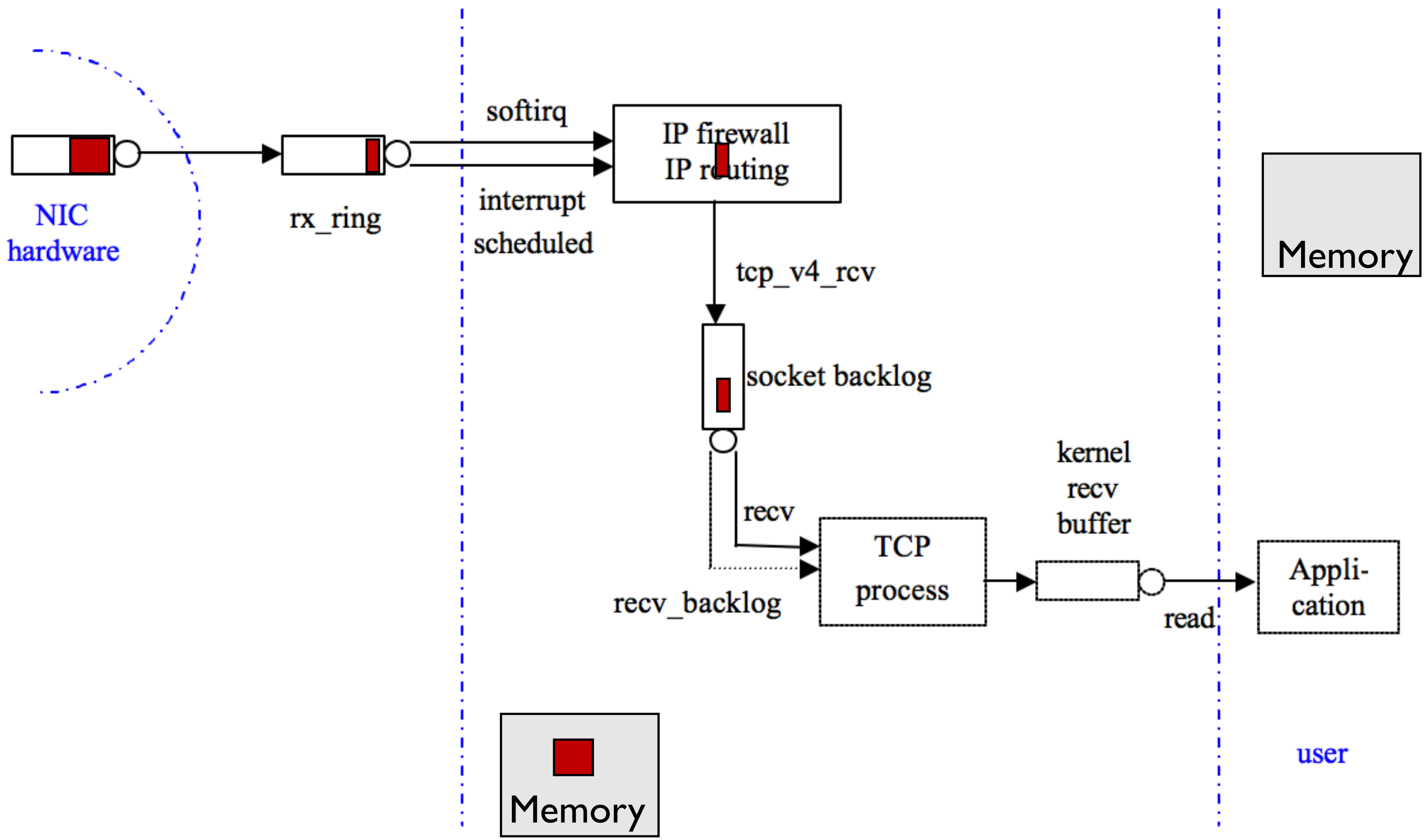
- Understanding Host Network Stack Overheads
  - Cai et. al., SIGCOMM’21

- (Many figures and materials for today’s class taken directly from the paper).

