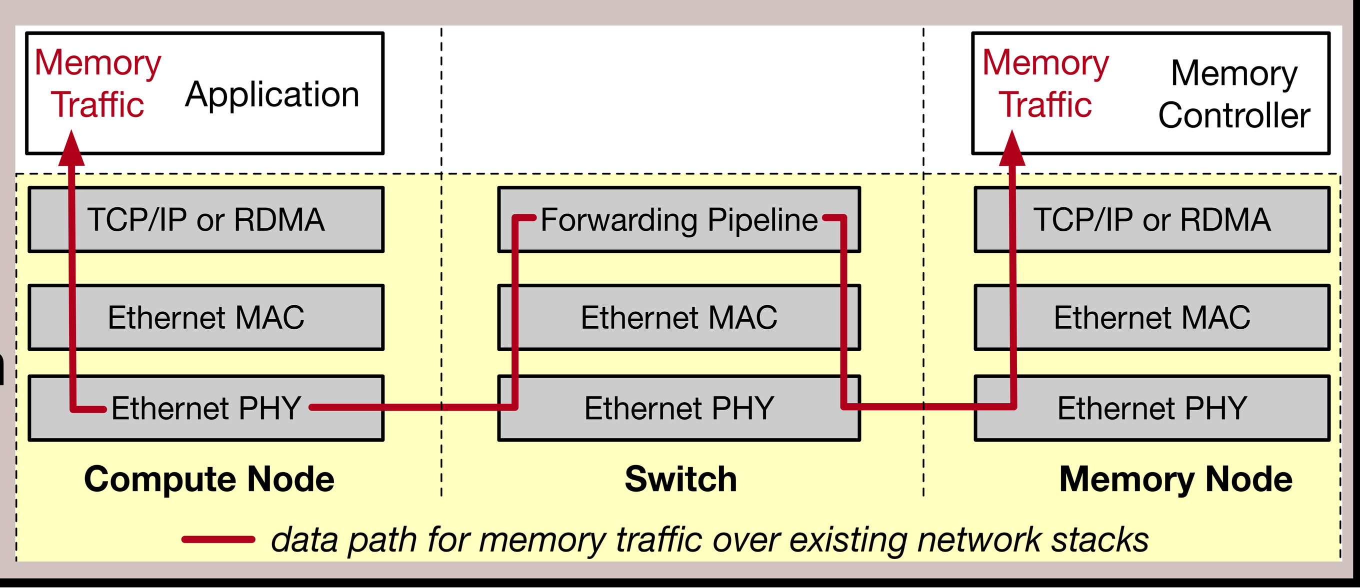
Rack-Scale Memory Disaggregation over Ethernet

Weigao Su, Vishal Shrivastav

PURDUE UNIVERSITY

Memory disaggregation over Ethernet

- Using Ethernet fabric
- High compute density
- Fine-grained provision
- Seamless scaling

