Attention: Tutorial is being recorded

AI System Tradeoff & Resource Analysis simulator

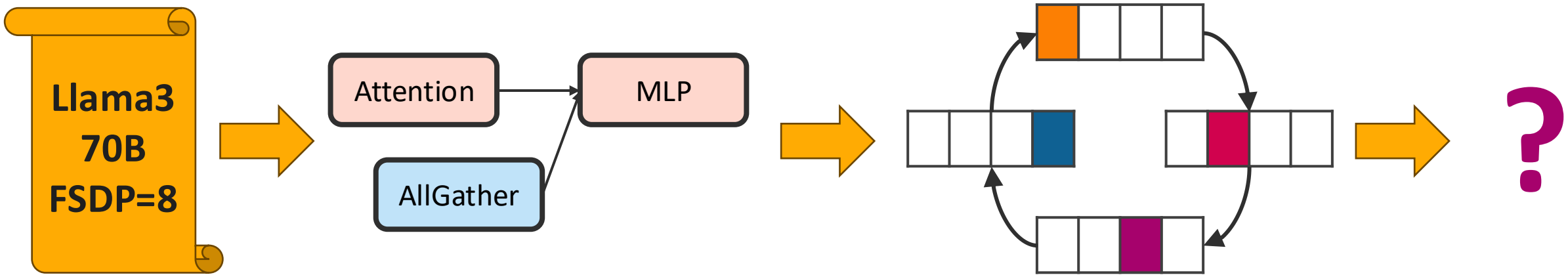
Enabling Software-Hardware Co-Design Exploration for Distributed AI Platforms

Jinsun Yoo
Georgia Institute of Technology

ASTRA-sim Tutorial
@ISCA 2026
June 26, 2026

https://astra-sim.github.io/tutorials/isca-2026
https://astra-sim.github.io

10,000ft view



Model Design

Workload Layer (Chakra):
Graph Rep. of Workload

System Layer:
Breaking down collectives

NetworkAPI

- `sim_send(msg_size, dest, callback)`
  - **Simulate asynchronous message send** of size msg_size to dest and **invoke callback function** once transmission has finished

- `sim_recv(msg_size, src, callback)`
  - **Simulate asynchronous message receive** of size msg_size from src and **invoke callback function** once transmission has finished

Declared at: astra-sim/common/AstraNetworkAPI.hh

NetworkAPI at System Layer

- Ring All-Reduce algorithm implementation

```
bool Ring::ready() {
        (...)
        stream->owner->sim_send(0, Sys::dummy_data, msg_size, UINT8, packet.preferred_dest, stream-
        >stream_id,
        &snd_req, &Sys::handleEvent, nullptr);

        (...)
        stream->owner->sim_recv(0, Sys::dummy_data, msg_size, UINT8, packet.preferred_src, stream-
        >stream_id,
        &rcv_req, &Sys::handleEvent, ehd);

        reduce();
        return true;
}
```