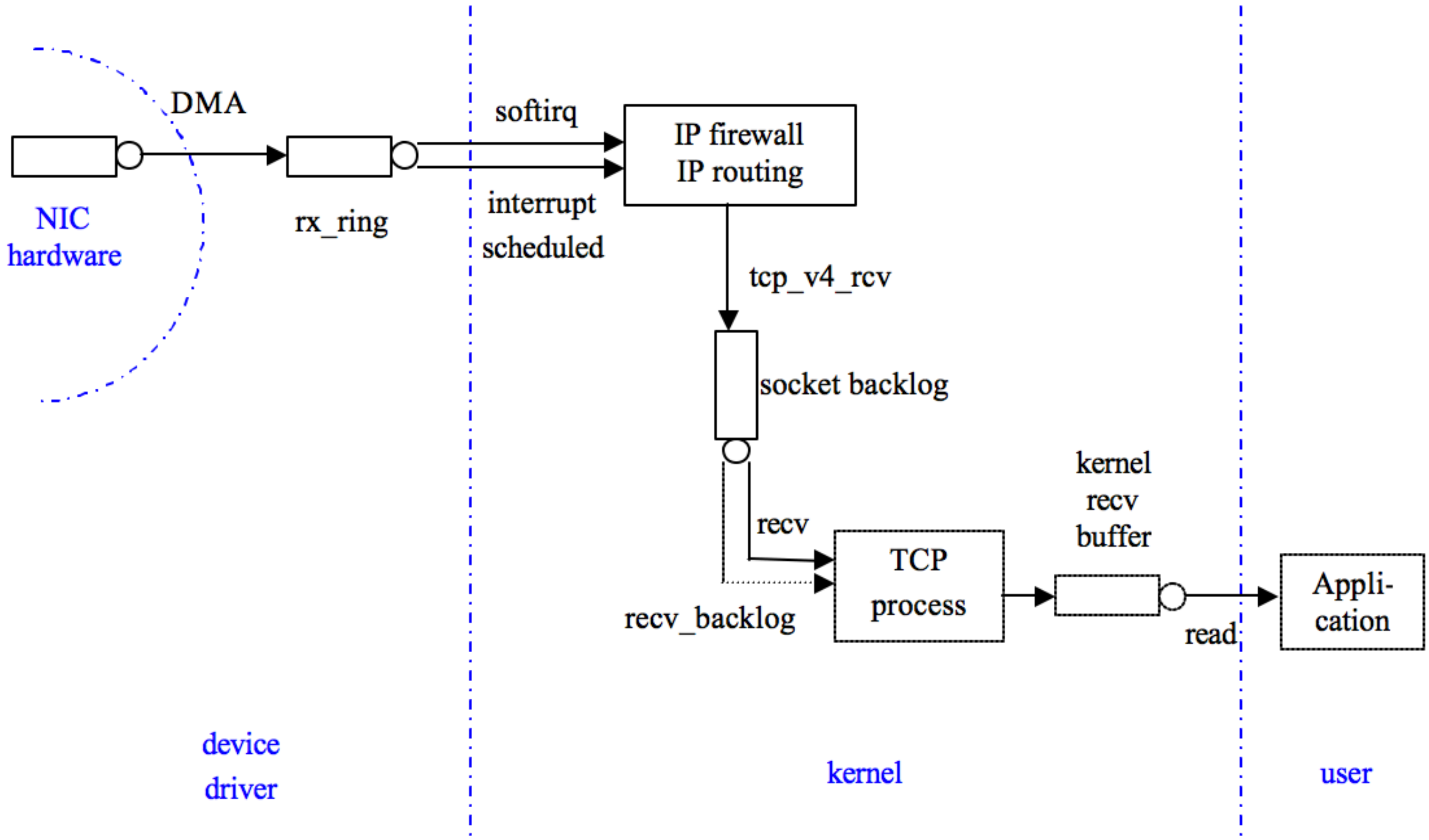
Tx Processing in the kernel



Rx Processing in the kernel



Rx Processing in the kernel



What are some sources of performance overheads?

Sources of overhead

- Protocol processing
- Data copy
- Cache contention (between flows sharing same NUMA node)
- CPU scheduling overheads (locking, context switching)
- Interrupts
- Managing heavy datastructures (skbs)

Optimizations

- NAPI polling:
  - reduces number of interrupts

- DCA/DDIO: Direct Cache Access / Data Direct IO:
  - receiver can DMA packet directly into cache.

Optimizations

- Receive Packet steering / Receive Flow Steering
  - Steering packets to specific queues (cores)
  - Can be done in software / NIC.

Mechanism	Description
Receive Packet Steering (RPS)	Use the 4-tuple hash for core selection.
Receive Flow Steering (RFS)	Find the core that the application is running on.
Receive Side Steering (RSS)	Hardware version of RPS supported by NICs.
accelerated RFS (aRFS)	Hardware version of RFS supported by NICs.