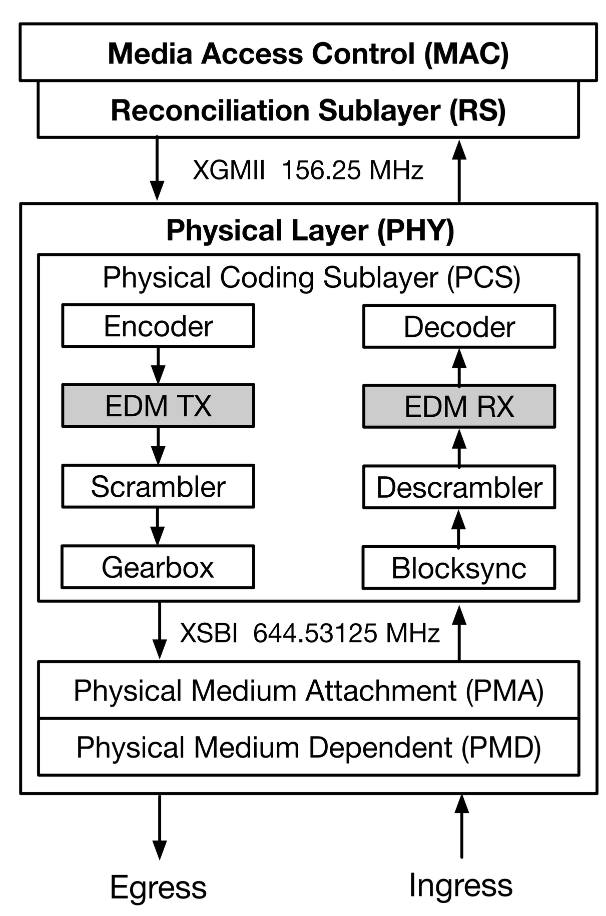


Requirements

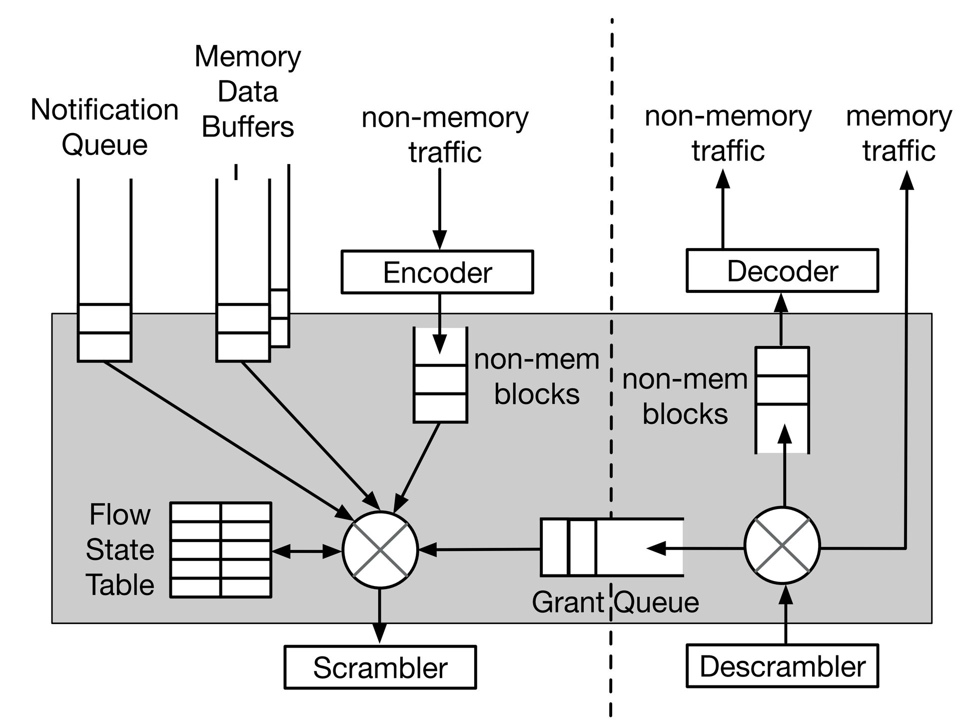
- Latency: Transmission delay needs to be close to local memory access (NUMA takes around 300ns)
- Utilization: Header encapsulation needs to be efficient since memory flows are extremely small, often less than 64B and potentially a single byte.

