https://astra-sim.github.io

https://github.com/mlcommons/chakra

ASTRA-sim Tutorial
@MICRO 2024
Nov 3rd, 2024

ASTRA-Sim and Chakra Tutorial: *Demo*

Joongun Park
Post Doctoral Researcher
School CS, Georgia Institute of Technology
jpark3234@gatech.edu

Outline

- **Prerequisites**
  - **Installing Chakra and ASTRA-sim2.0**

- **Part 1: Generating traces with Chakra**

- **Part 2: Running Simulation with ASTRA-Sim**

Installing Chakra and ASTRA-sim2.0

- We strongly **recommend** using the **provided Docker environment** for execution.
  - *https://astra-sim.github.io/tutorials/MICRO-2024/installation*

- Or you can build by your own with official repositories.
  - Chakra : https://github.com/astra-sim/chakra.git
  - ASTRA-Sim: https://github.com/astra-sim/astra-sim.git

Cloning Tutorial Repository

- We provide sandboxed Chakra/ASTRA-sim for tutorial purposes

```
$ git clone git@github.com:astra-sim/tutorials.git
$ cd tutorials/micro2024
$ ./clone_repos.sh
```

Launching Execution Environment (Docker)

- Download Docker Image

```
$ docker pull astrasim/tutorial-micro2024
```

- Start a Docker Container and **link current directory into**

```
$ docker run -i -t \
     -v $(pwd):/app/micro2024 \
     astrasim/tutorial-micro2024
[docker]$ cd ../micro2024/
```

Mount

Mounted directory

Build and Install

- Install Chakra

```
[docker]$ ./install_chakra.sh
```

- Compile ASTRA-sim with the analytical network backend

```
[docker]$ ./compile_astra_sim.sh
```

Outline

- Prerequisites

- Part 1: Generating traces with Chakra
    - (Chakra Demo 1) Simple Chakra ET – Chakra API
    - (Chakra Demo 2) Text based approach - Chakra converter
    - (Chakra Demo 3) Synthesizing Chakra ET - Synthetic Trace Generator
    - (Chakra Demo 4) Collecting Traces from Actual Runs – PyTorch / Chakra

- Part 2: Running Simulation with ASTRA-Sim

Simple Chakra ET - Manually

- Chakra offers ET Generation API
  - For manual design and implementation of arbitrary chakra ETs

generate_all_reduce.py:

```
(...)
node = ChakraNode()
node.id = 1
node.name = "All-Reduce"
node.type = COMM_COLL_NODE
node.attr.append(ChakraAttr(name="comm_type", int64_val=ALL_REDUCE))
node.attr.append(ChakraAttr(name="comm_size", uint64_val=1_048_576))
encode_message(et, node)
(...)
```

Create Chakra node

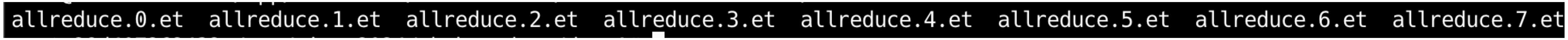
Store Chakra ET file

Simple Chakra ET – Microbenchmark

- 1 MB All-Reduce among 8 NPUs

```
[docker]$ cd chakra-demo
[docker]$ cd demo1
[docker]$ python3 generate_all_reduce.py
```

./demo1/allreduce



```
allreduce.0.et  allreduce.1.et  allreduce.2.et  allreduce.3.et  allreduce.4.et  allreduce.5.et  allreduce.6.et  allreduce.7.et
```