  - RPS goal: load-balancing
  - RFS goal: reducing CPU context switches

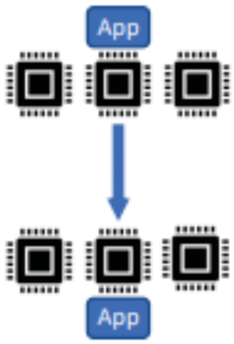
Optimizations

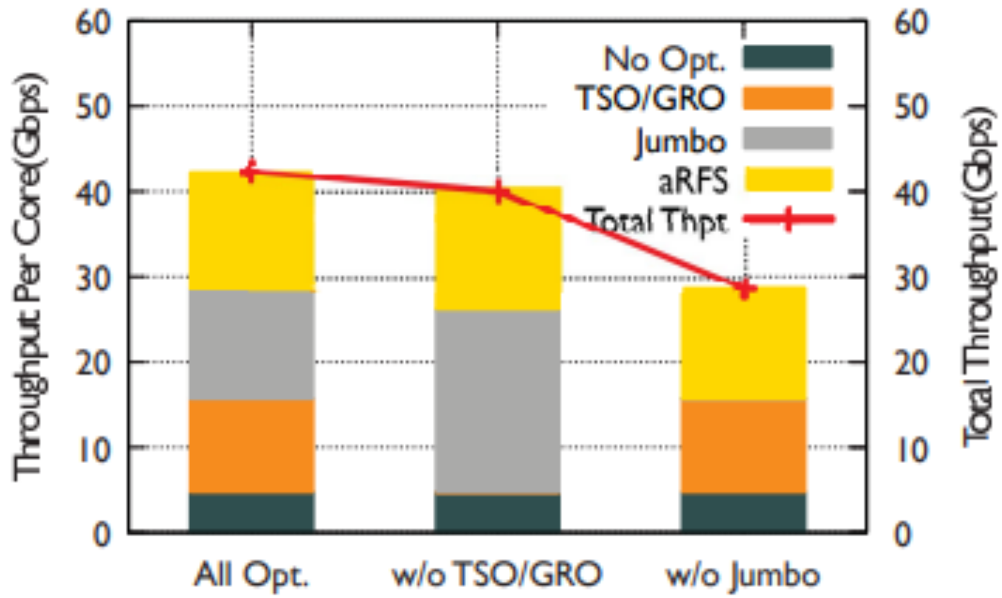
- Segmentation and Receive Offloads
  - Kernel stack processes large 64KB chunks
  - Transmission (tx) side: split chunks into MTU sized (1500 bytes) packets.
    - GSO (generic segmentation offload): splitting in software
    - TSO (TCP segmentation offload): : splitting in NIC
  - Receive (rx) side: aggregate packets into larger chunks
    - GRO (generic receive offload): aggregation in software
    - LRO (large receive offload): aggregation in hardware.

- Jumbo frames: large MTUs (~9000 bytes)

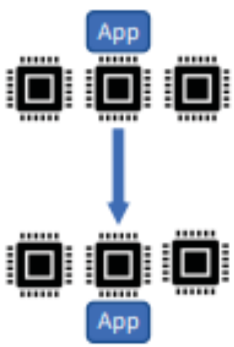
- Zero-copy kernel (not evaluated in the paper)

