The Benefits of General-Purpose On-NIC Memory

Boris Pismenny † Liran Liss §

Adam Morrison ‡ Dan Tsafrir †^

† Technion Israel Institute of Technology

‡ TEL AVIV UNIVERSITY

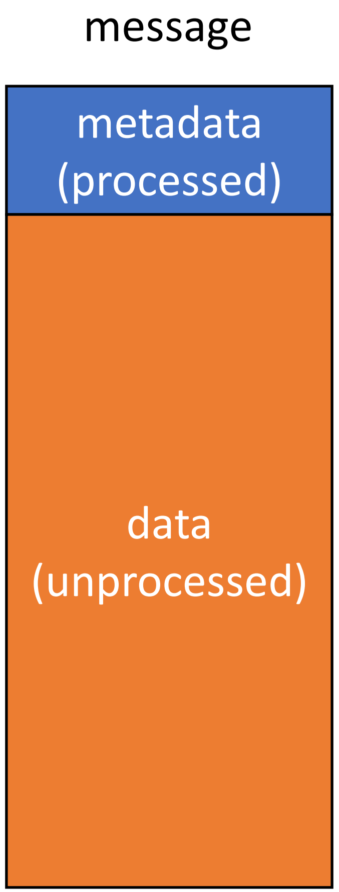
^ vmware RESEARCH

§ NVIDIA

Data movers – definition

Apps that are

1. Network intensive
2. Process message metadata
3. Do not process message data



Data movers – types

1. Apps that process headers but not payload
   - Examples: SW routers, NAT, load balancers, multicast, …

2. Apps that get item key and return item data
   - Examples: key-value stores (Memcached, …), static webservers (Apache, …)

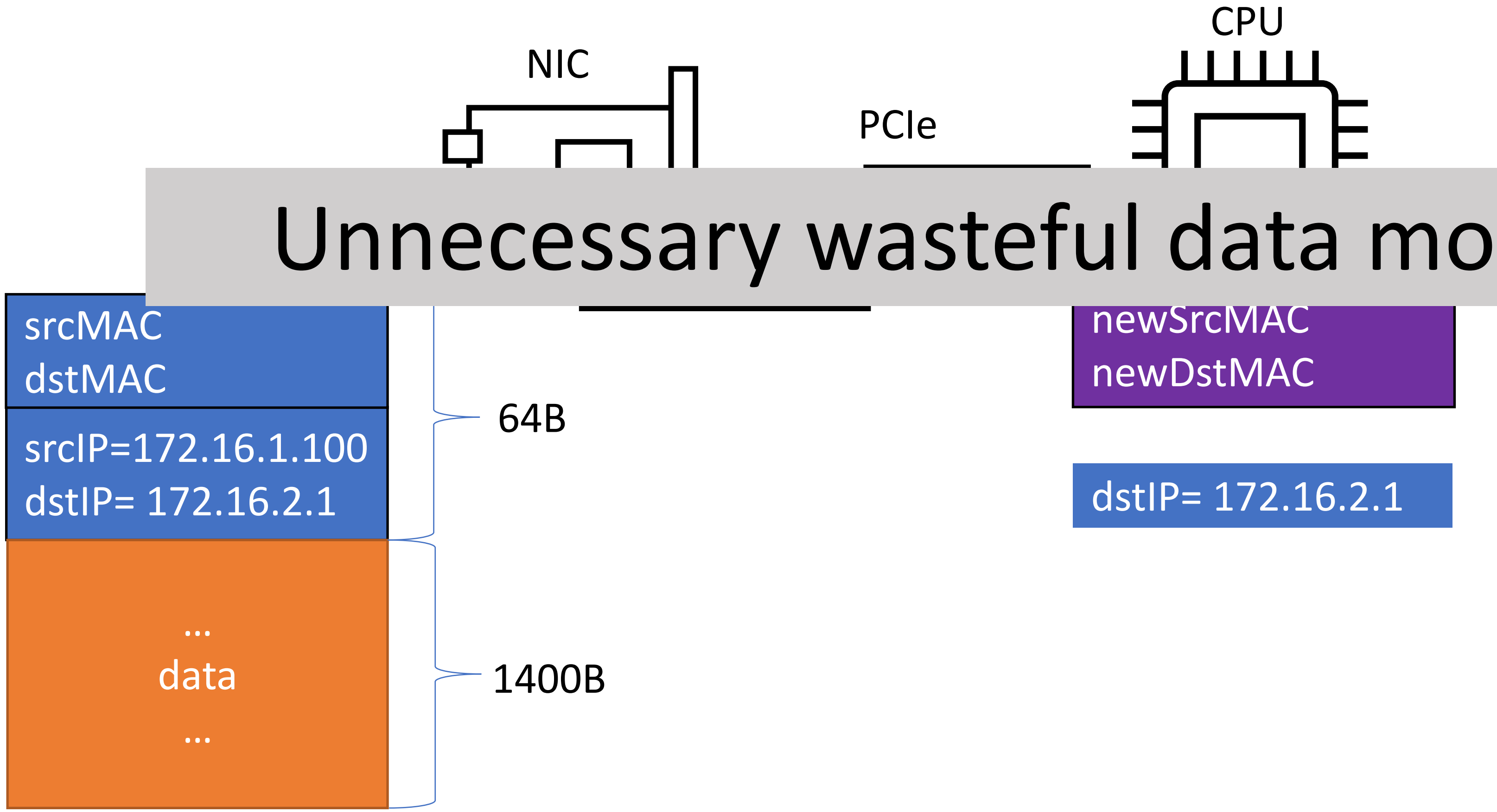
Data movers – types

1. Apps that process headers but not payload
    - Examples: SW routers, NAT, load balancers, multicast, …
2. Apps that associate item key with item data
    - Examples: key-value stores (Memcached, …), static webservers (Apache, …)

This talk is about the first, the second is in the paper

Data movers – cost

Example: software router



Unnecessary wasteful data movement!

