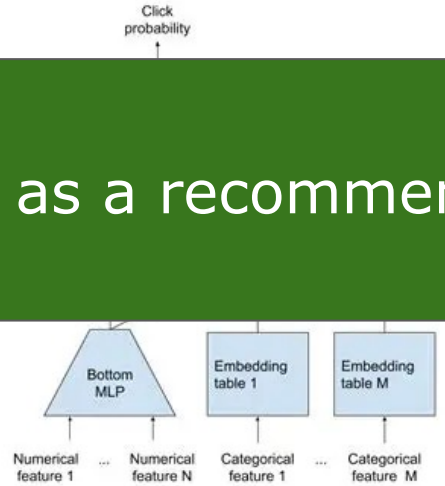
TRACI: Network Acceleration of Input-Dynamic Communication for Large-Scale Deep Learning Recommendation Model

Guyue Huang[1], Hao Li[2], Le Qin[3], Jiayi Huang[3], Yangwook Kang[4], Yufei Ding[5], Yuan Xie[6]

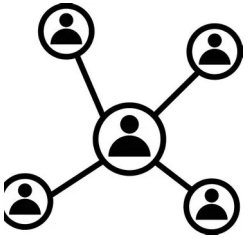
[1] University of California, Santa Barbara
[2] University of Minnesota
[3] The Hong Kong University of Science and Technology (Guangzhou)
[4] Samsung Electronics
[5] University of California, San Diego
[6]The Hong Kong University of Science and Technology