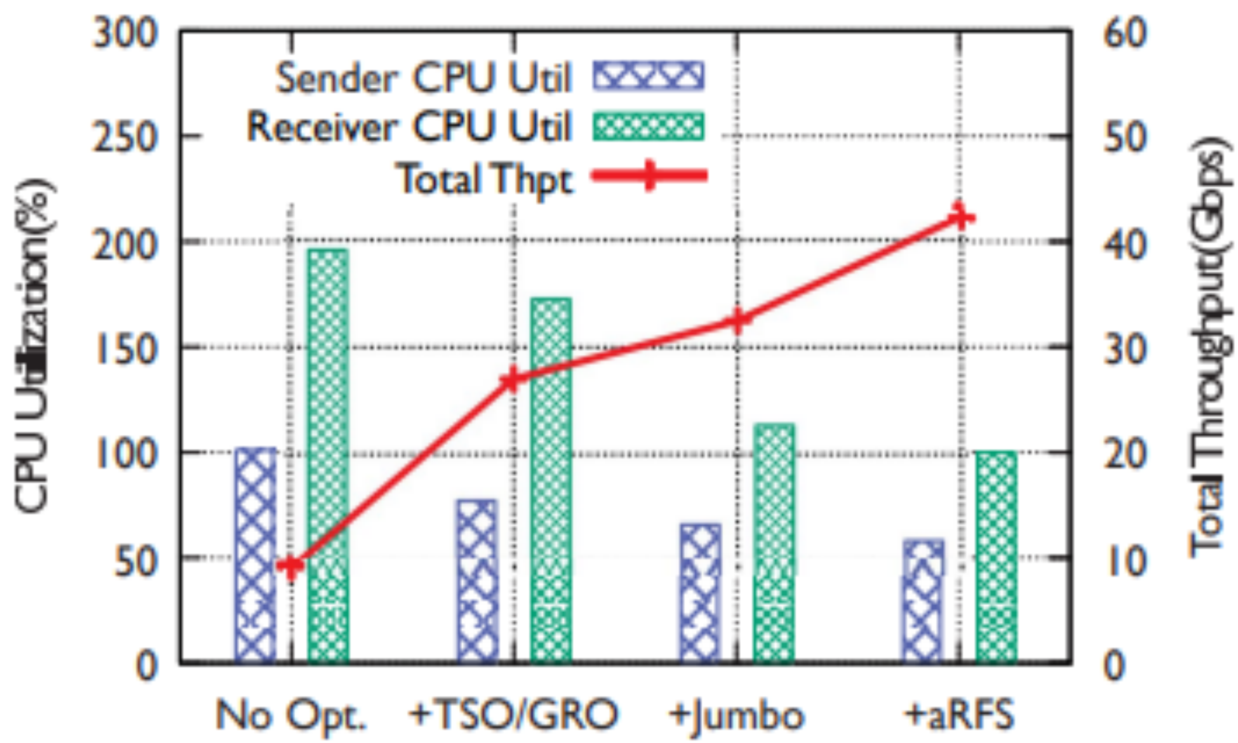
Results


App
App


Throughput Per Core(Gbps)
Total Throughput(Gbps)
60
50
40
30
20
10
0
No Opt.
TSO/GRO
Jumbo
aRFS
Total Thpt
All Opt.
w/o TSO/GRO
w/o Jumbo

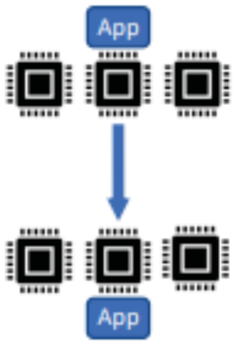
Results

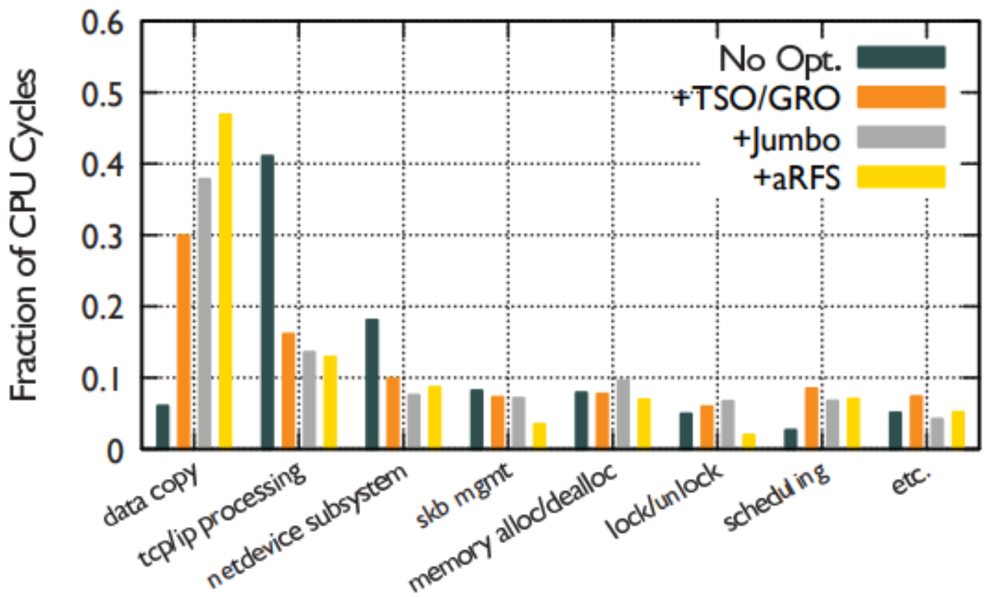




What's the bottleneck?