Send a chunk

Receive a chunk

NetworkAPI

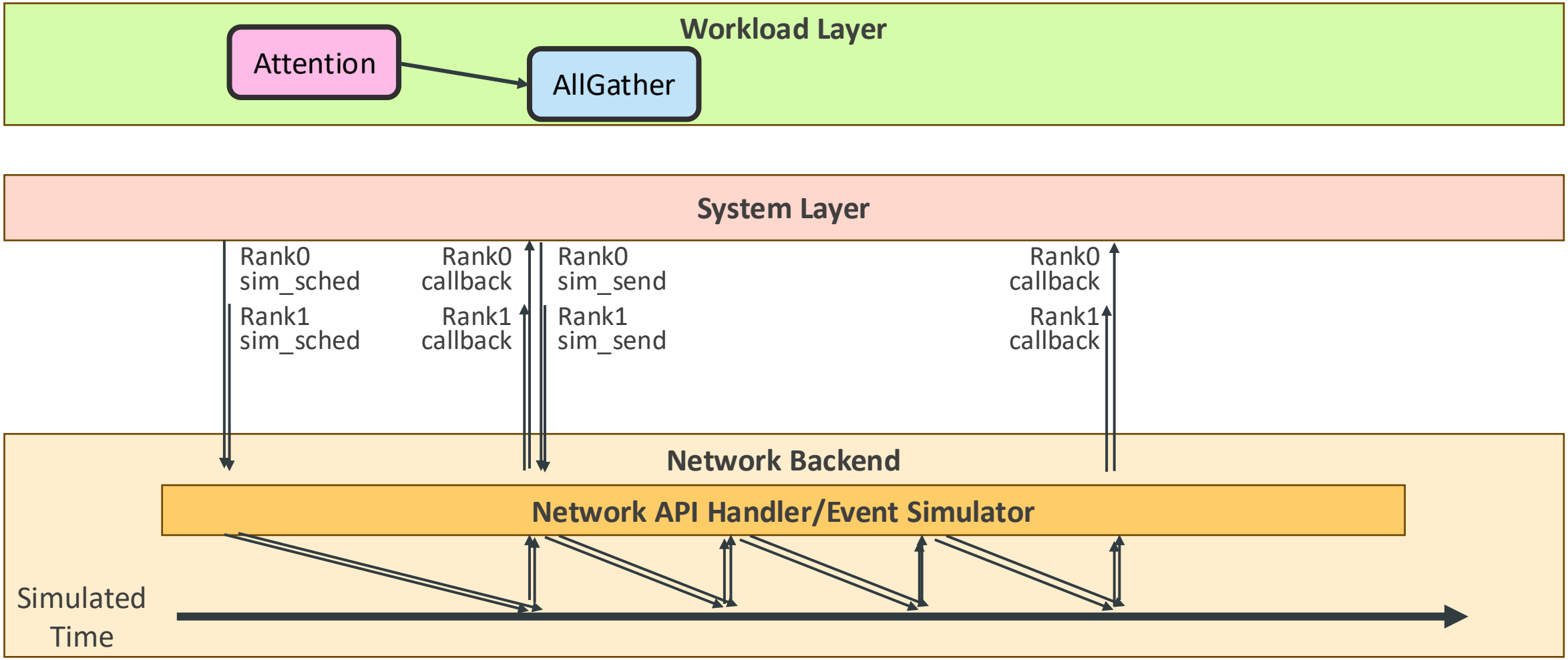
- **`sim_send(msg_size, dest, callback)`**
  - **Simulate asynchronous message send** of size msg_size to dest and **invoke callback function** once transmission has finished

- **`sim_recv(msg_size, src, callback)`**
  - **Simulate asynchronous message receive** of size msg_size from src and **invoke callback function** once transmission has finished

- **`sim_sched(time_delta, callback)`**
  - **Simulate passing of time** and **invoke callback function** once time passed

Declared at: astra-sim/common/AstraNetworkAPI.hh

Network Backend also Manages Simulated Time



Available Network Backends

- Network backends are maintained separately and are imported as **submodule**.

- ASTRA-sim OSS currently has **4 network backends**

Backend	Purpose	Notable Feature
analytical/analytical	analytical equation-based simulation	fast simulation, hierarchical topologies
analytical/congestion	congestion-aware analytical simulation	first-order congestion (queueing) modeling
ns-3	Packet level simulation based on ns-3	RoCE modelling
htsim	Packet level simulation based on htsim	UET modelling

Analytical Backend

- Leverages **analytical equation** to estimate communication delay

- `delay(msg_size, src, dest) =`
  `(#hops)×(link latency) + (msg_size)/(link BW)`

- No congestion modeling
  - Appropriate for topology-aware collectives without network congestion
- **Fast simulation** for large-scale systems

(ISPASS '23) ASTRA-sim2.0: Modeling Hierarchical Networks and Disaggregated Systems for Large-model Training at Scale

For more details, refer to the MICRO 2024 tutorial: https://astra-sim.github.io/tutorials/micro-2024

Analytical Network Backend

- Network backends are maintained separately and are imported as **submodule**.