- Search Engine

Deep Learning Recommendation Model as a recommender

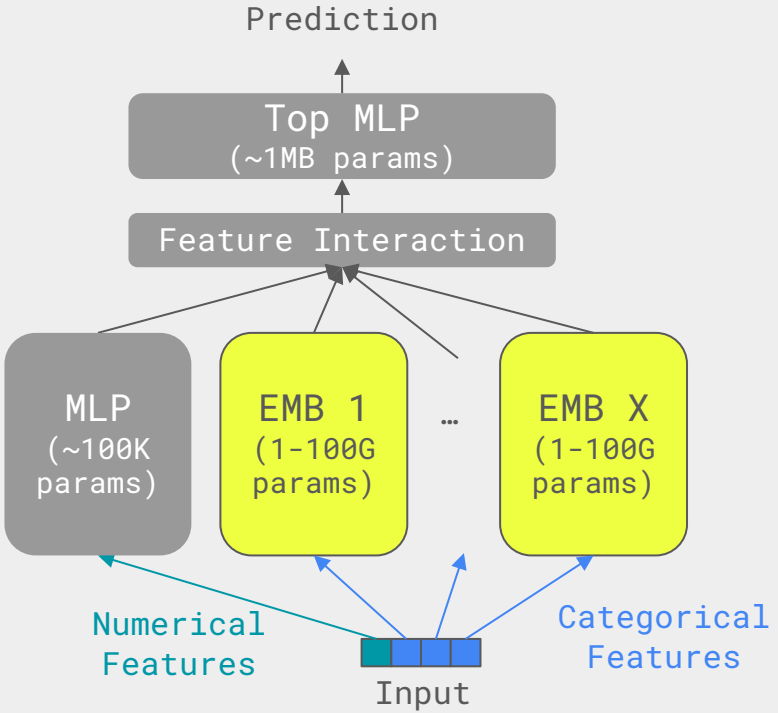


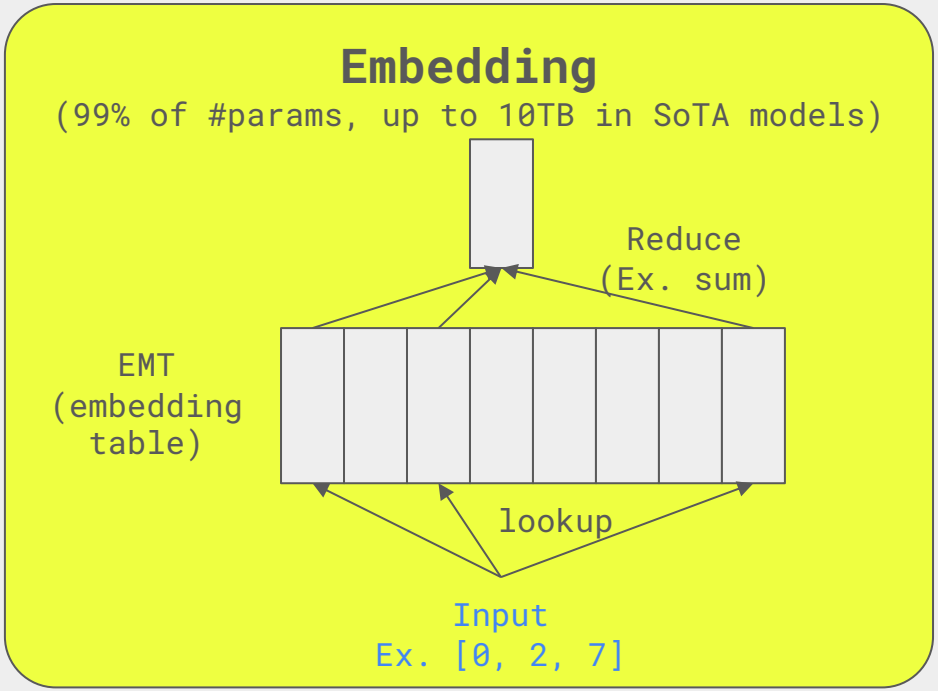
- Social Network

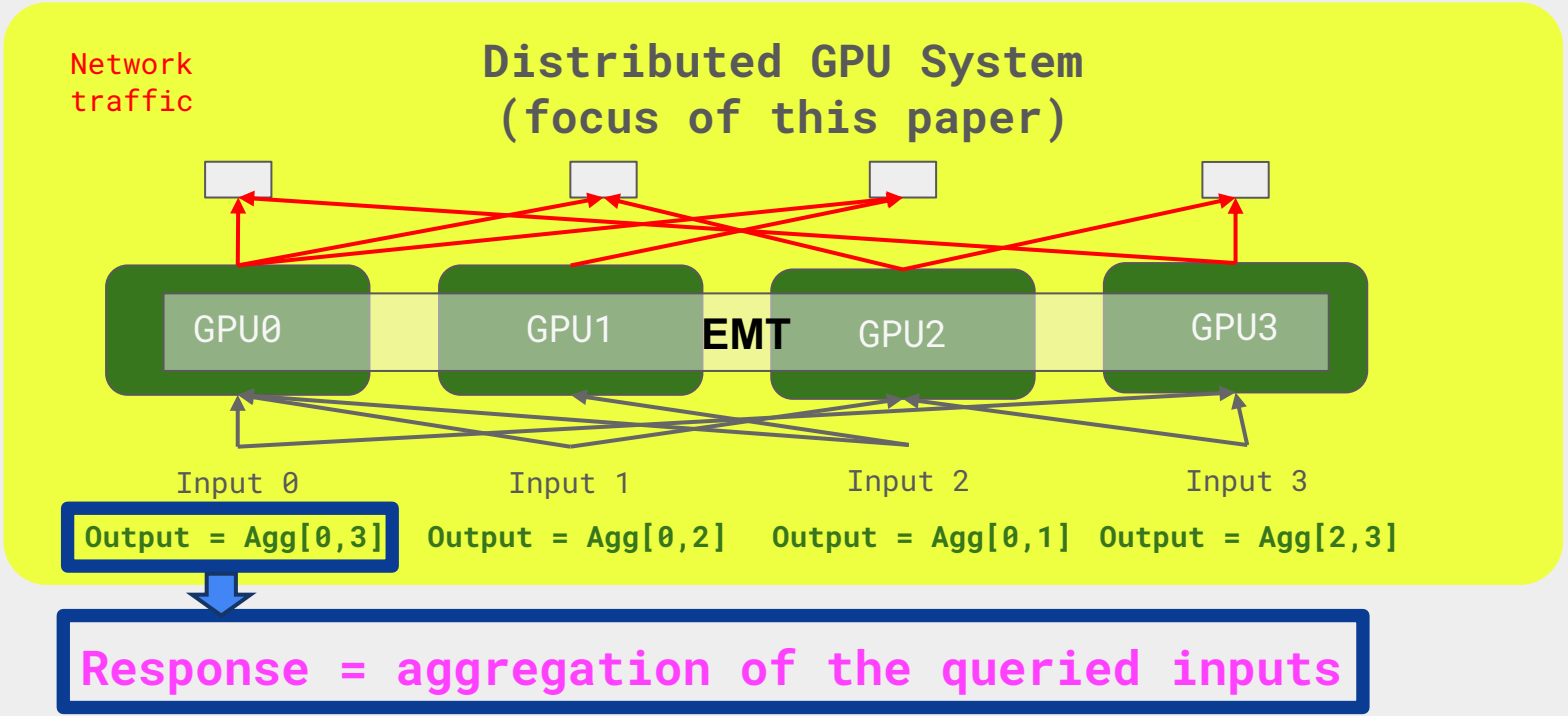


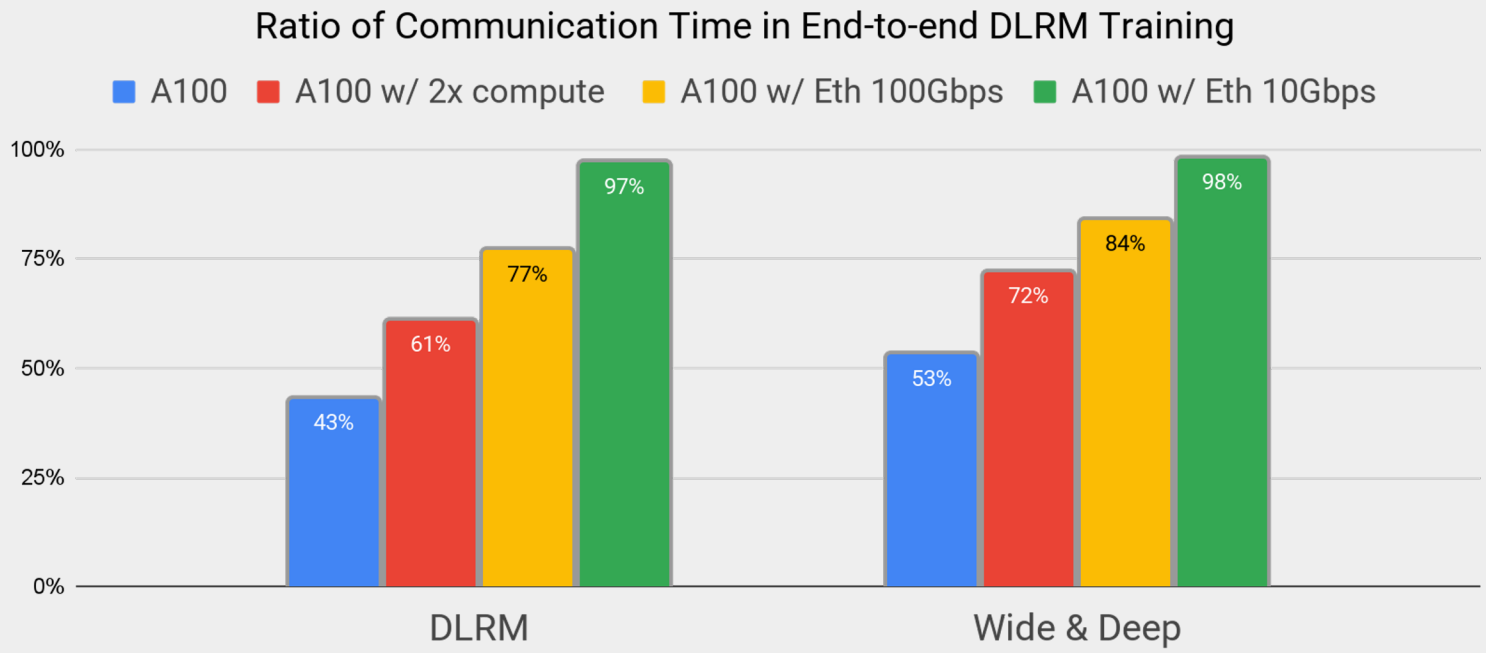
DLRM Algorithm

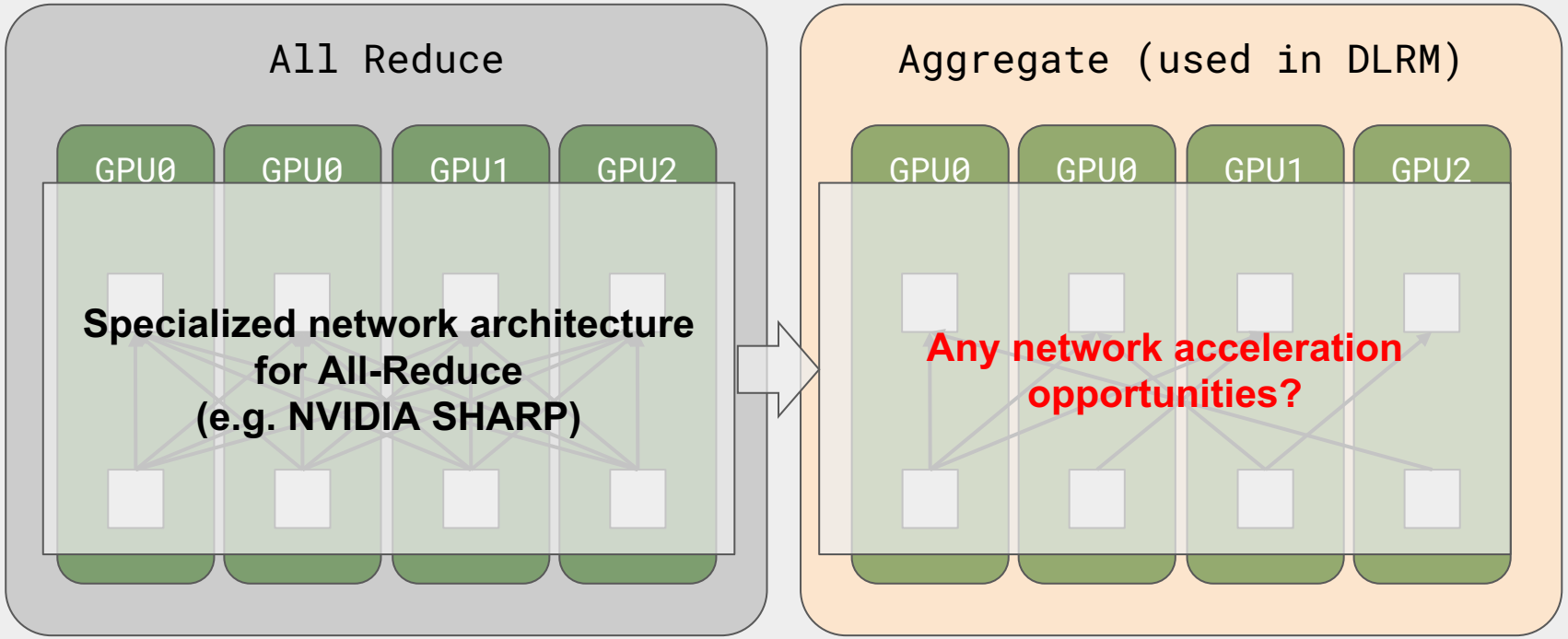
DLRM Systems | Performance Bottleneck