Routing table	
Dst IP	Output port
172.16.1.0/24	0
172.16.2.0/24	1
[illegible]	[illegible]
…	…

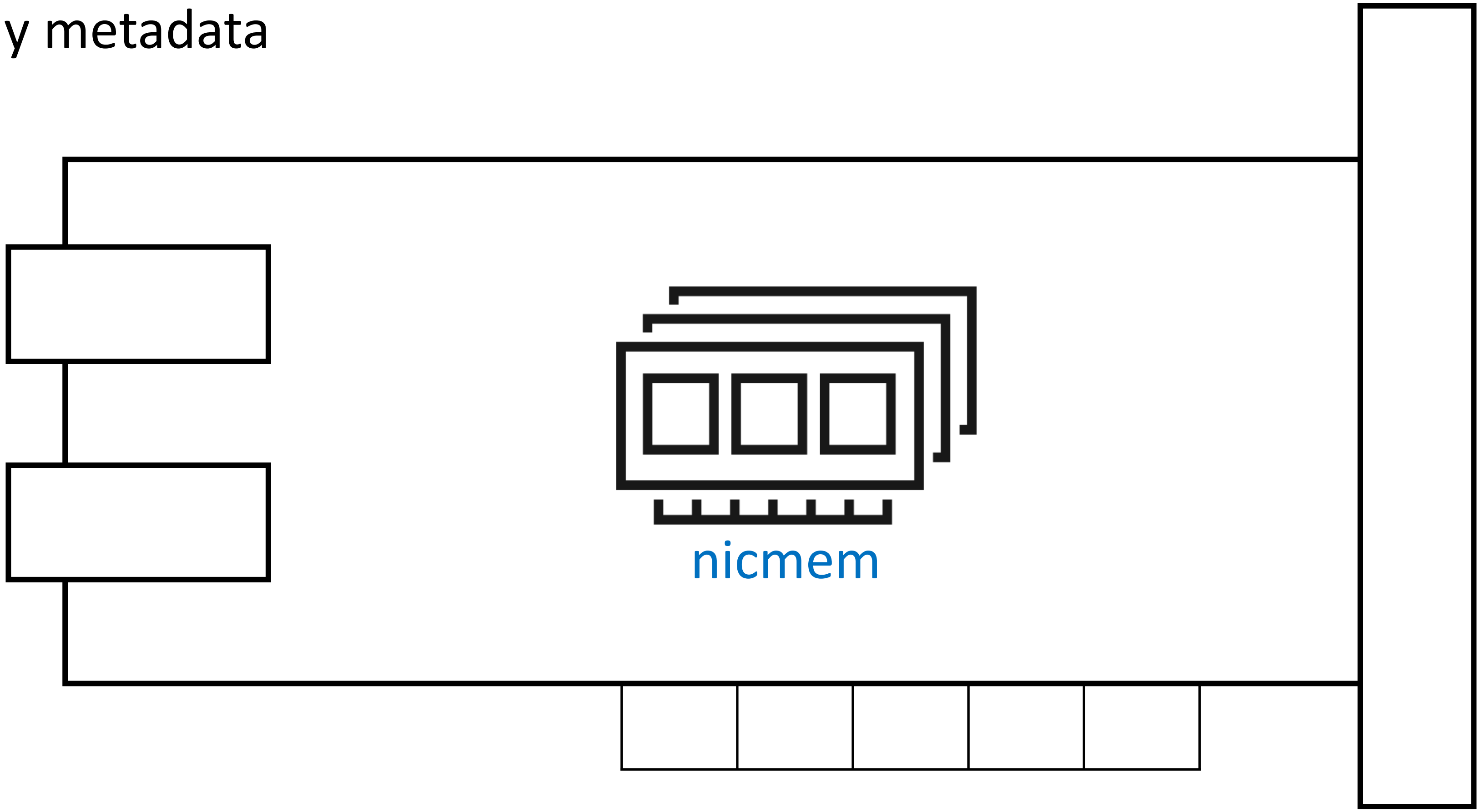
Data movers – cost

Waste

- PCIe bandwidth
- Memory bandwidth
- CPU cycles (if mover isn’t zero-copy)
- LLC space & bandwidth
  - DDIO allows NIC to directly accesses LLC

What we do in a nutshell

- Leave data on nicmem
- Copy only metadata

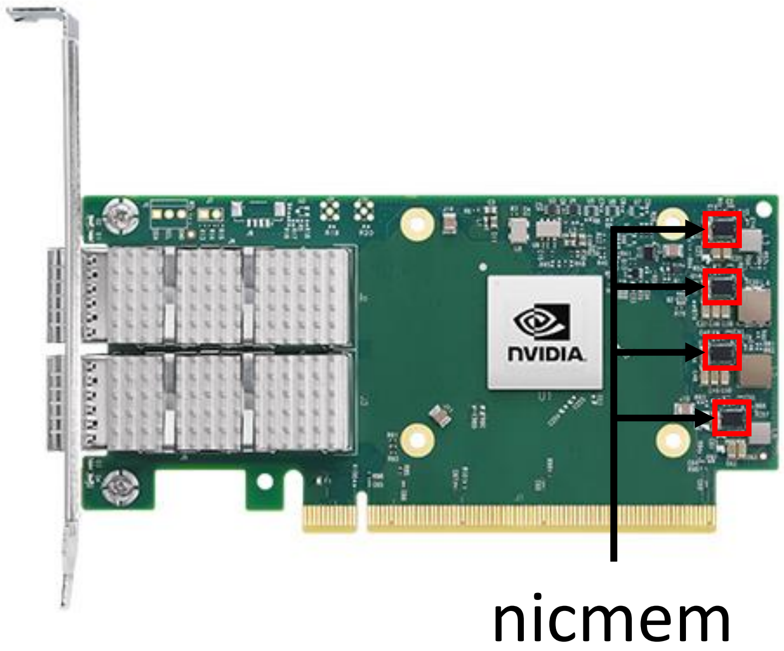


NIC memory (nicmem) today

- Most NICs have internal SRAM memory
  - For stateful offloading
    - RDMA, steering, SRIOV, …
  - Size: few MBs