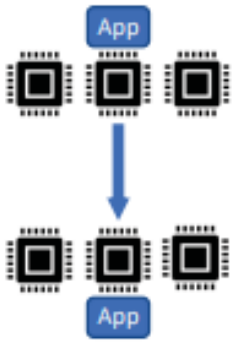
Results

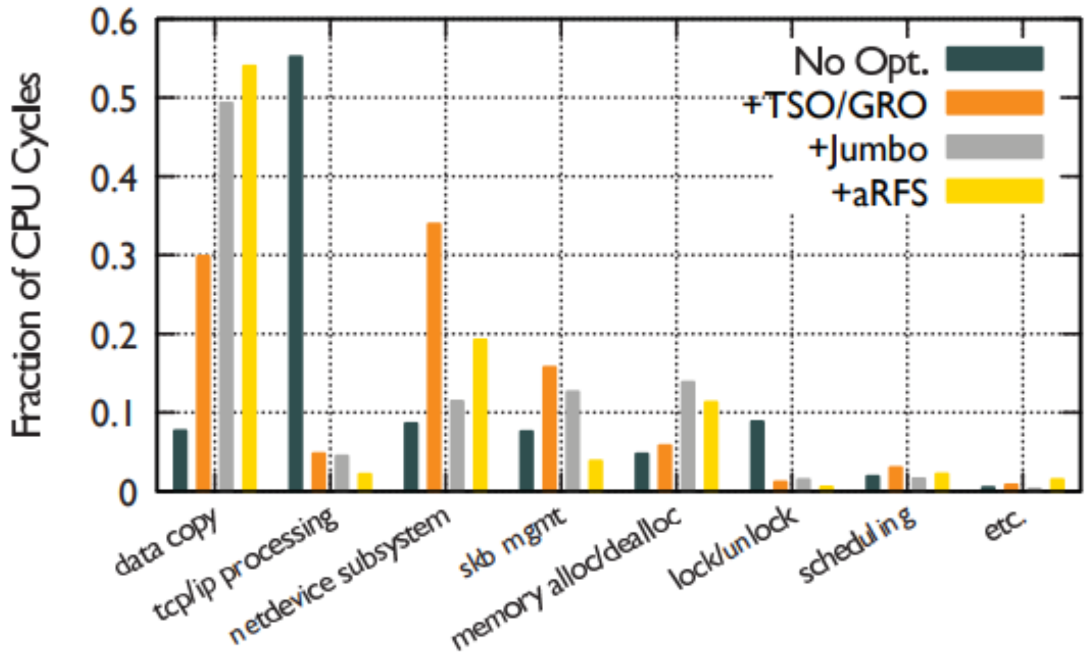




(c) Sender CPU breakdown

Results

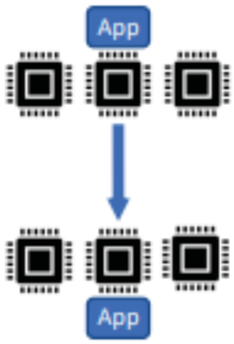


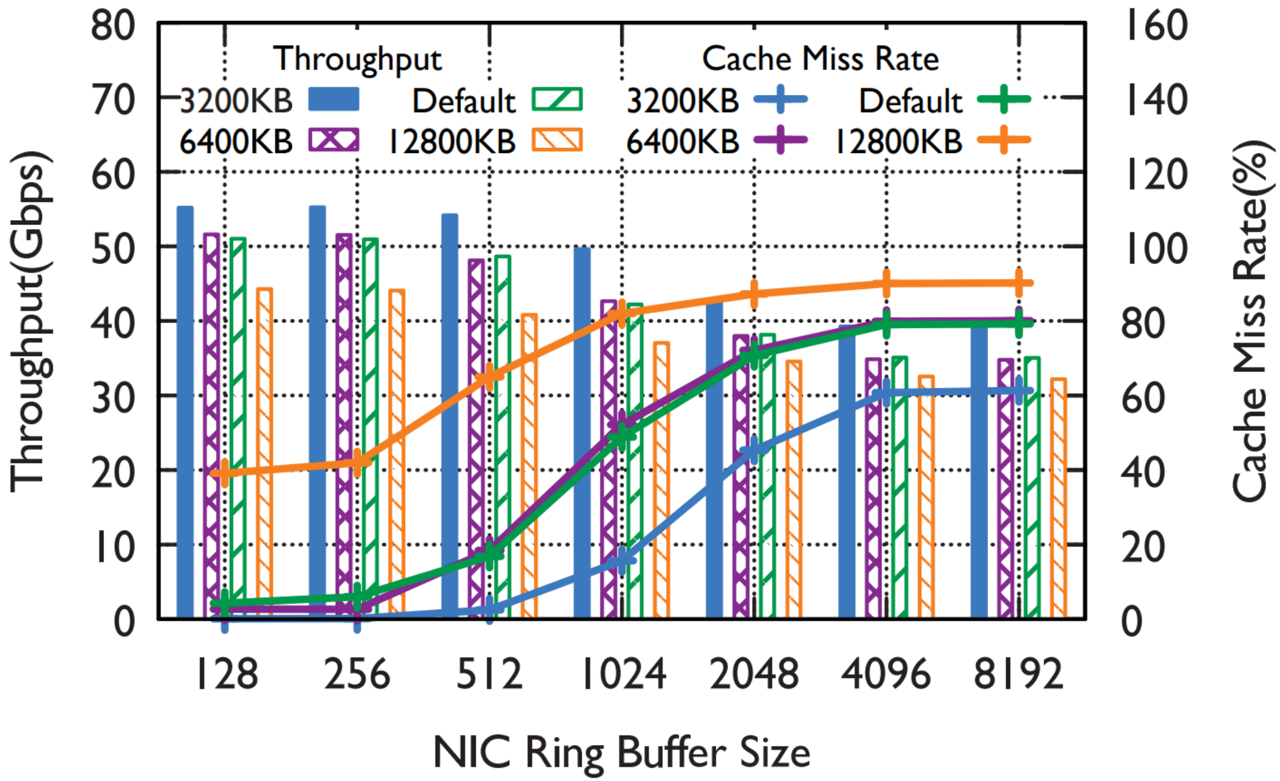


(d) Receiver CPU breakdown

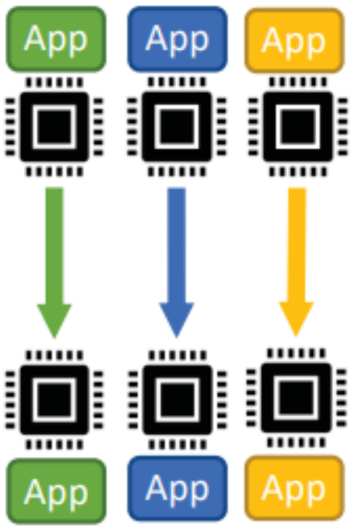
What's the bottleneck if your workload comprises of lots of short flows?