Existing Limitation

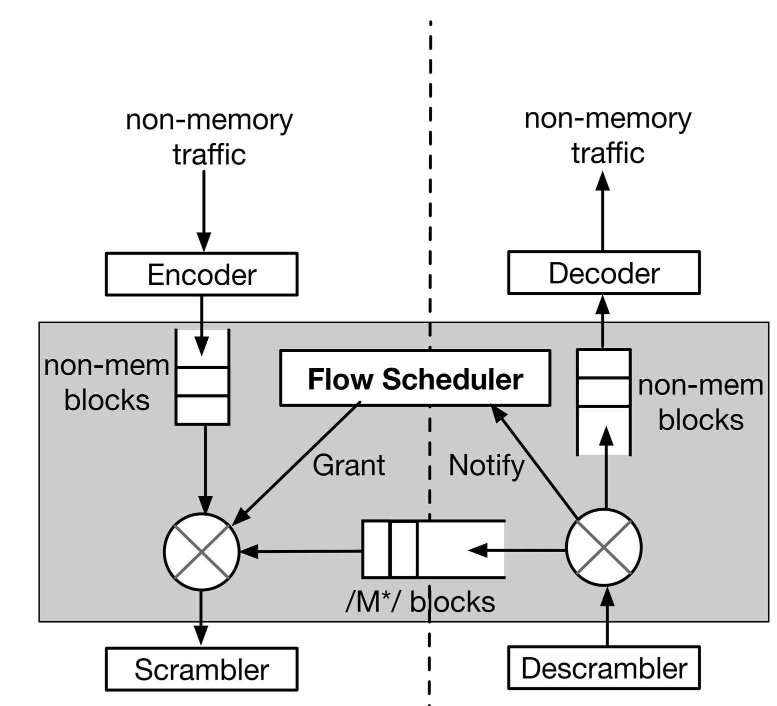
1. Minimum frame size overhead
2. Inter-frame gap (IFG) overhead.
3. No intra-frame preemption.
4. Layer 2 switching overhead.
5. Transport layer overhead.
6. Queueing delay at switch



(a) Ethernet Stack



(b) EDM Host Network Stack



(c) EDM Switch Network Stack

FPGA Prototype

Added latency:

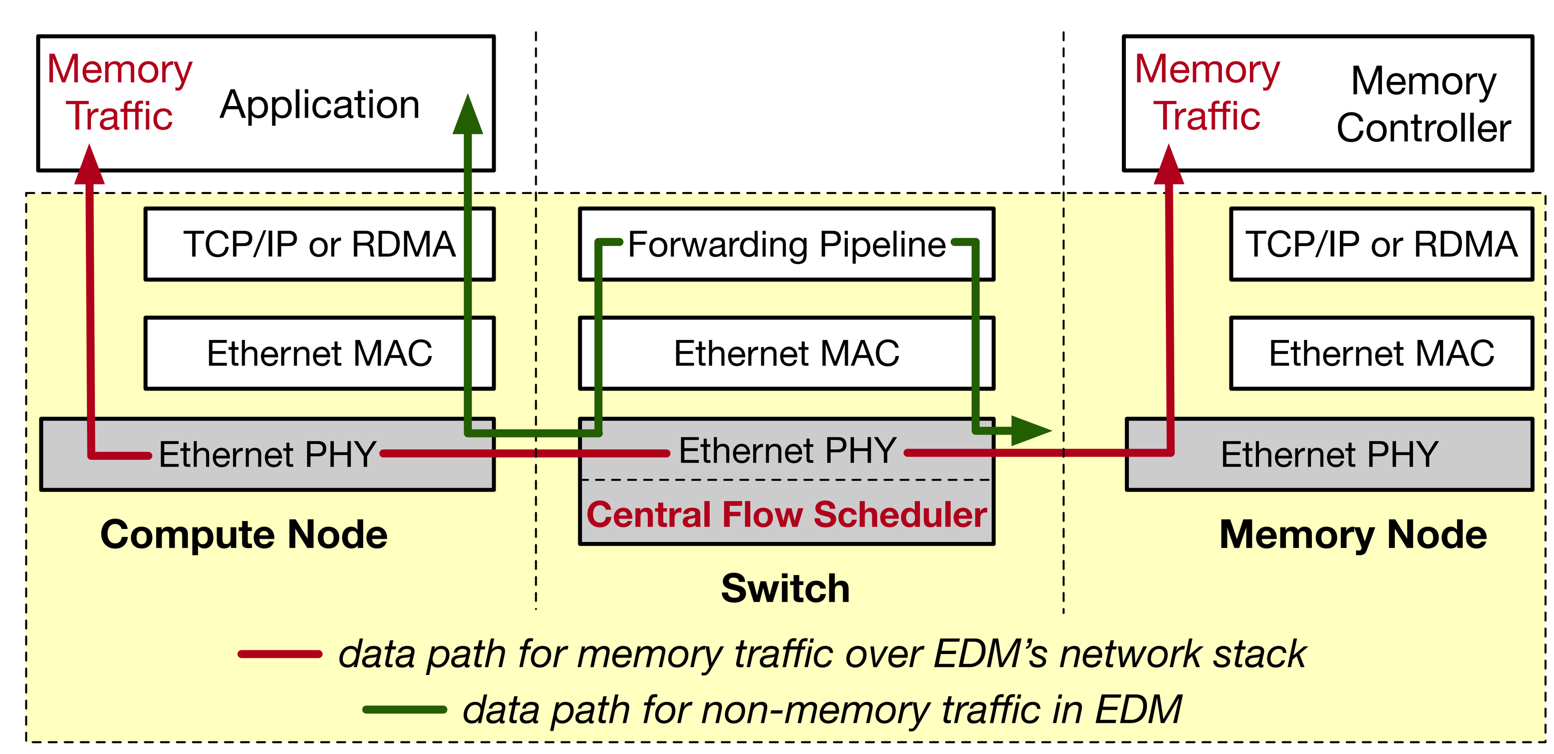
268.8ns for read;