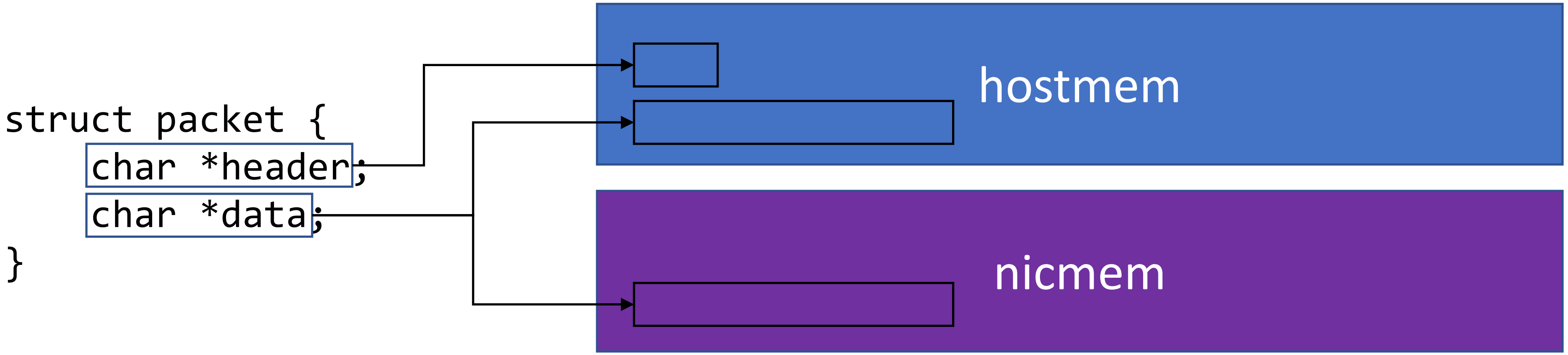
- Nicmem is underutilized
  - Only 15% used by default in recent NVIDIA (Mellanox) NICs

- Nicmem is cheap & can easily be enlarged
  - About 0.2$ per MB at 7nm
  - 3D stacking further reduces area + cost



Nicmem is like regular memory

- Expose nicmem as regular memory
  - MMIO (like GPU frame buffers)
  - Map into process virtual address space
  - Dereference via regular pointers
  - NIC queues can point to nicmem

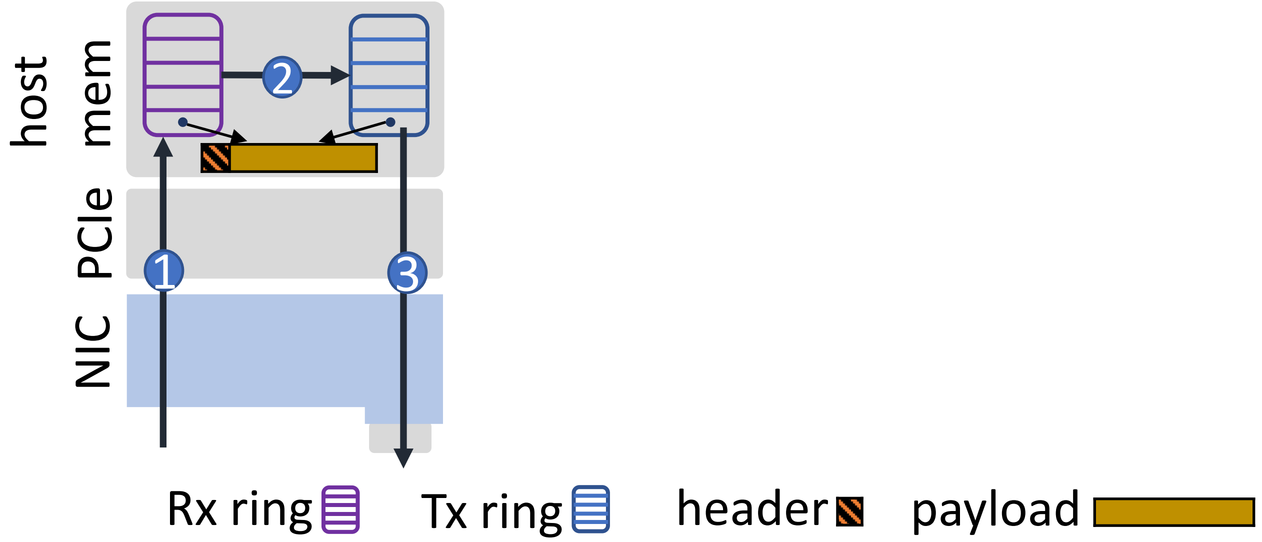


```
struct packet {
    char *header;
    char *data;
}
```

Leveraging Nicmem for NFV

- Baseline: host memory stores header and payload
  1. NIC DMA writes packet
  2. NF processes packet header
  3. NIC DMA reads packet

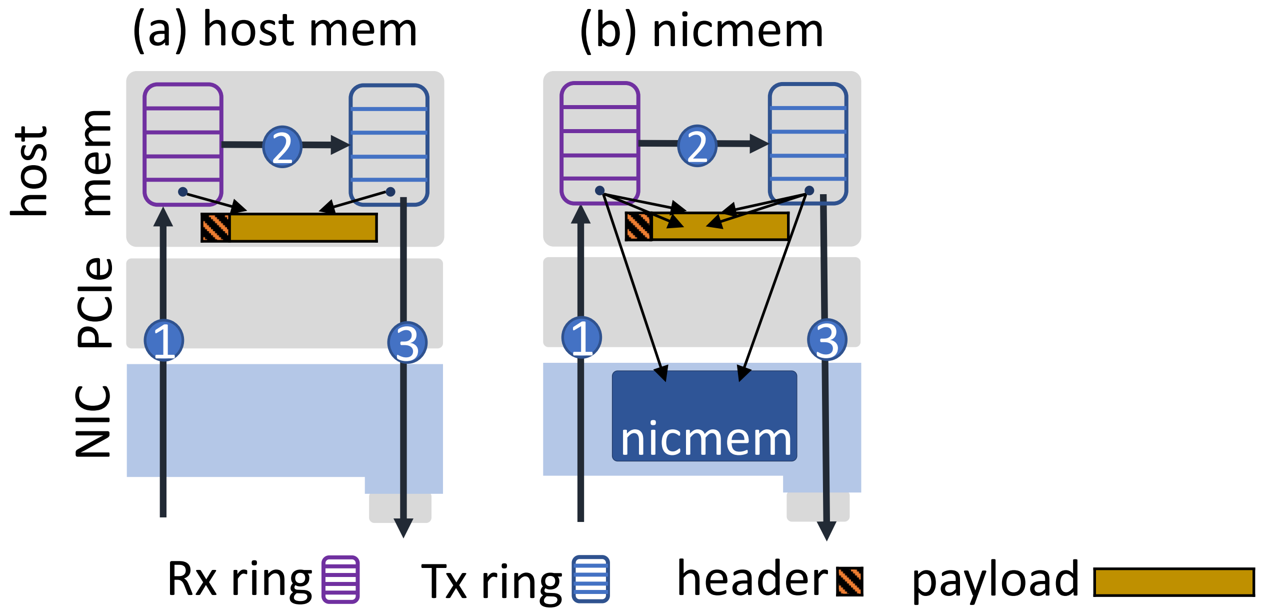
(a) host mem



Leveraging Nicmem for NFV

- Baseline: host memory stores header and payload
  1. NIC DMA writes packet
  2. NF processes packet header
  3. NIC DMA reads packet