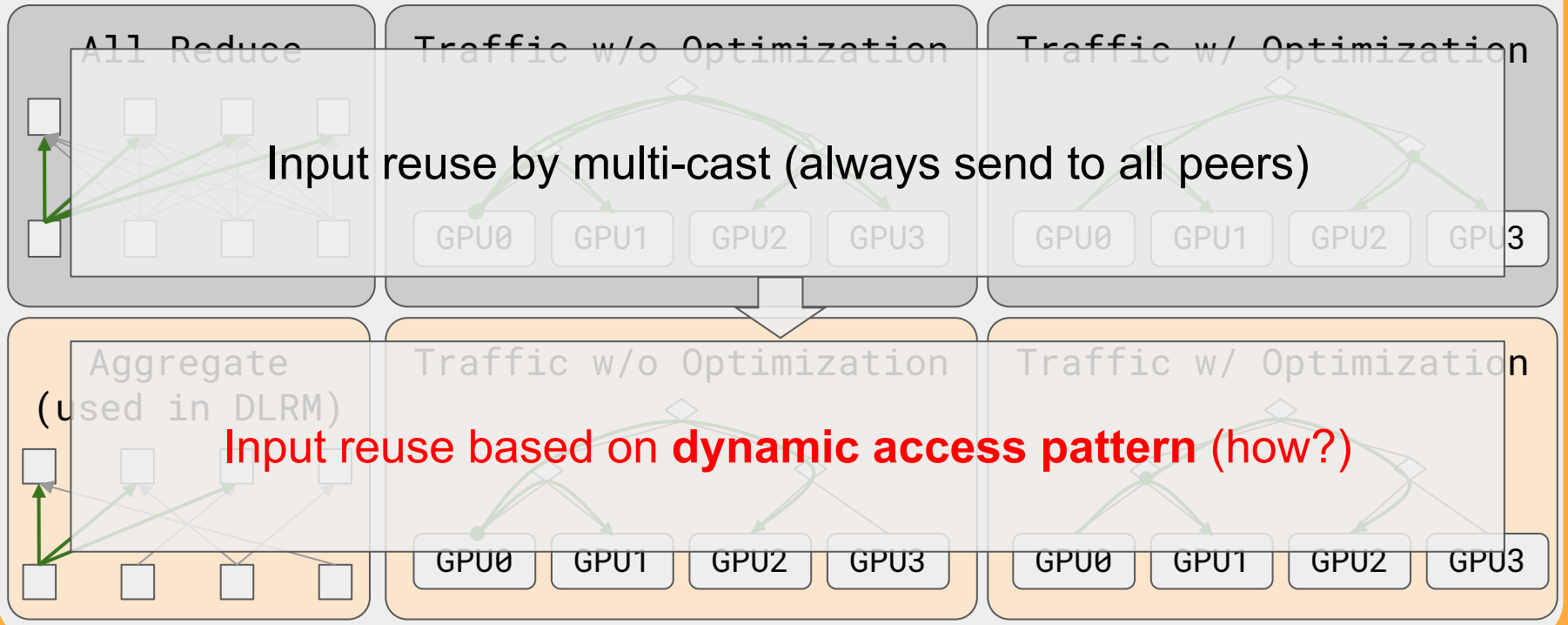




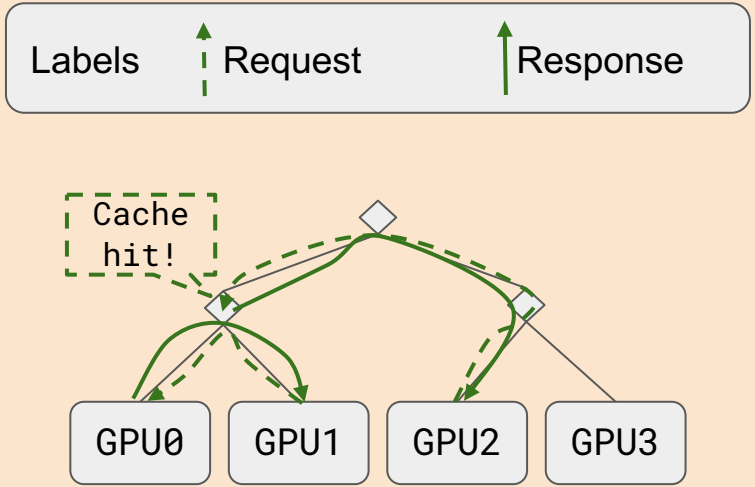

Network traffic
Distributed GPU System (focus of this paper)
GPU0
GPU1
EMT
GPU2
GPU3
Input 0
Input 1
Input 2
Input 3
Output = Agg[0,3]
Output = Agg[0,2]
Output = Agg[0,1]
Output = Agg[2,3]
Response = aggregation of the queried inputs