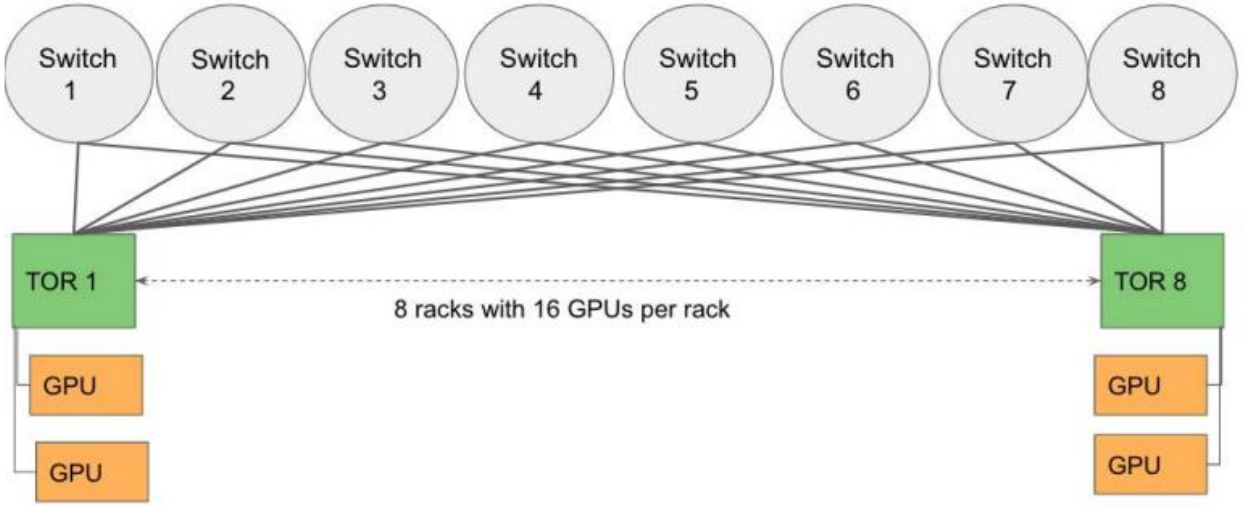
- ASTRA-sim OSS currently has **4 network backends**

Backend	Purpose	Notable Feature
analytical/analytical	analytical equation-based simulation	fast simulation, hierarchical topologies
analytical/congestion	congestion-aware analytical simulation	first-order congestion (queueing) modeling
ns-3	Packet level simulation based on ns-3	RoCE modelling
htsim	Packet level simulation based on htsim	UET modelling

For more details, refer to the MICRO 2024 tutorial: https://astra-sim.github.io/tutorials/micro-2024

ns-3 Backend

- Network simulator for inter-node **RDMA communication, specifically RoCE**
- Used to model ML training in **multi-tiered GPU clusters**
- Packet level RoCE simulation.



(HOTI '22) Current RoCE congestion control methods have little impact on ML training workloads

What do we mean by “Packet Level”?

```
int SwitchNode::GetOutDev(Ptr<const Packet> p, CustomHeader &ch){
  // look up entries
  auto routes = m_rtTable.find(ch.dip);

  buf.u32[0] = ch.sip;
  buf.u32[1] = ch.dip;
  buf.u32[2] = ch.udp.sport | ((uint32_t)ch.udp.dport << 16);

  uint32_t idx = EcmpHash(buf.u8, 12, m_ecmpSeed) % nexthops.size();
  return routes[idx];
}
```

Found at: ns-3/src/point-to-point/model/switch-node.cc

ns-3 Network Configurations

- Detailed internetwork behavior modeling/simulation

PACKET_PAYLOAD_SIZE	packet size
CC_MODE	Congestion control algorithm
BUFFER_SIZE	switch buffer size
	0: ACK has same priority with data packet
ACK_HIGH_PRIO	1: prioritize ACK
RATE_BOUND	Bound rate to a limited rate
ENABLE_QCN	Whether QCN (Quantized Congestion Notification) is enabled
L2_BACK_TO_ZERO	(Go-Back-N protocol) Layer 2 go back to zero transmission
L2_CHUNK_SIZE	(Go-Back-N protocol) Layer 2 chunk size
L2_ACK_INTERVAL	(Go-Back-N protocol) Layer 2 Ack intervals
HAS_WIN	Whether to use a window
	0: different server pairs use their own RTT as T
GLOBAL_T	1: use the max base RTT as the global T
VAR_WIN	Whether the window size is variable
RATE_BOUND	Use rate limitor
ACK_HIGH_PRIO	Prioritize acknowledgement packets
KMAX_MAP	a map from link bandwidth to ECN threshold kmax
KMIN_MAP	a map from link bandwidth to ECN threshold kmin
PMAX_MAP	a map from link bandwidth to ECN threshold pmax
RATE_AI	Rate increment unit in AI period
RATE_HAI	Rate increment unit in hyperactive AI period
MIN_RATE	Minimum rate of a throttled flow

ns-3 Network Topology

```
9 1 8
8
8 0 400Gbps 0.0005ms 0
8 1 400Gbps 0.0005ms 0
8 2 400Gbps 0.0005ms 0
8 3 400Gbps 0.0005ms 0
8 4 400Gbps 0.0005ms 0
8 5 400Gbps 0.0005ms 0
8 6 400Gbps 0.0005ms 0
8 7 400Gbps 0.0005ms 0
```