- Nicmem
  - Splits header and payload
  - Stores payload on NIC memory

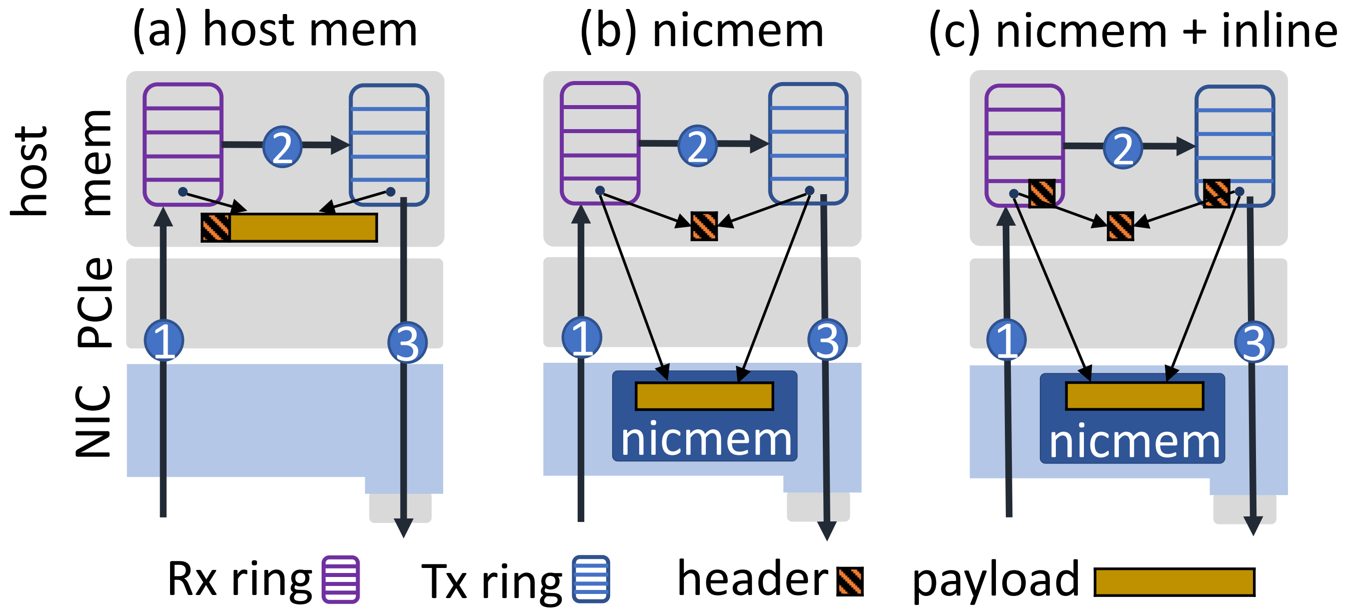


Leveraging Nicmem for NFV

- Baseline: host memory stores header and payload
  1. NIC DMA writes packet
  2. NF processes packet header
  3. NIC DMA reads packet

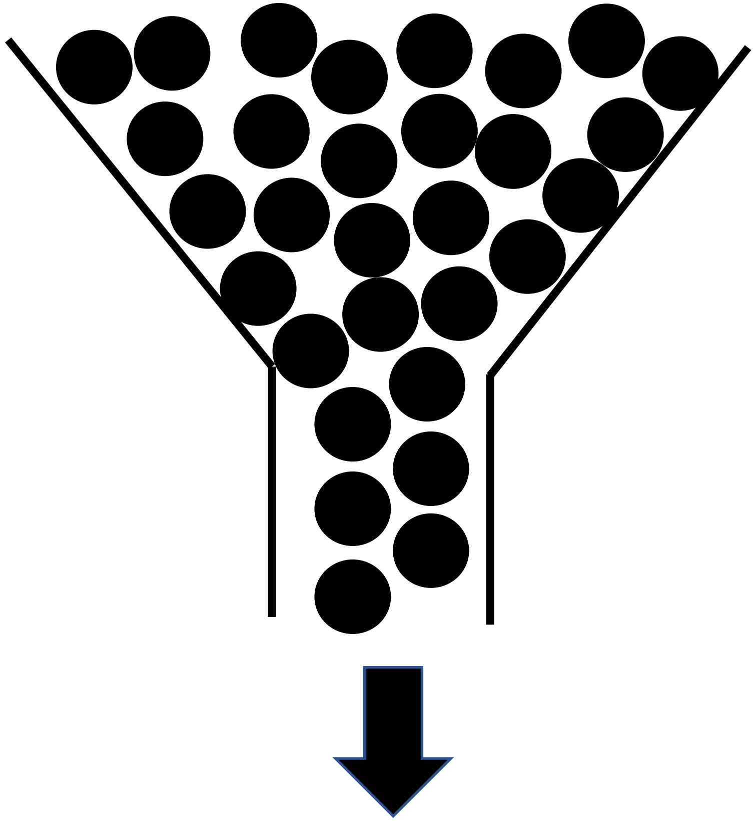
- Nicmem
  - Splits header and payload
  - Stores payload on NIC memory

- Header inlining
  - Write header inside descriptor
  - Back to one descriptor per packet