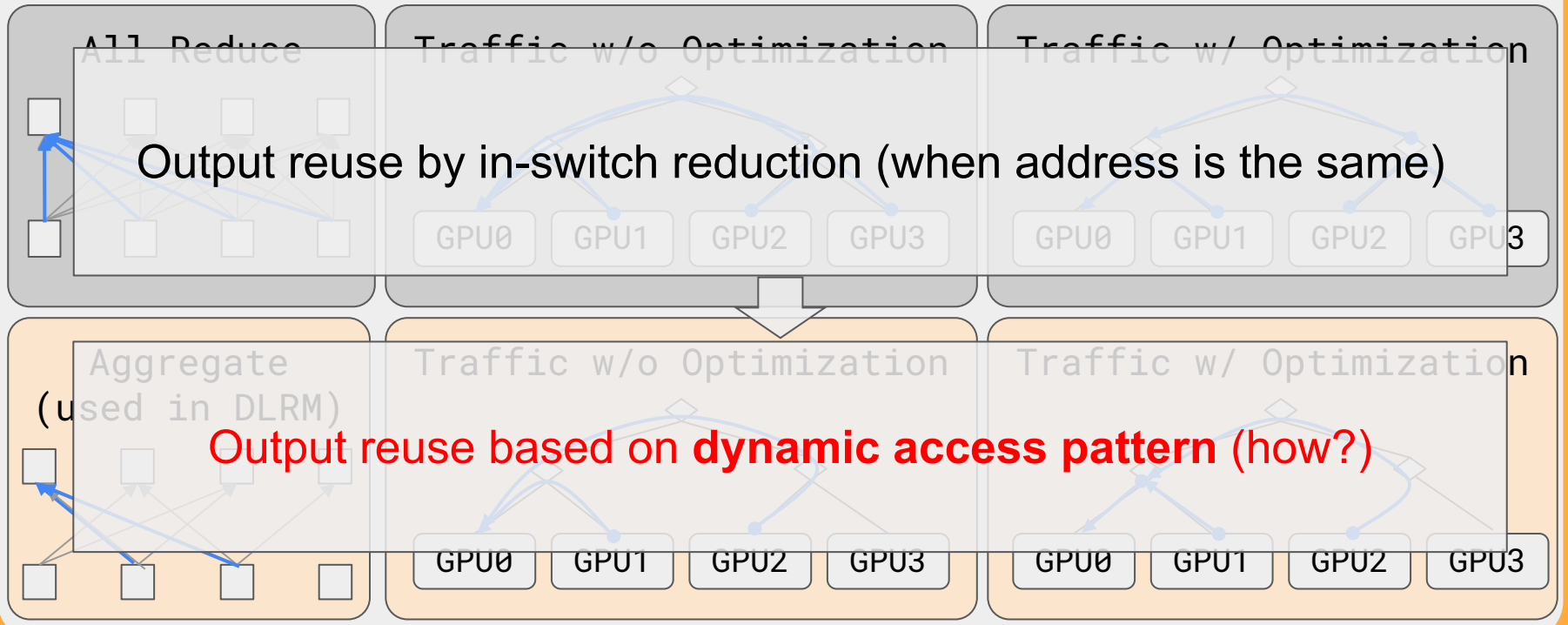

Ratio of Communication Time in End-to-end DLRM Training
A100
A100 w/ 2x compute
A100 w/ Eth 100Gbps
A100 w/ Eth 10Gbps
100%
75%
50%
25%
0%
43%
61%
77%
97%
53%
72%
84%
98%
DLRM
Wide & Deep


All Reduce
GPU0
GPU0
GPU1
GPU2
Specialized network architecture
for All-Reduce
(e.g. NVIDIA SHARP)
Aggregate (used in DLRM)
GPU0
GPU0
GPU1
GPU2
Any network acceleration
opportunities?

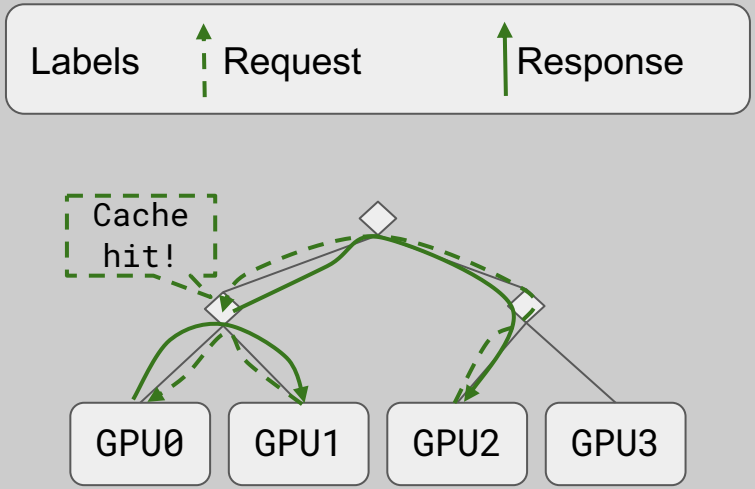

All Reduce
Traffic w/o Optimization
Traffic w/ Optimization
GPU0
GPU1
GPU2
GPU3
Input reuse by multi-cast (always send to all peers)
Aggregate
(used in DLRM)
Traffic w/o Optimization
Traffic w/ Optimization
Input reuse based on **dynamic access pattern** (how?)