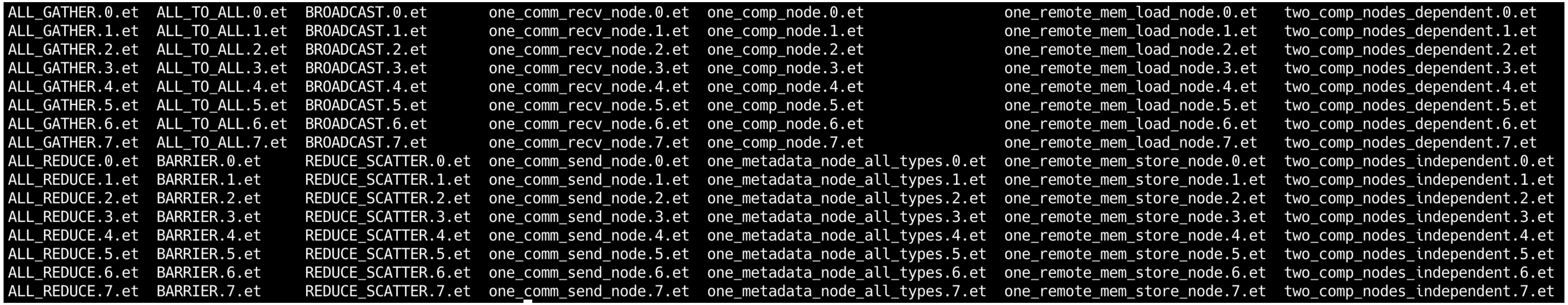
Generated Chakra ET Files

Simple Chakra ET – Microbenchmark

- 1 MB All-Reduce among 8 NPUs

```
[docker]$ cd demo1
[docker]$ python3 generate_all_reduce.py
```

./demo1/traces



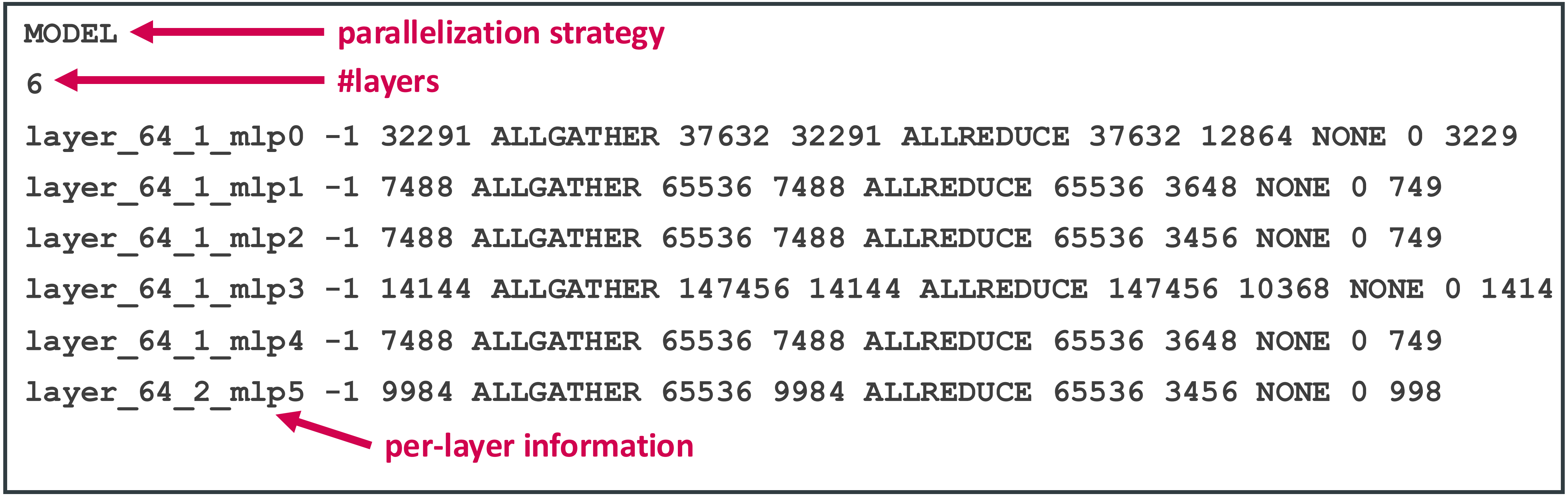
```
ALL_GATHER.0.et  ALL_TO_ALL.0.et  BROADCAST.0.et       one_comm_recv_node.0.et  one_comp_node.0.et                one_remote_mem_load_node.0.et   two_comp_nodes_dependent.0.et
ALL_GATHER.1.et  ALL_TO_ALL.1.et  BROADCAST.1.et       one_comm_recv_node.1.et  one_comp_node.1.et                one_remote_mem_load_node.1.et   two_comp_nodes_dependent.1.et
ALL_GATHER.2.et  ALL_TO_ALL.2.et  BROADCAST.2.et       one_comm_recv_node.2.et  one_comp_node.2.et                one_remote_mem_load_node.2.et   two_comp_nodes_dependent.2.et
ALL_GATHER.3.et  ALL_TO_ALL.3.et  BROADCAST.3.et       one_comm_recv_node.3.et  one_comp_node.3.et                one_remote_mem_load_node.3.et   two_comp_nodes_dependent.3.et
ALL_GATHER.4.et  ALL_TO_ALL.4.et  BROADCAST.4.et       one_comm_recv_node.4.et  one_comp_node.4.et                one_remote_mem_load_node.4.et   two_comp_nodes_dependent.4.et
ALL_GATHER.5.et  ALL_TO_ALL.5.et  BROADCAST.5.et       one_comm_recv_node.5.et  one_comp_node.5.et                one_remote_mem_load_node.5.et   two_comp_nodes_dependent.5.et
ALL_GATHER.6.et  ALL_TO_ALL.6.et  BROADCAST.6.et       one_comm_recv_node.6.et  one_comp_node.6.et                one_remote_mem_load_node.6.et   two_comp_nodes_dependent.6.et