#total deices, #switches, #links

switch device IDs

link information

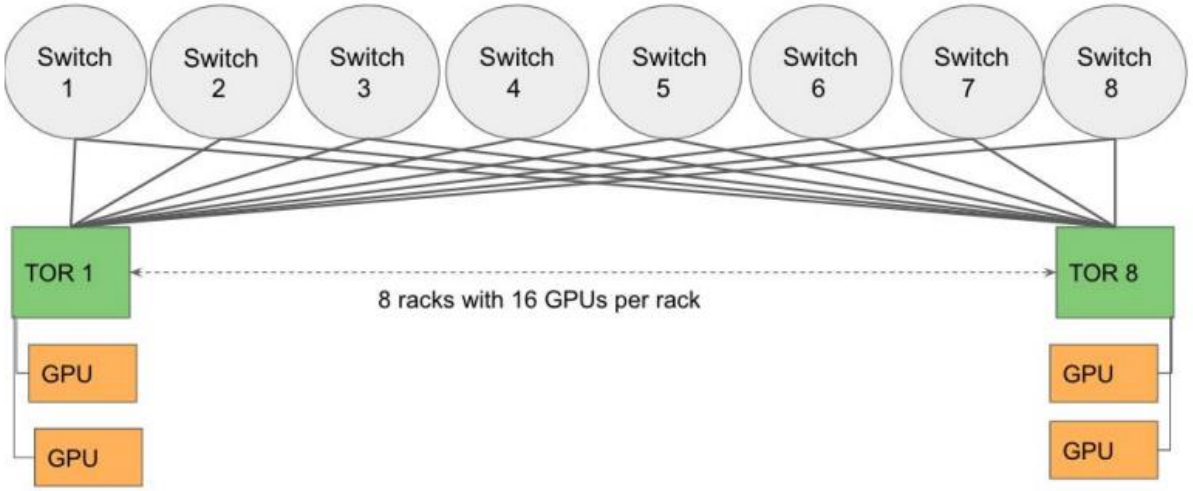
ns-3 Network Topology

```
144 16 192
128 129 130 … 142 143
0 128 200Gbps 0.005ms 0
1 128 200Gbps 0.005ms 0
2 128 200Gbps 0.005ms 0
3 128 200Gbps 0.005ms 0
128 136 200Gbps 0.0125ms 0
128 137 200Gbps 0.0125ms 0
128 138 200Gbps 0.0125ms 0
```

#total deices, #switches, #links

switch device IDs

link information



Physical Topology Setup

8_nodes_1_switch.txt:

Total # node(NPU) + Switch **#Switches** **#Links**

List of Switch ID

List of
- **Endpoint ID 1**
- **Endpoint ID 2**
- **Bandwidth**
- **Latency**
- **Error Rate**

```
9 1 8
8
8 0 400Gbps 0.0005ms 0
8 1 400Gbps 0.0005ms 0
8 2 400Gbps 0.0005ms 0
8 3 400Gbps 0.0005ms 0
8 4 400Gbps 0.0005ms 0
8 5 400Gbps 0.0005ms 0
8 6 400Gbps 0.0005ms 0
8 7 400Gbps 0.0005ms 0
```

Physical Topology Setup

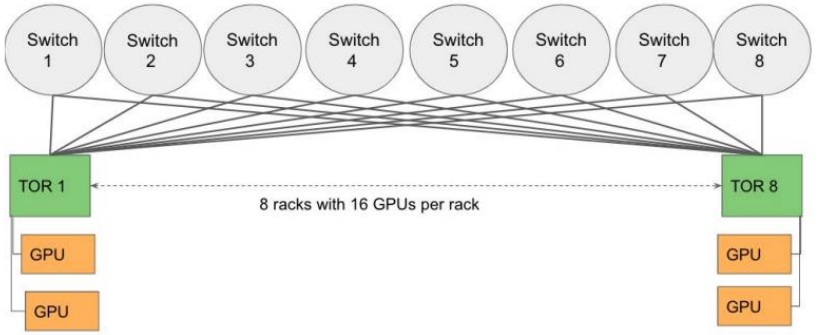
128_nodes_16_switch.txt:

Total # node(NPU) + Switch **#Switches** **#Links**

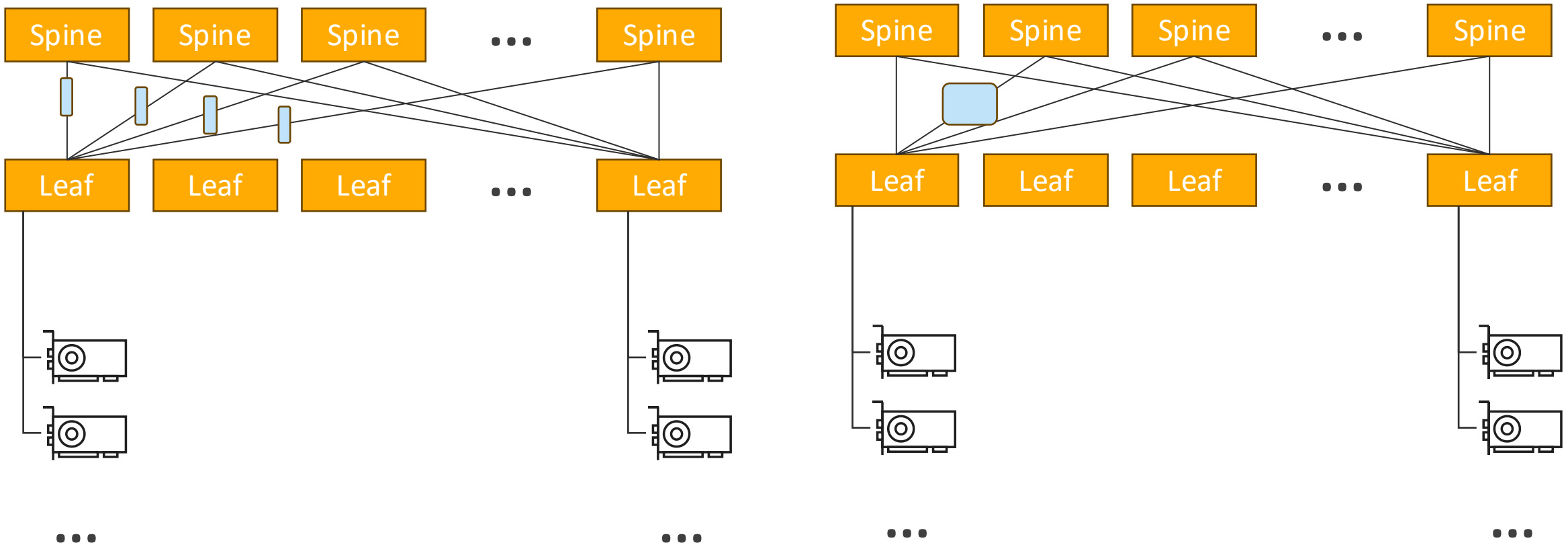
List of Switch ID

List of
- **Endpoint ID 1**
- **Endpoint ID 2**
- **Bandwidth**
- **Latency**
- **Error Rate**

```
144 16 192
128 129 130 …  142 143
0 128 200Gbps 0.005ms 0
1 128 200Gbps 0.005ms 0
2 128 200Gbps 0.005ms 0
3 128 200Gbps 0.005ms 0
128 136 200Gbps 0.0125ms 0
128 137 200Gbps 0.0125ms 0
128 138 200Gbps 0.0125ms 0
```



Case Study: Collective Algorithm and Load Balancing



Case Study: Collective Algorithm and Load Balancing

System Input

system_bigchunks.json

```
{
  "preferred-dataset-splits": 1,
  "all-to-all-implementation”:
    ["ring”]
}
```

system_manychunks.json

```
{
  "preferred-dataset-splits": 1024,
  "all-to-all-implementation”:
    ["ring”]
}
```