Part 1 Input Reuse via Cache

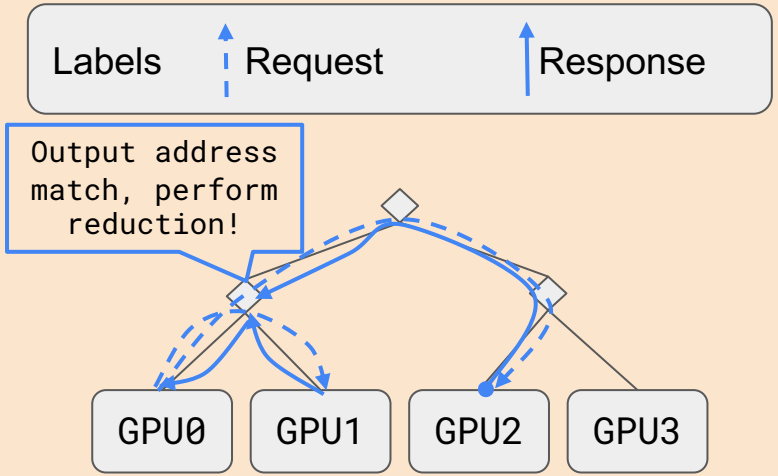


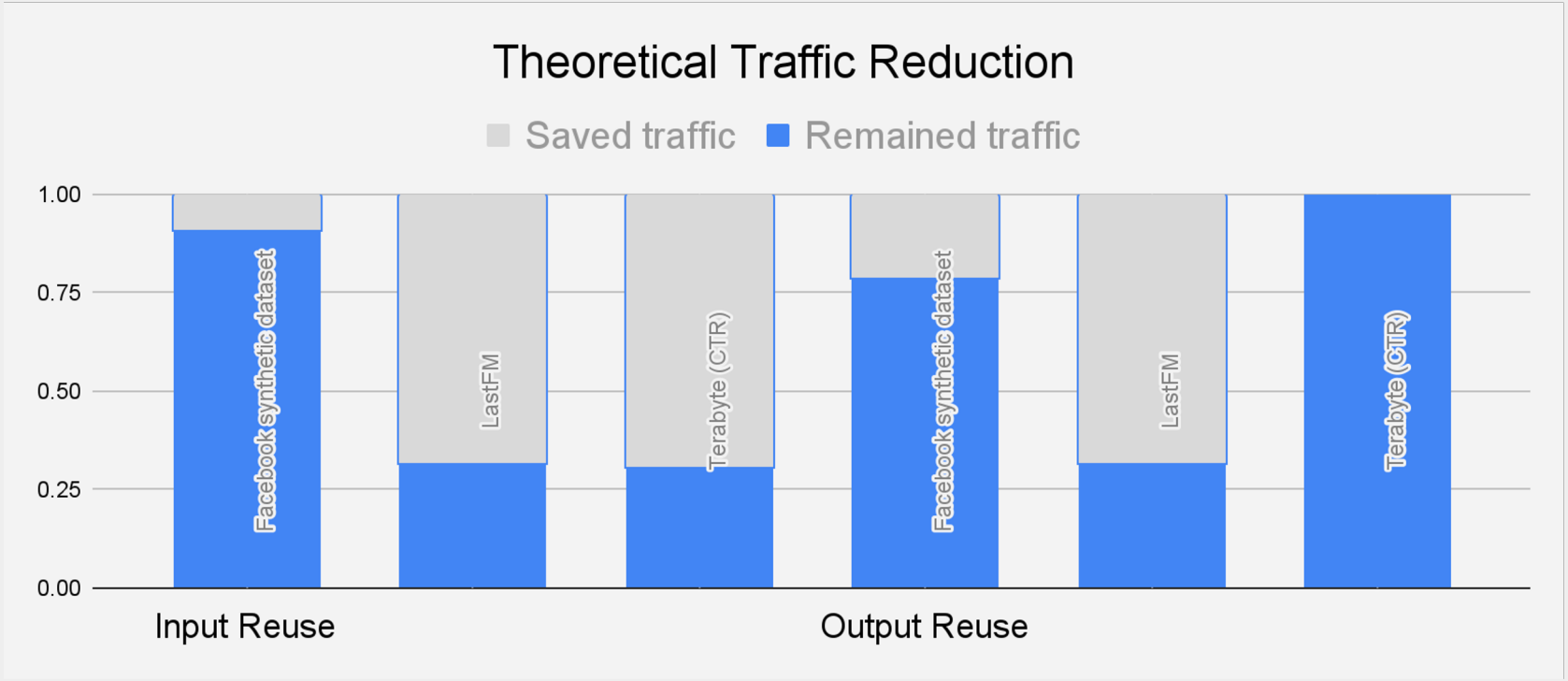

All Reduce
Traffic w/o Optimization
Traffic w/ Optimization
GPU0
GPU1
GPU2
GPU3
Output reuse by in-switch reduction (when address is the same)
Aggregate
(used in DLRM)
Traffic w/o Optimization
Traffic w/ Optimization
Output reuse based on **dynamic access pattern** (how?)

Part 1 Input Reuse via Cache



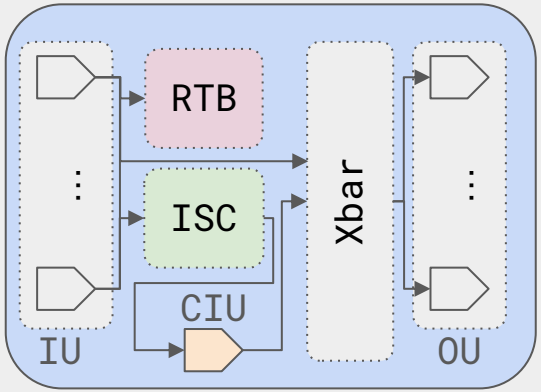
Part 2 Output Reuse via in network reduction




Theoretical Traffic Reduction
Saved traffic
Remained traffic
1.00
0.75
0.50
0.25
0.00
Facebook synthetic dataset
LastFM
Terabyte (CTR)
Facebook synthetic dataset
LastFM
Terabyte (CTR)
Input Reuse
Output Reuse

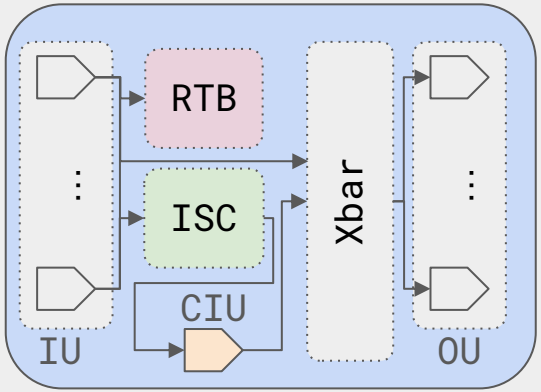
In-Switch Cache

Reduction Table



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

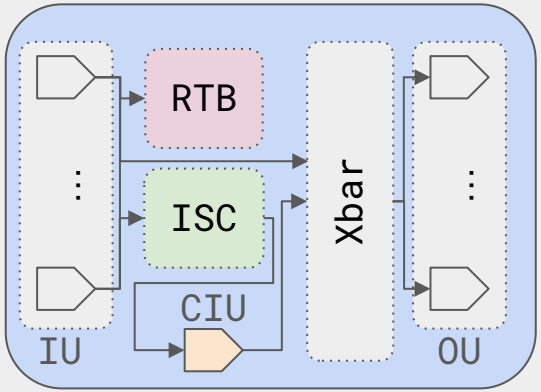
Reduction Table

Invalid state
(Addr=X)