ALL_GATHER.7.et  ALL_TO_ALL.7.et  BROADCAST.7.et       one_comm_recv_node.7.et  one_comp_node.7.et                one_remote_mem_load_node.7.et   two_comp_nodes_dependent.7.et
ALL_REDUCE.0.et  BARRIER.0.et     REDUCE_SCATTER.0.et  one_comm_send_node.0.et  one_metadata_node_all_types.0.et  one_remote_mem_store_node.0.et  two_comp_nodes_independent.0.et
ALL_REDUCE.1.et  BARRIER.1.et     REDUCE_SCATTER.1.et  one_comm_send_node.1.et  one_metadata_node_all_types.1.et  one_remote_mem_store_node.1.et  two_comp_nodes_independent.1.et
ALL_REDUCE.2.et  BARRIER.2.et     REDUCE_SCATTER.2.et  one_comm_send_node.2.et  one_metadata_node_all_types.2.et  one_remote_mem_store_node.2.et  two_comp_nodes_independent.2.et
ALL_REDUCE.3.et  BARRIER.3.et     REDUCE_SCATTER.3.et  one_comm_send_node.3.et  one_metadata_node_all_types.3.et  one_remote_mem_store_node.3.et  two_comp_nodes_independent.3.et
ALL_REDUCE.4.et  BARRIER.4.et     REDUCE_SCATTER.4.et  one_comm_send_node.4.et  one_metadata_node_all_types.4.et  one_remote_mem_store_node.4.et  two_comp_nodes_independent.4.et
ALL_REDUCE.5.et  BARRIER.5.et     REDUCE_SCATTER.5.et  one_comm_send_node.5.et  one_metadata_node_all_types.5.et  one_remote_mem_store_node.5.et  two_comp_nodes_independent.5.et
ALL_REDUCE.6.et  BARRIER.6.et     REDUCE_SCATTER.6.et  one_comm_send_node.6.et  one_metadata_node_all_types.6.et  one_remote_mem_store_node.6.et  two_comp_nodes_independent.6.et
ALL_REDUCE.7.et  BARRIER.7.et     REDUCE_SCATTER.7.et  one_comm_send_node.7.et  one_metadata_node_all_types.7.et  one_remote_mem_store_node.7.et  two_comp_nodes_independent.7.et
```