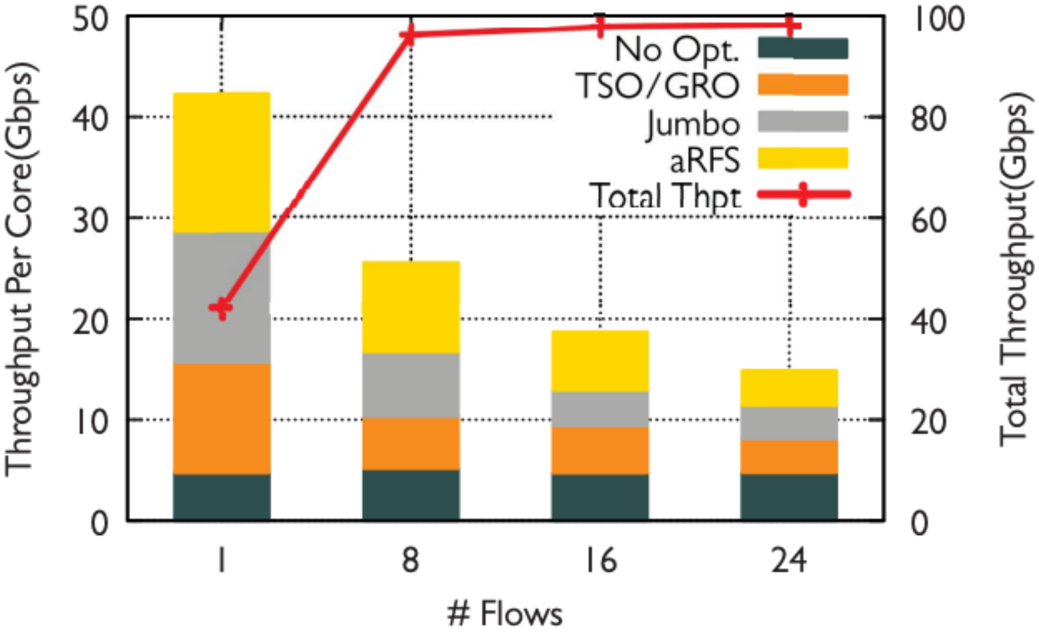
Results


App
App


Throughput
3200KB
6400KB
Default
12800KB
Cache Miss Rate
3200KB
6400KB
Default
12800KB
Throughput(Gbps)
Cache Miss Rate(%)
NIC Ring Buffer Size
0
10
20
30
40
50
60
70
80
0
20
40
60
80
100
120
140
160
128
256
512
1024
2048
4096
8192