S	Addr	Data
00		
00		

State (S) codebook:
00: invalid
01: valid address, incomplete data
10: valid address & data

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

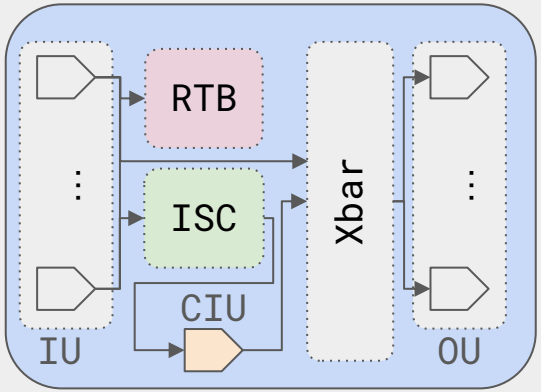
Reduction Table

Incoming **data** packet
(Addr=X)

S	Addr	Data
01	X	(incomplete)
00		

State (S) codebook:
00: invalid
01: valid address, incomplete data
10: valid address & data

In-Switch Cache



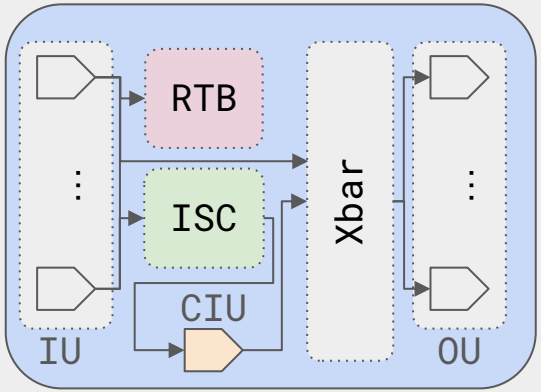
IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

Reduction Table

S	Addr	Data
10	X	(complete)
00		

State (S) codebook:
00: invalid
01: valid address, incomplete data
10: valid address & data

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

Reduction Table

Incoming **request** packet
(Addr=X) hit!

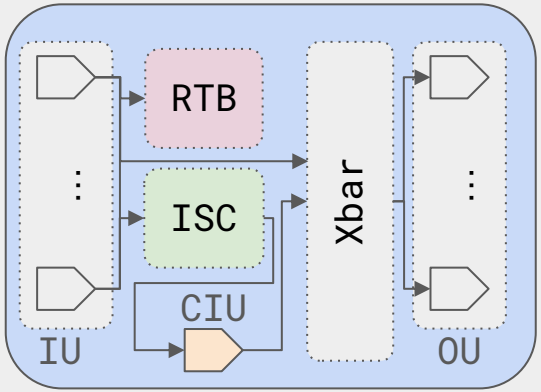
S	Addr	Data
10	X	(complete)
00		

Create data packet from
cache entry 0

To CIU

State (S) codebook:
00: invalid
01: valid address,
incomplete data
10: valid address & data

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

Reduction Table

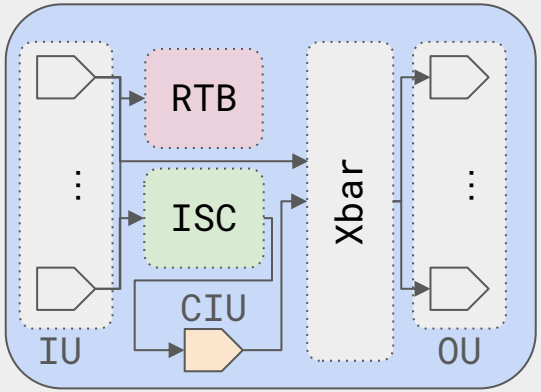
Stall request in IU and retry next cycle

Incoming **request** packet (Addr=Y) hit on incomplete cacheline

S	Addr	Data
10	X	(complete)
01	Y	(incomplete)

State (S) codebook:
00: invalid
01: valid address, incomplete data
10: valid address & data

In-Switch Cache



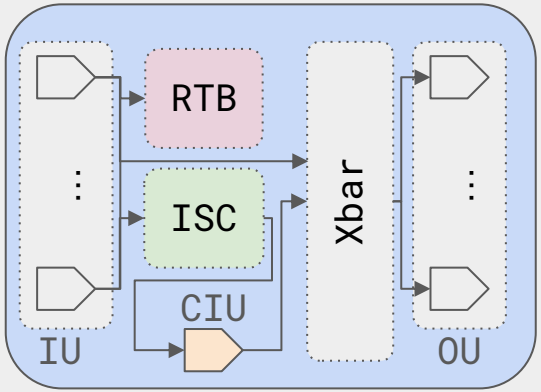
IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

Reduction Table

Addr	Data	W	A
		0	0

W: waiting number of responses
A: arrived number of responses

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

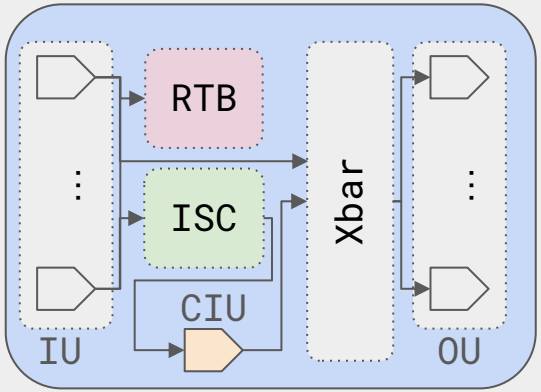
Reduction Table

Incoming **request** packet
(output address=X)

Addr	Data	W	A
X	0	1	0

W: waiting number of responses
A: arrived number of responses

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

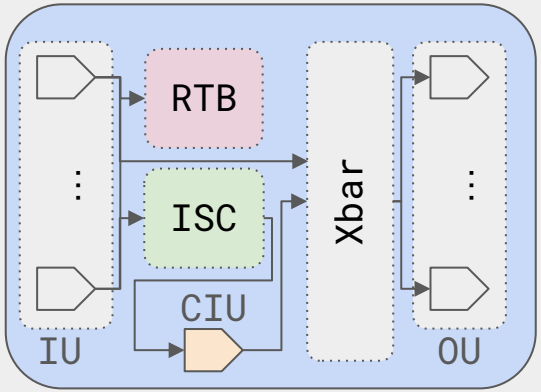
Reduction Table

Another **request** packet
(output address=X)