Generated sets of Chakra ET Files

Text-to-Chakra Wrapper

- **ASTRA-sim1.0's text-based** end-to-end workload representation

*./demo2/*text_workloads/MLP_ModelParallel.txt



```
MODEL
6
layer_64_1_mlp0 -1 32291 ALLGATHER 37632 32291 ALLREDUCE 37632 12864 NONE 0 3229
layer_64_1_mlp1 -1 7488 ALLGATHER 65536 7488 ALLREDUCE 65536 3648 NONE 0 749
layer_64_1_mlp2 -1 7488 ALLGATHER 65536 7488 ALLREDUCE 65536 3456 NONE 0 749
layer_64_1_mlp3 -1 14144 ALLGATHER 147456 14144 ALLREDUCE 147456 10368 NONE 0 1414
layer_64_1_mlp4 -1 7488 ALLGATHER 65536 7488 ALLREDUCE 65536 3648 NONE 0 749
layer_64_2_mlp5 -1 9984 ALLGATHER 65536 9984 ALLREDUCE 65536 3456 NONE 0 998
```

parallelization strategy

#layers

per-layer information

Text-to-Chakra Wrapper

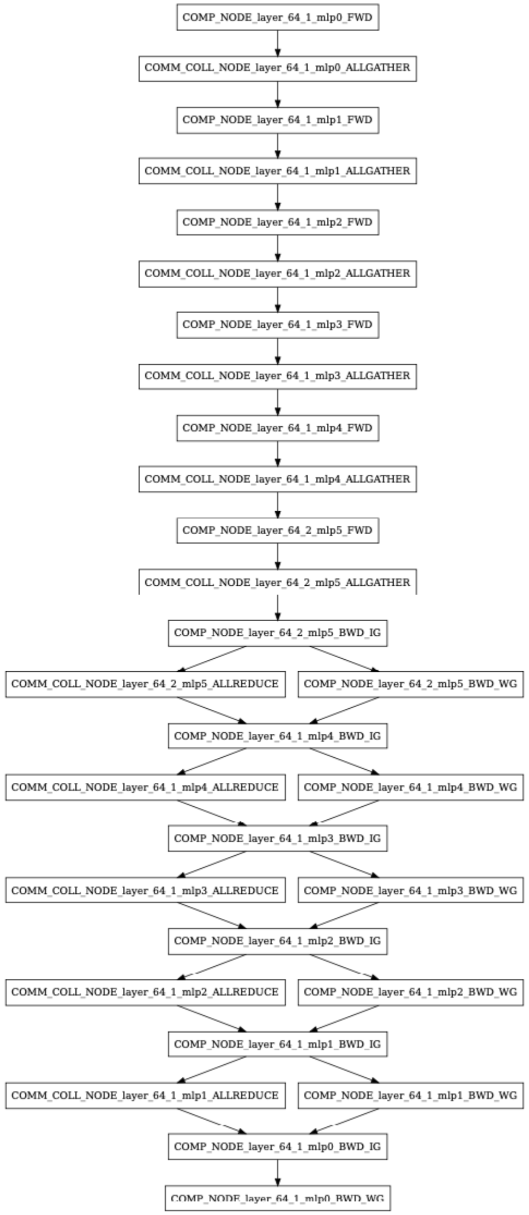
- **ASTRA-sim1.0's text-based** end-to-end workload representation

```
[docker]$ cd ../demo2
[docker]$ ./run_demo2.sh
```