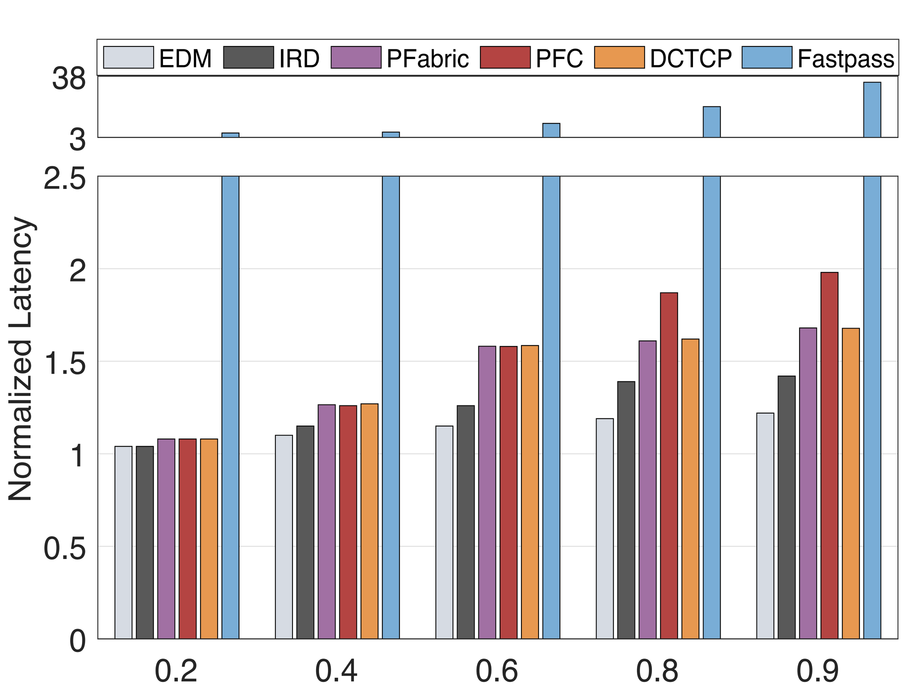
262.4ns for write

Comparable to one hop NUMA; 4x faster than the raw Ethernet.