Addr	Data	W	A
X	0	2	0

W: waiting number of responses
A: arrived number of responses

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

Reduction Table

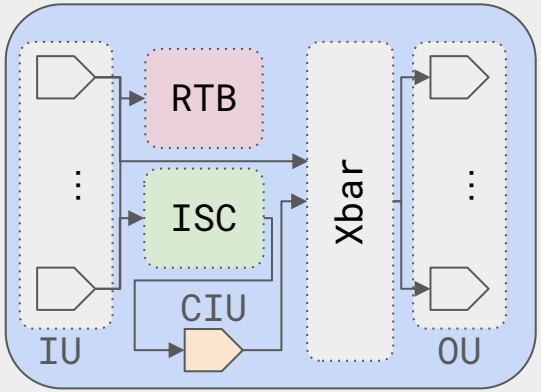
Incoming **data** packet
(output address=X, data=5)

Addr	Data	W	A
X	5	1	1

This packet is dropped and not forwarded to next hop

W: waiting number of responses
A: arrived number of responses

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

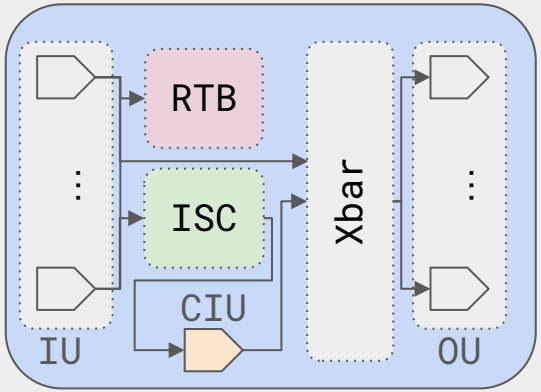
Reduction Table

Another **data** packet
(output address=X, data=2)

Addr	Data	W	A
X	7 (5+2)	0	2

W: waiting number of responses
A: arrived number of responses

In-Switch Cache



IU: input units
OU: output units
CIU: cache input unit
RTB: reduction table
ISC: in-switch cache

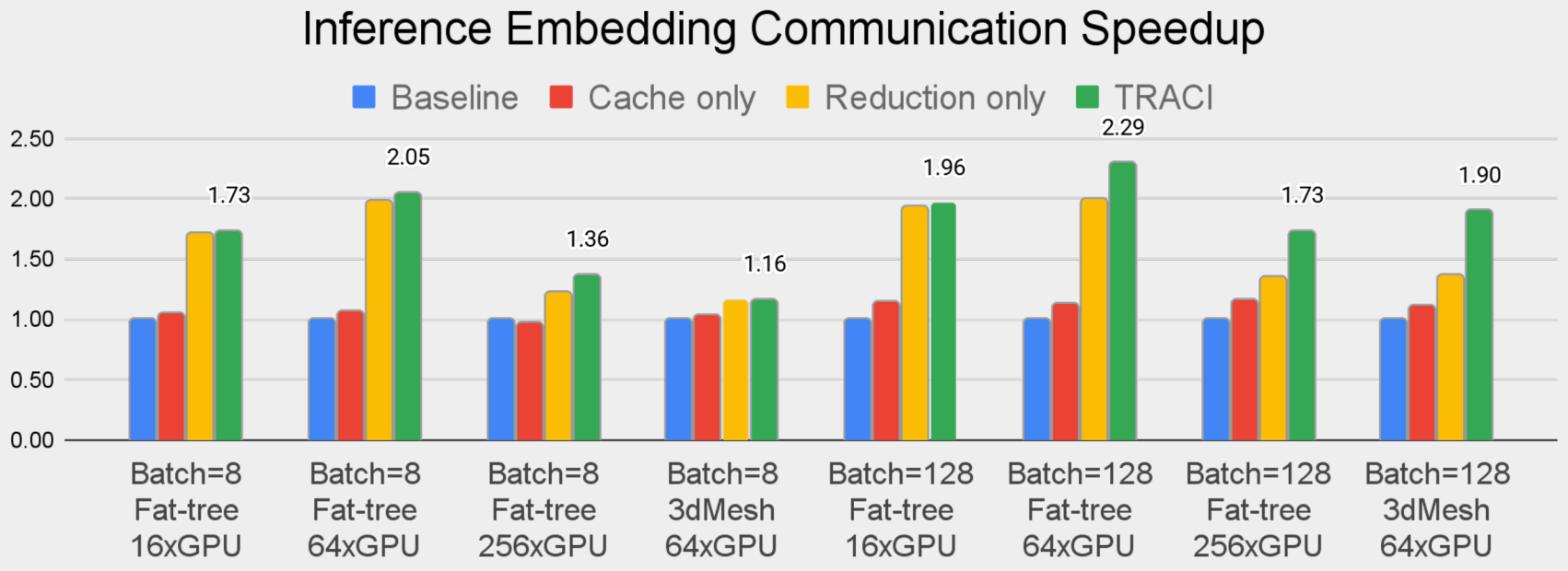
Reduction Table

Addr	Data	W	A
X	7 (5+2)	0	2

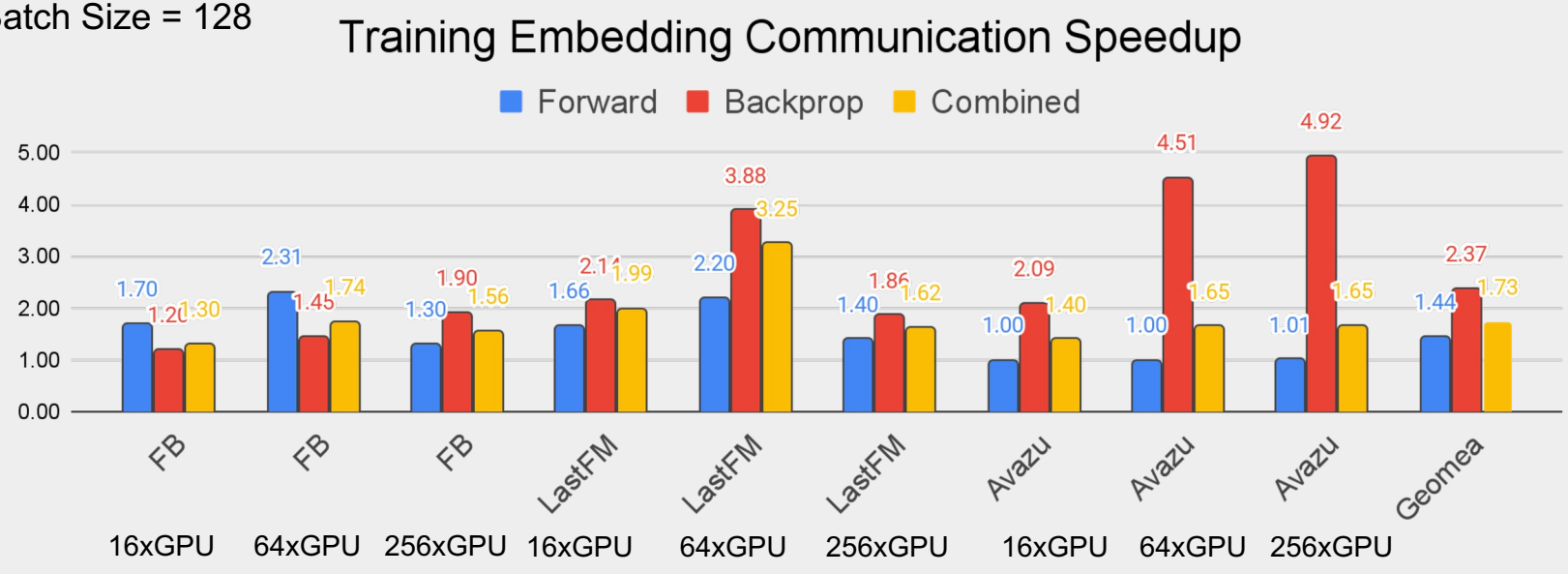
W=0, create packet and forward to next hop (output address=X, data=7, A)