Using Text-to-Chakra Wrapper

- Visualization Result

```
[docker]$ ./visualize.sh
```

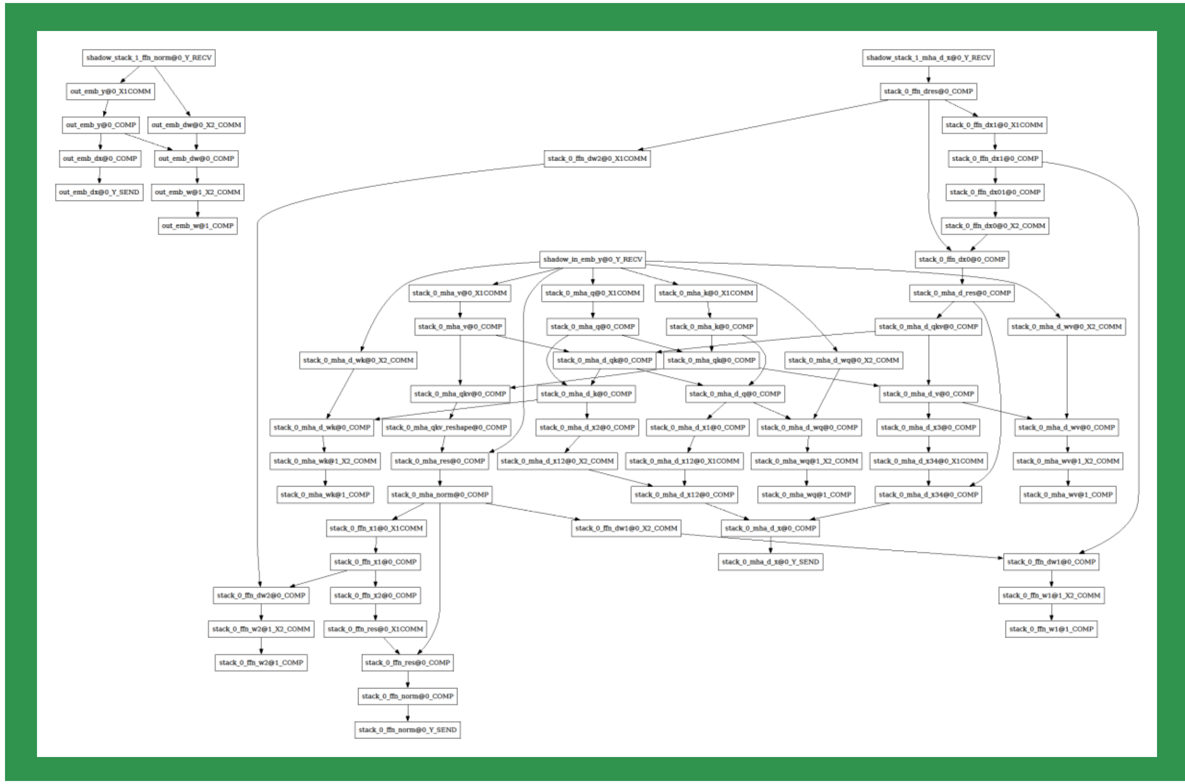

COMP_NODE_layer_64_1_mlp0_FWD
COMM_COLL_NODE_layer_64_1_mlp0_ALLGATHER
COMP_NODE_layer_64_1_mlp1_FWD
COMM_COLL_NODE_layer_64_1_mlp1_ALLGATHER
COMP_NODE_layer_64_1_mlp2_FWD
COMM_COLL_NODE_layer_64_1_mlp2_ALLGATHER
COMP_NODE_layer_64_1_mlp3_FWD
COMM_COLL_NODE_layer_64_1_mlp3_ALLGATHER
COMP_NODE_layer_64_1_mlp4_FWD
COMM_COLL_NODE_layer_64_1_mlp4_ALLGATHER
COMP_NODE_layer_64_2_mlp5_FWD
COMM_COLL_NODE_layer_64_2_mlp5_ALLGATHER
COMP_NODE_layer_64_2_mlp5_BWD_IG
COMM_COLL_NODE_layer_64_2_mlp5_ALLREDUCE
COMP_NODE_layer_64_2_mlp5_BWD_WG
COMP_NODE_layer_64_1_mlp4_BWD_IG
COMM_COLL_NODE_layer_64_1_mlp4_ALLREDUCE
COMP_NODE_layer_64_1_mlp4_BWD_WG
COMP_NODE_layer_64_1_mlp3_BWD_IG
COMM_COLL_NODE_layer_64_1_mlp3_ALLREDUCE
COMP_NODE_layer_64_1_mlp3_BWD_WG
COMP_NODE_layer_64_1_mlp2_BWD_IG
COMM_COLL_NODE_layer_64_1_mlp2_ALLREDUCE
COMP_NODE_layer_64_1_mlp2_BWD_WG
COMP_NODE_layer_64_1_mlp1_BWD_IG
COMM_COLL_NODE_layer_64_1_mlp1_ALLREDUCE
COMP_NODE_layer_64_1_mlp1_BWD_WG
COMP_NODE_layer_64_1_mlp0_BWD_IG
COMP_NODE_layer_64_1_mlp0_BWD_WG

Visualization Result

Synthetic Chakra ET Generator

- STG can produce synthetic large language model (LLM) traces.

- Supports multiple parallelism strategies, including:
    - Data Parallelism (DP),
    - Tensor Parallelism (TP)
    - Pipeline Parallelism (PP)
    - Sequence Parallelism (SP).



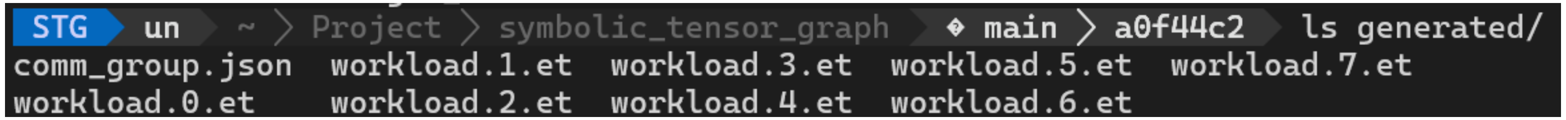
Visualization Result

Public repository: https://github.com/astra-sim/symbolic_tensor_graph

Synthetic Chakra ET Generator

- Generate synthetic workload traces with various parallel strategy exploration

```
[docker]$ cd ../demo3
[docker]$ ./run_demo3.sh
```



```
STG  un  ~ > Project > symbolic_tensor_graph  ◆ main > a0f44c2  ls generated/
comm_group.json  workload.1.et  workload.3.et  workload.5.et  workload.7.et
workload.0.et    workload.2.et  workload.4.et  workload.6.et
```