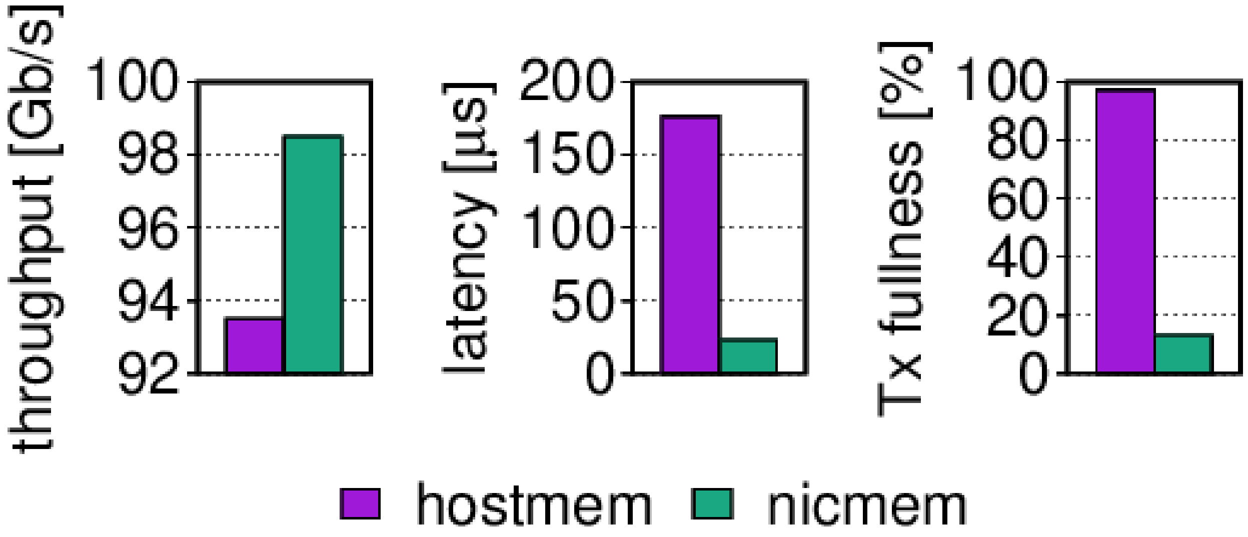
Bottlenecks

- NIC
- PCIe
- Memory bandwidth



Bottleneck: inside the NIC

- NIC Tx queue overflows
- Nicmem avoids the issue

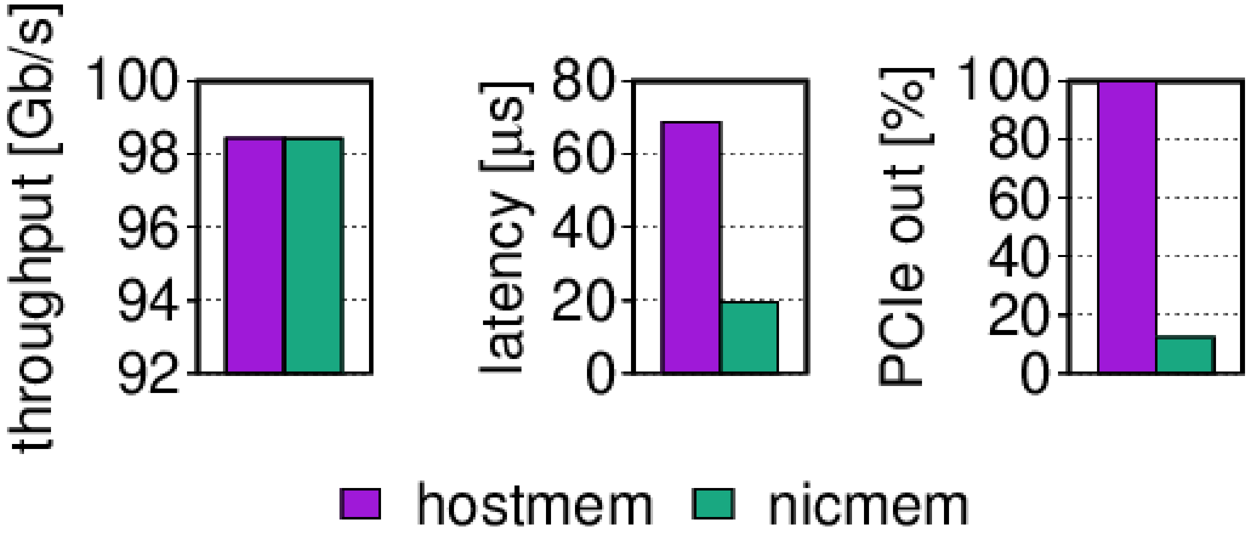


(DPDK l3fwd running on a single core)

Bottleneck: PCIe

- PCIe links towards the host are full
  - Increasing latency by 3x

- Nicmem avoids the issue

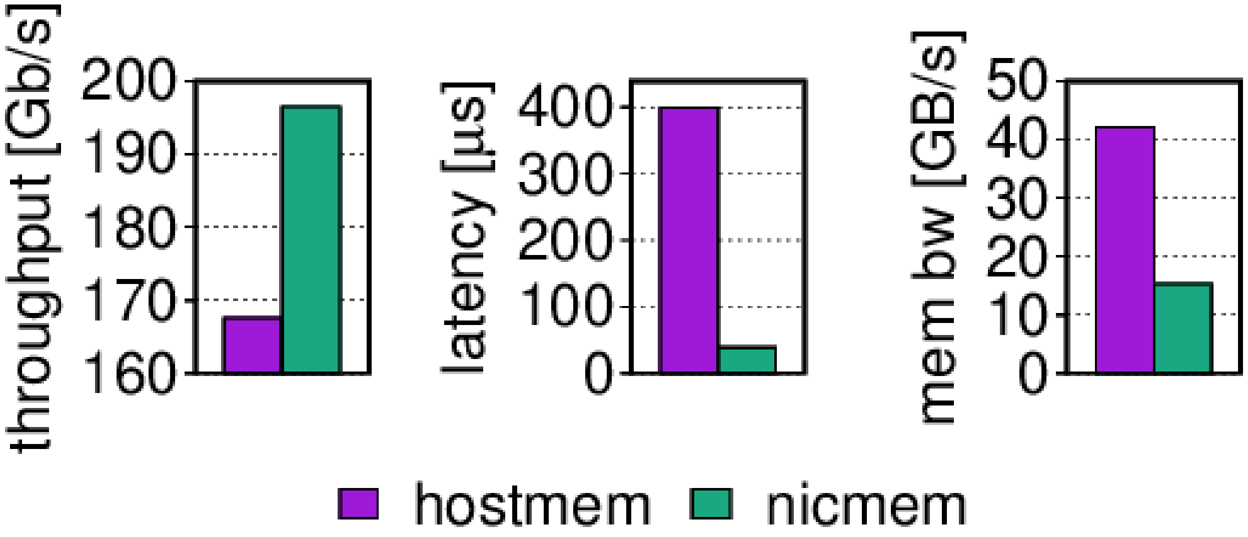


(DPDK l3fwd running on a two cores)