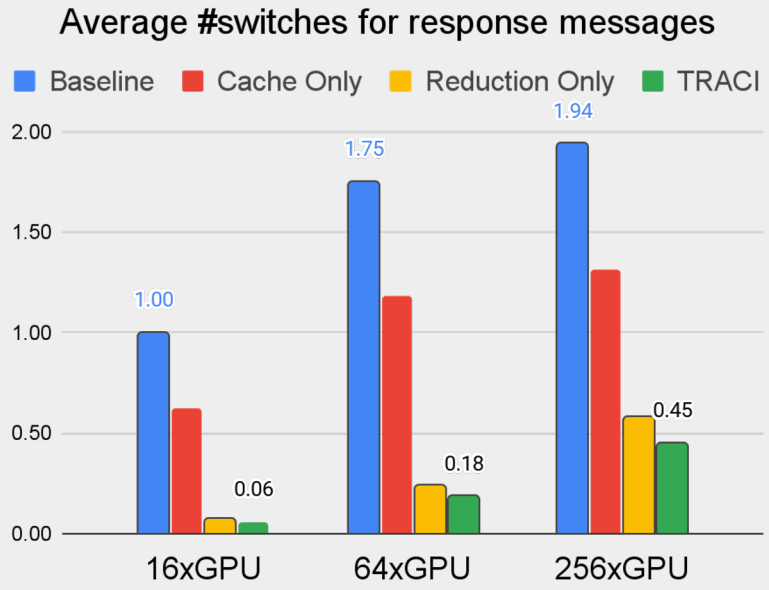
W: waiting number of responses
A: arrived number of responses

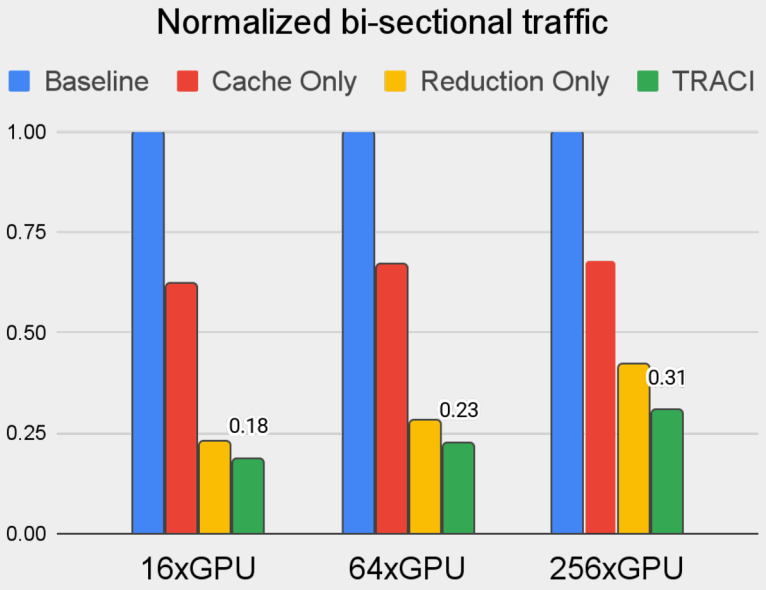
- Candidate network: high-speed accelerator interconnect (e.g. NVLink)
- Candidate topology: two-layer fat-tree; 3D mesh
- Communication simulation: gem5 Garnet network simulator
- End-to-end application simulation: Astra-sim
- Benchmark: 23 DLRM application traces
  - 17 facebook synthetic datasets
  - 3 Click-Through Rate (CTR) datasets
  - 3 web-review datasets


Inference Embedding Communication Speedup
Baseline
Cache only
Reduction only
TRACI
2.50
2.00
1.50
1.00
0.50
0.00
1.73
2.05
1.36
1.16
1.96
2.29
1.73
1.90
Batch=8 Fat-tree 16xGPU
Batch=8 Fat-tree 64xGPU
Batch=8 Fat-tree 256xGPU
Batch=8 3dMesh 64xGPU
Batch=128 Fat-tree 16xGPU
Batch=128 Fat-tree 64xGPU
Batch=128 Fat-tree 256xGPU
Batch=128 3dMesh 64xGPU

Batch Size = 128




Average #switches for response messages
Baseline
Cache Only
Reduction Only
TRACI
2.00
1.50
1.00
0.50
0.00
1.00
0.06
1.75
0.18
1.94
0.45
16xGPU
64xGPU
256xGPU


Normalized bi-sectional traffic
Baseline
Cache Only
Reduction Only
TRACI
1.00
0.75
0.50
0.25
0.00
0.18
0.23
0.31
16xGPU
64xGPU
256xGPU

	Size	Area (mm^2)	Percentage
In-Switch Cache	2 MB	3.37	1.15%
Reduction Table	2 MB	4.92	1.67%
Total		8.29	2.82%

ISC and RTB both contains 8192 entries of 64 FP32 numbers, so each table is 2MB size. Area is evaluated with CACTI in 22nm and scaled to the same process node as NVLink4 NVSwitch. For total switch area, we use NVLink4 NVSwitch as a reference (294mm^2).

Summary

- TRACI is a network architecture (protocol and switch design) to accelerate the input-dynamic communication in distributed DLRM. It is built on top of DL accelerator network like NVLink.
- Main idea
  - The same input is accessed many times, in-switch cache can avoid repeatedly injecting the same data into the network and improve response latency.
  - Many messages will be reduced to the same output, reduction table can perform this reduction in switches to reduce network traffic.
- Results
  - Up to 2.29x speedup for inference communication and 3.25x for training communication
  - Benefit is verified by significant reduction of bi-sectional traffic and message traversal hops.