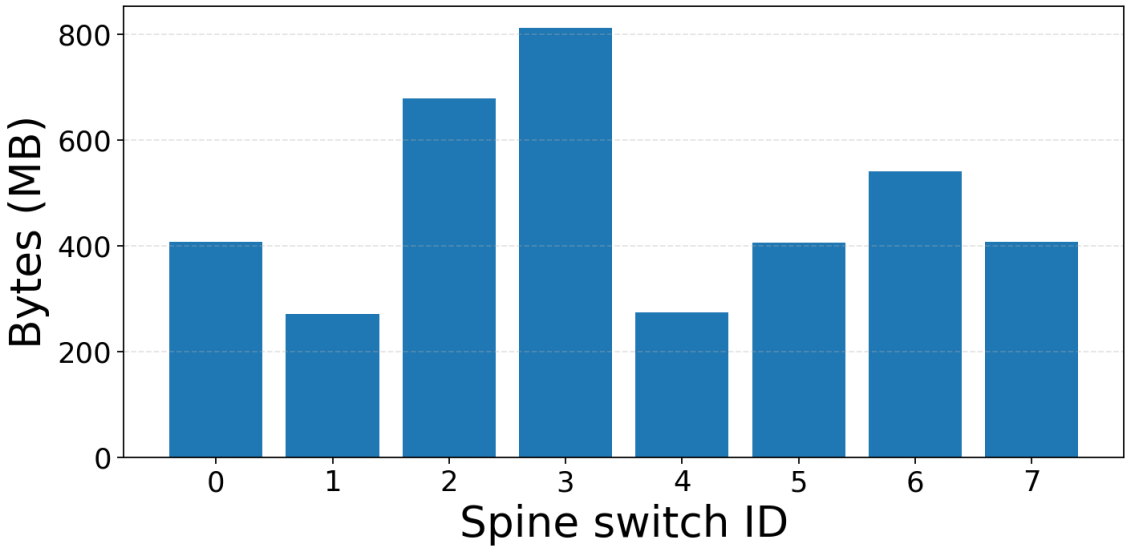
NS-3 Input

topology.txt

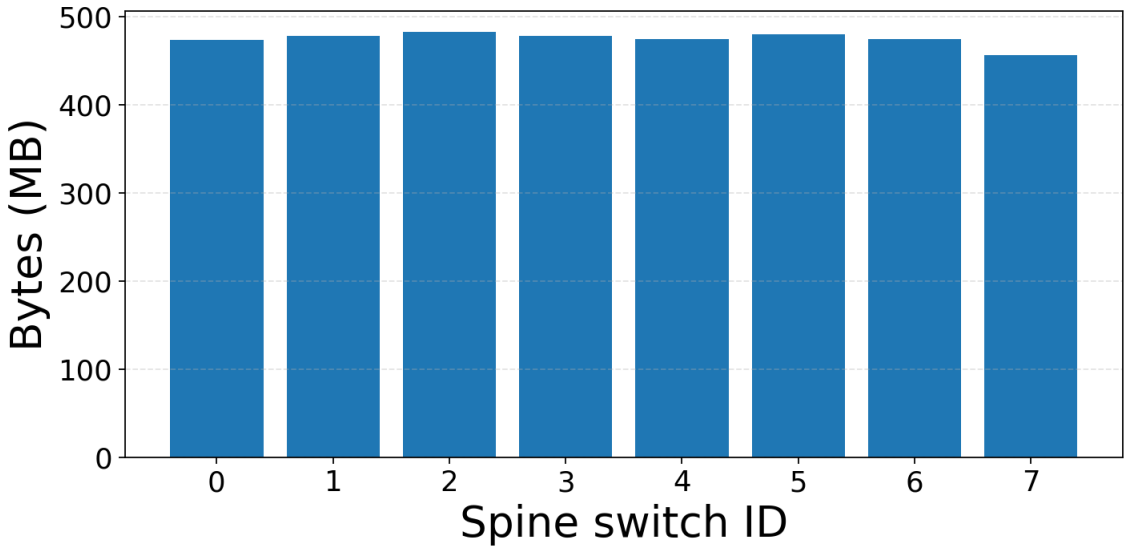
```
24 16 72
8 9 10 11 12 13 14 15 16 17 18
19 20 21 22 23
0 8 100Gbps 0.005ms 0
1 9 100Gbps 0.005ms 0
2 10 100Gbps 0.005ms 0
```

Case Study: Collective Algorithm and Load Balancing

Small number of big messages



Large number of small messages



Creating Your Own Network Backend

- Network API handler: frontend to implement network API and calls internal network simulation

  class AstraSimnetwork in astra-sim/astra-sim/network_frontend/ns3/AstraSimNetwork.cc

- main.cc: Parses arguments. Instantiates Network API Handler, Workload, Sys classes

  astra-sim/astra-sim/network_frontend/ns3/AstraSimNetwork.cc

- CMakeLists.txt:
  - Builds the network simulation as a shared library.
  - Builds the network frontend/API handler. Imports network simulation & AstraSim shared library

  astra-sim/extern/network_backend/ns-3/CMakeLists.txt

- build.sh: Toggles the build process

  astra-sim/build/astra_ns3/build.sh

Example PR: https://github.com/astra-sim/astra-sim/pull/119