Bottleneck: memory bandwidth

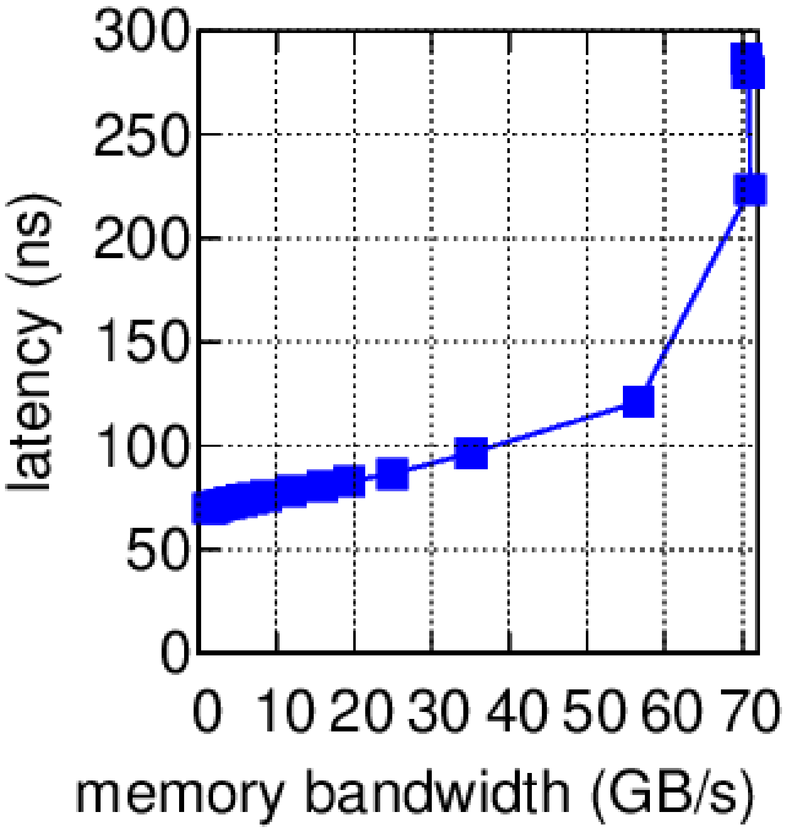
- Memory bandwidth is 2.5x
  - 15% lower throughput
  - 10x higher latency

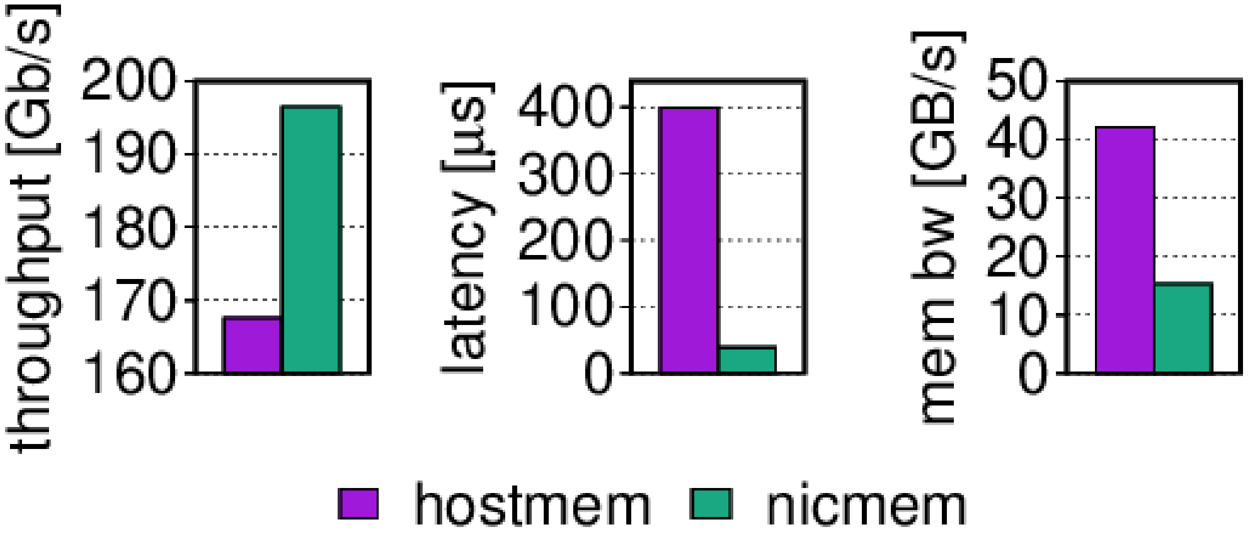
- Nicmem avoids the issue



(DPDK l3fwd running on eight cores)

Bottleneck: memory bandwidth



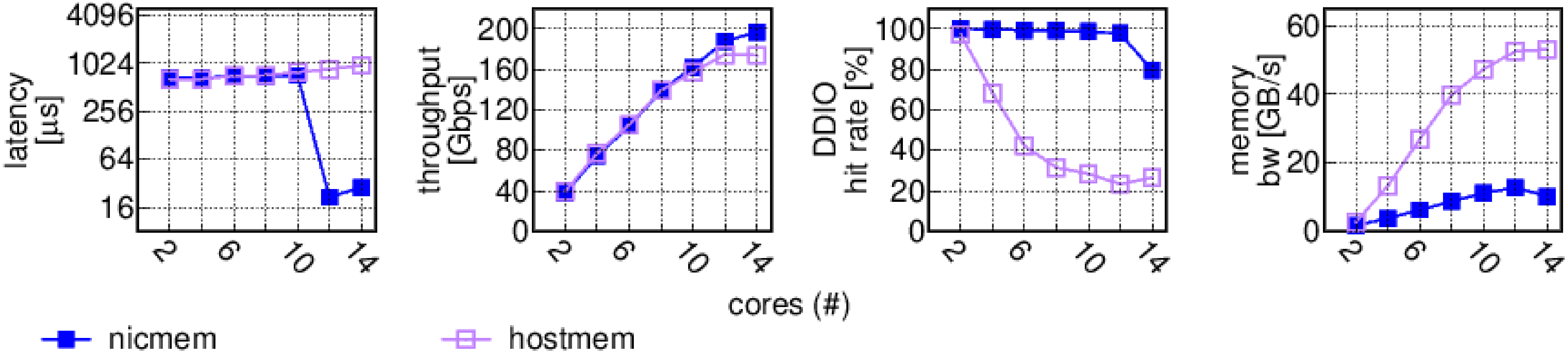


(DPDK l3fwd running on eight cores)