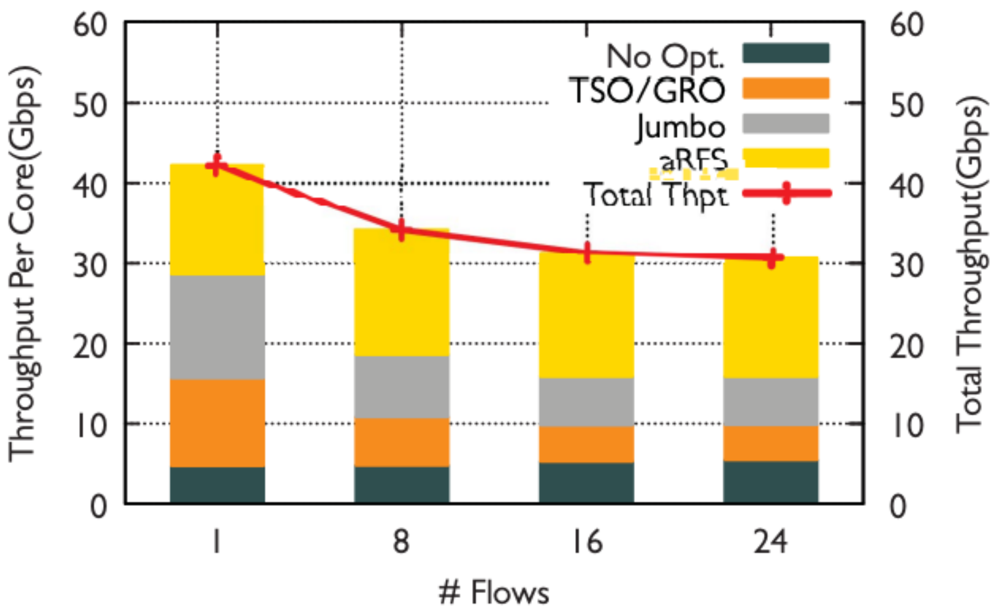
Results





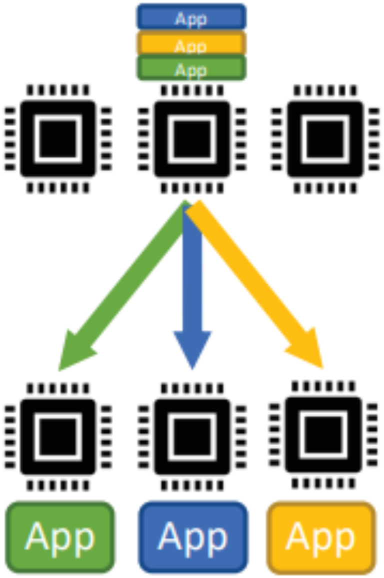
Reasons?

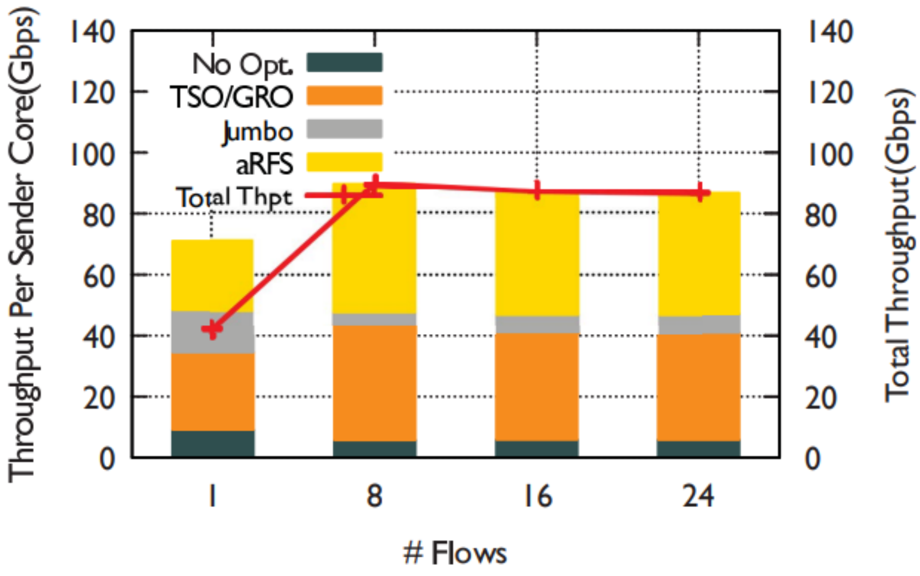
Results



Reasons?