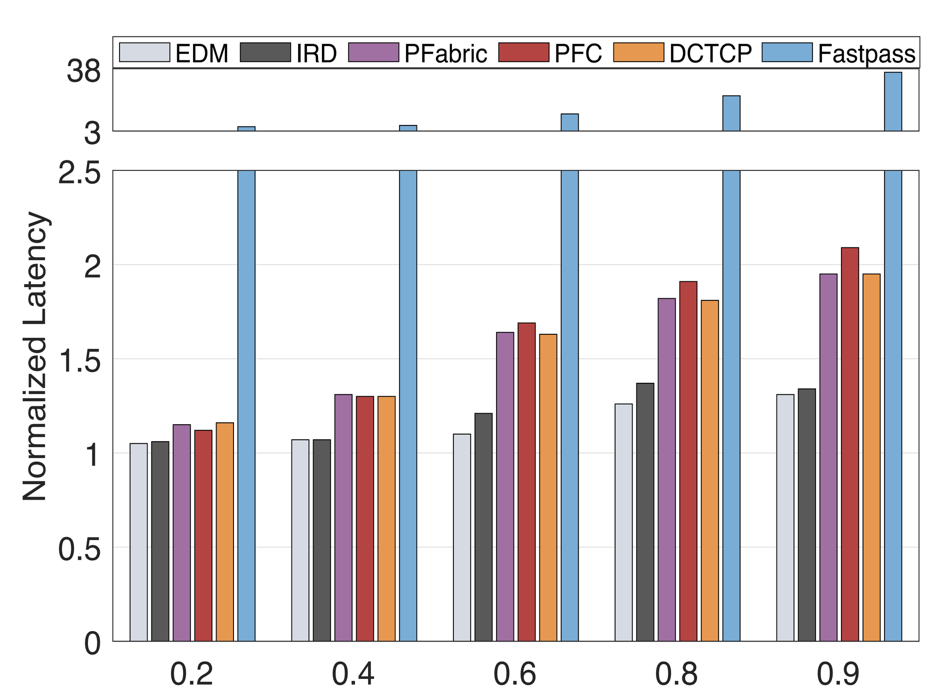
Network Simulation

EDM keeps the end-to-end latency within **1.2x** and **1.3x** the ideal unloaded latency for READ and WRITE requests respectively.





READ



Write

Latency Source	Raw Ethernet		EDM	
	READ	WRITE	READ	WRITE
Compute Node				
MAC	2 * 19.2ns	19.2ns	0	0
PHY (PCS)	2 * 19.2ns	19.2ns	2 * 12.8 + 32ns	3 * 12.8 + 70.4ns
Switch				
Layer 2 fwd	2 * 400ns	400ns	0	0
MAC	4 * 19.2ns	2 * 19.2ns	0	0
PHY (PCS)	4 * 19.2ns	2 * 19.2ns	4 * 12.8 +70.4ns	4 * 12.8 +70.4ns
Memory Node				
MAC	2 * 19.2ns	19.2ns	0	0
PHY (PCS)	2 * 19.2ns	19.2ns	2 * 12.8 +64ns	12.8 +19.2ns
Network Stack Latency	**1.11μs**	**553.6ns**	**268.8ns**	**262.4ns**
Transmission Delay	4 * 51.2ns	2* 51.2ns	6.4 + 51.2ns	12.8+51.2ns
Propagation Delay	4 * 10ns	2 * 10ns	4 * 10ns	4 * 10ns
Total Latency	1.35μs	676ns	366.4ns	366.4ns

EDM (Ethernet Disaggregated Memory)

- Centralized flow scheduling:

  Queuing avoidance & High bandwidth utilization.

  ✦ Maximal-matching: At most one sender sending to a receiver.

  ✦ Zero-delay forwarding: No packet processing pipeline needed.

  ✦ Reduced transport overhead: A no-loss environment guaranteed by maximal-matching.

  ✦ Near-optimal flow completion time: Achieved by a configurable priority queue.

- Bypassing higher layers:

  Low latency & Minimal encapsulation overhead

  ✦ Host: Tx interacts with application through *notification queue*, while Rx asynchronously updates *grant queue* for responding remote requests. States are stored in a *flow state table*.

  ✦ Switch: maintains a notification (priority) queue to proactively avoid congestion and shape traffic.