Additional experimental results

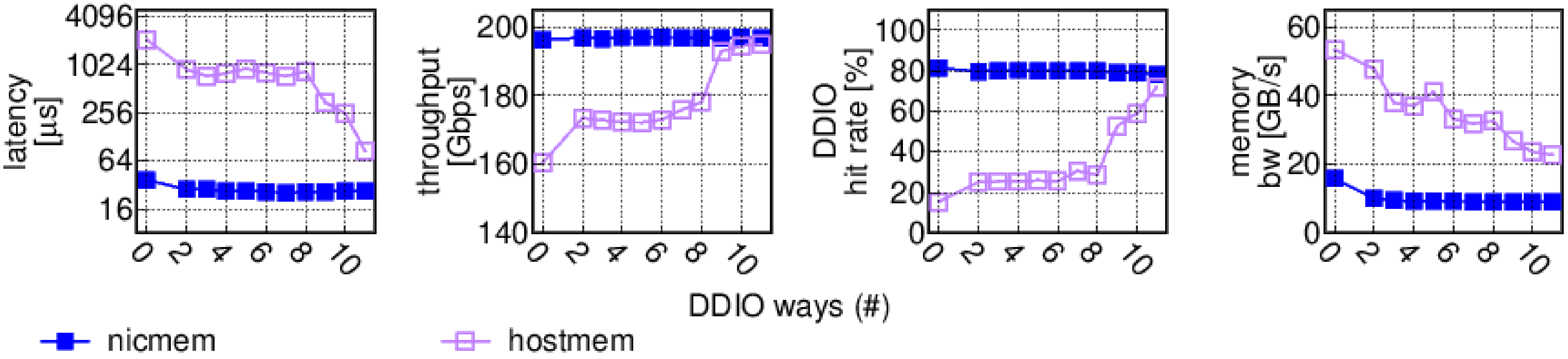
- Nicmem improves scalability
- Nicmem is better than DDIO
- Nicmem outperforms NFV hardware acceleration

Nicmem improves scalability



(FastClick NAT loaded with 200Gbps)

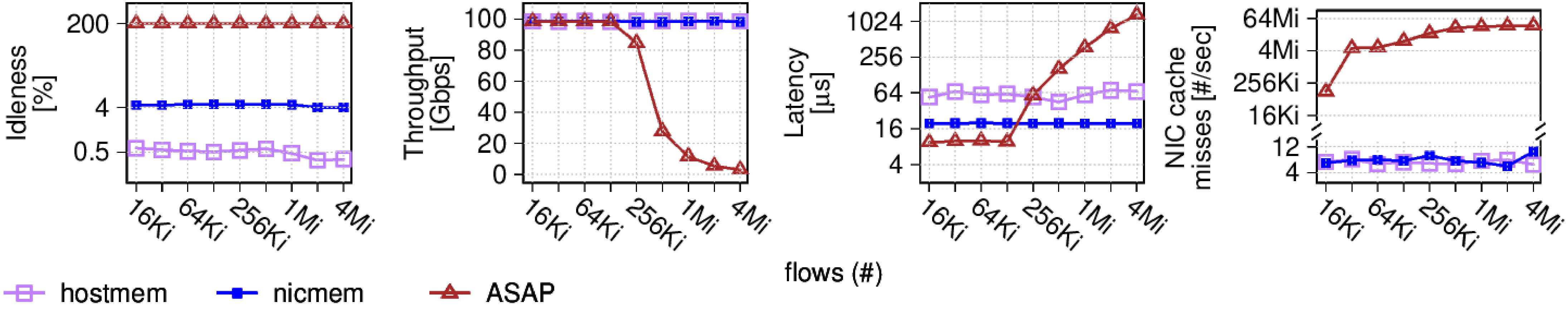
Nicmem reduces DDIO use



(FastClick NAT running on 14 cores and loaded with 200Gbps)

Nicmem is preferrable to NIC acceleration

- NIC memory can be used by
  - Software as nicmem; or
  - Hardware for per-flow acceleration state
- NIC acceleration eliminates CPU overhead
  - But it doesn't scale



(DPDK per-flow packet and byte counters running on 2 queues)

Conclusion

- Nicmem benefits data-mover applications
- Nicmem eliminates NIC, PCIe, and memory bandwidth bottlenecks
- Nicmem complements DDIO and outperform NFV acceleration in hardware