Results

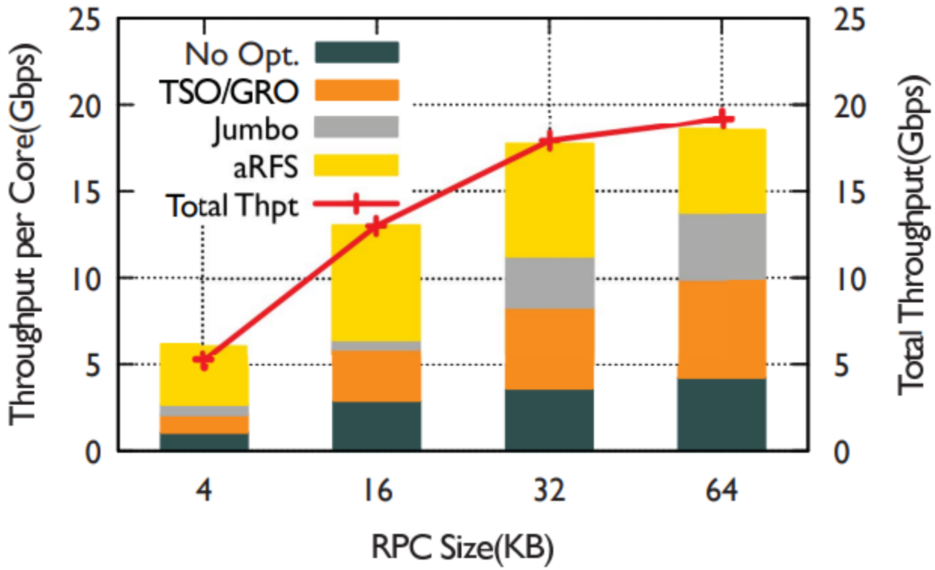




Reasons?

Results

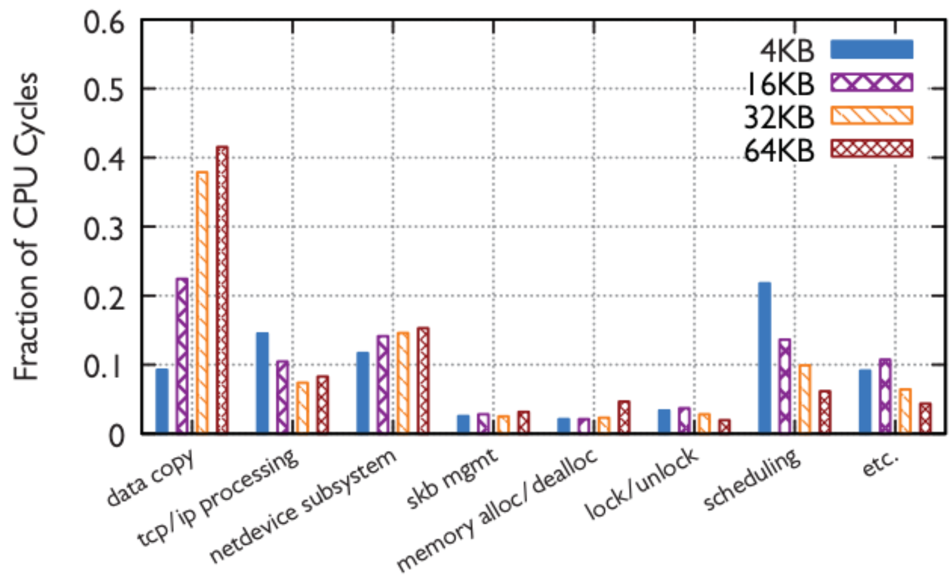
16:1 incast
ping-pong workload
vary flow (RPC) size



Reasons?

Results

16:1 incast
ping-pong workload
vary flow (RPC) size



In following classes

- Kernel bypass
  - Userspace network stacks
- Software bypass
  - RDMA