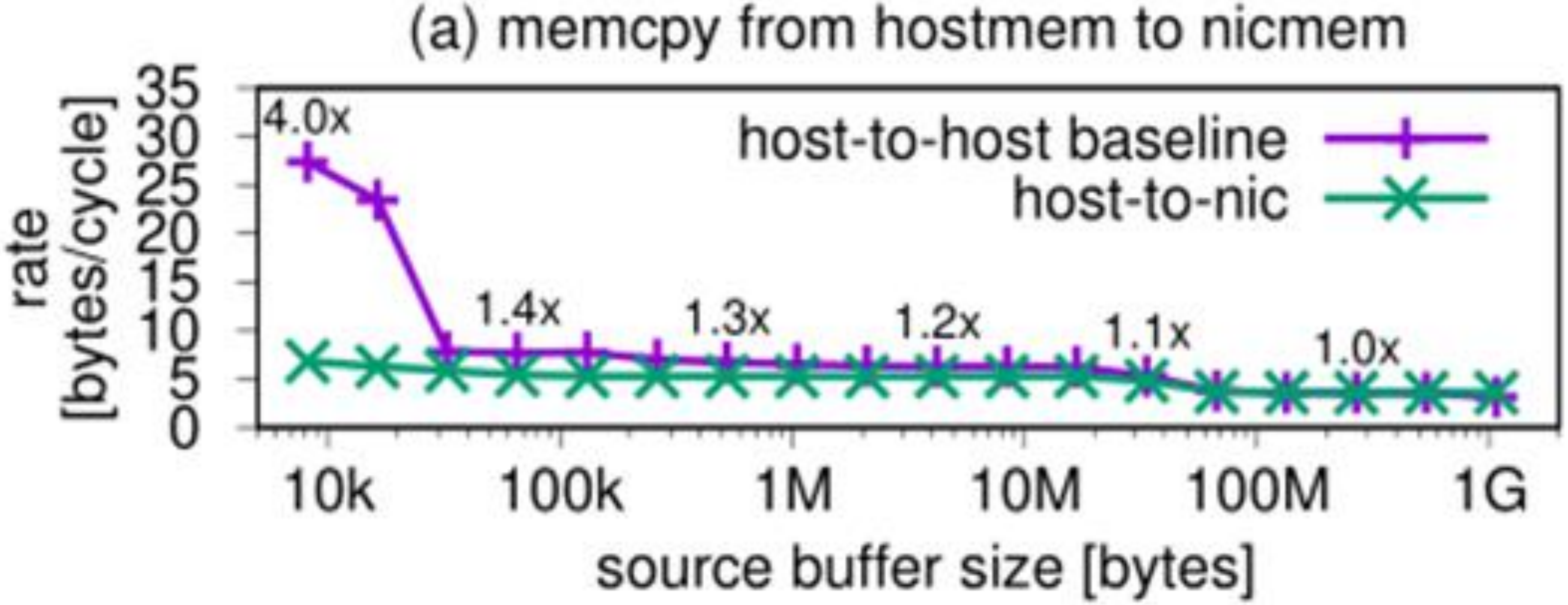
Conclusion

- Nicmem benefits data-mover applications
- Nicmem eliminates NIC, PCIe, and memory bandwidth bottlenecks
- Nicmem complements DDIO and outperform NFV acceleration in hardware

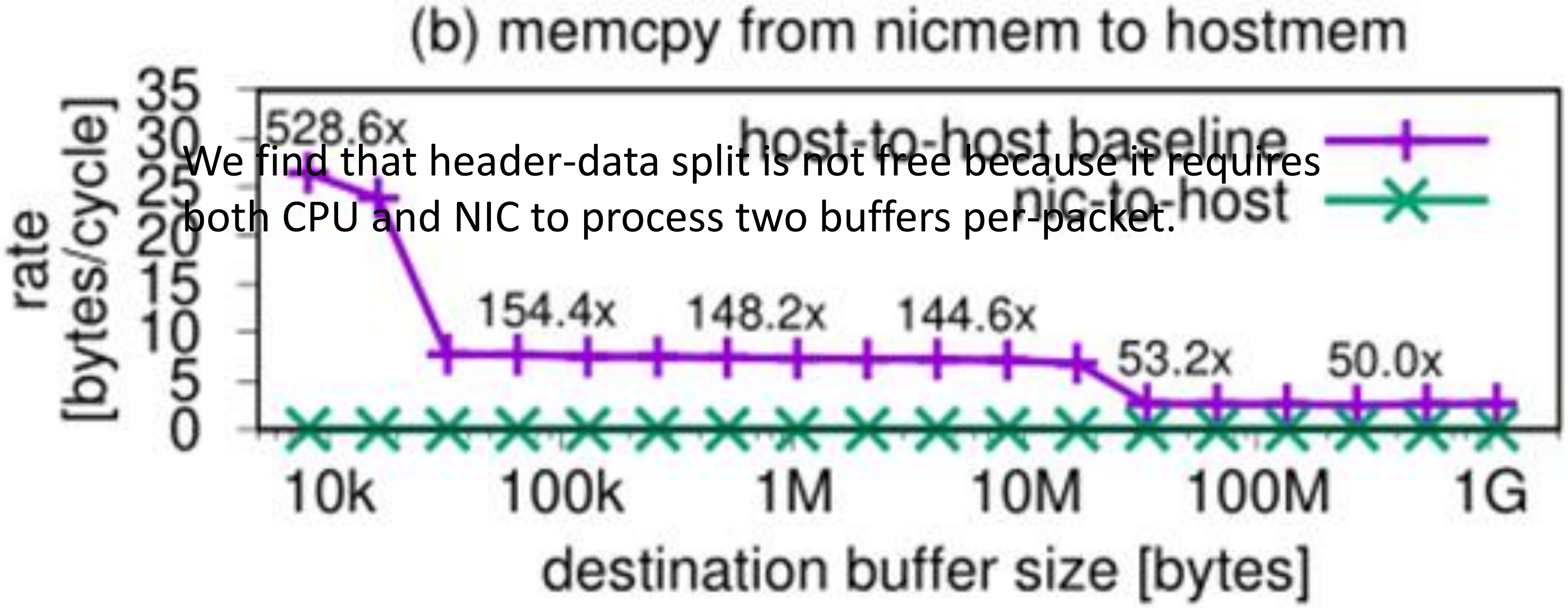
Have any question? Send me an email

Boris Pismenny: borispi@cs.technion.ac.il

Non-data mover applications (1)



Non-data mover applications (2)



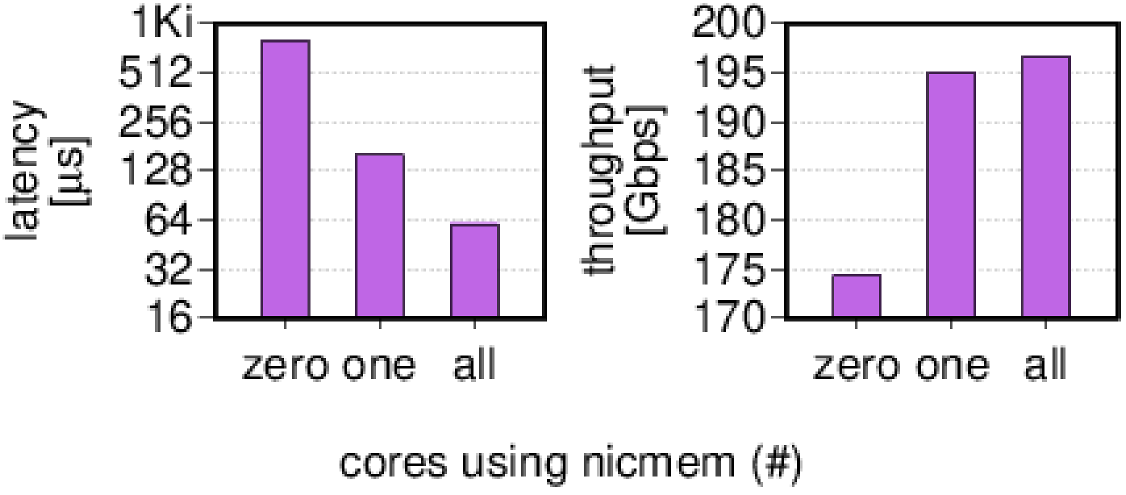
We find that header-data split is not free because it requires both CPU and NIC to process two buffers per-packet.

Practical considerations

- Today’s nicmem is small
  - Each core’s queue is 1.5MB

- Single nicmem queue eliminates the PCIe bottleneck

- All nicmem queues reduces memory bandwidth



(FastClick NAT running on 14 cores with 200Gbps)