Chakra ET Collection

- Profile/Collect Real System Trace from PyTorch

```
et = ExecutionGraphObserver()
et.register_callback("et_file.json")
et.start()

# run PyTorch model

et.stop()
et.unregister_callback()
```

Start
ET collection

Run model

Stop collection

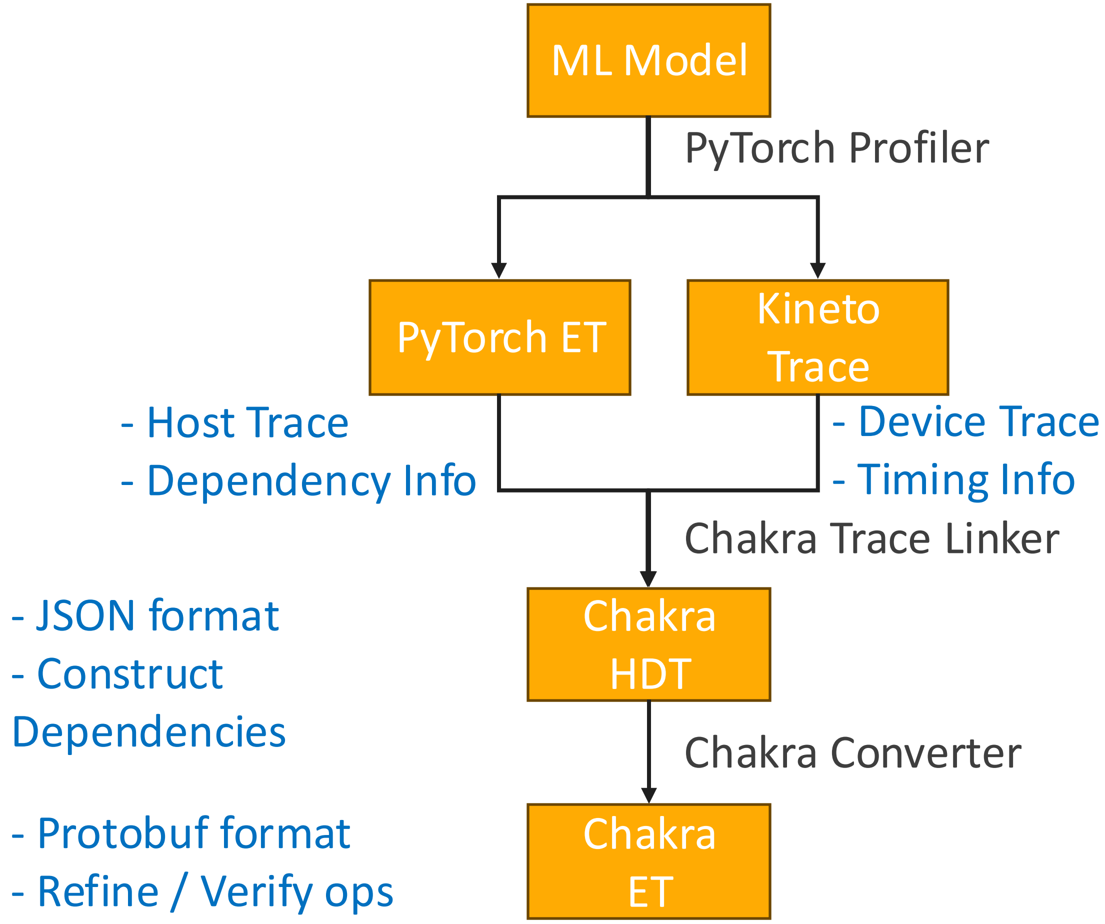
Collected Traces

- PyTorch Execution Trace (Host)

- Kineto Trace

PyTorch Trace: Flow

- Collect traces:
  - PyTorch Execution (Host) Trace
  - Kineto (device) Trace
- Link traces into:
  - Chakra Host-Device Trace (JSON)
- Convert into:
  - Chakra Execution Trace (Protobuf)



ML Model

PyTorch Profiler

PyTorch ET

Kineto Trace

- Host Trace
- Dependency Info

- Device Trace
- Timing Info

Chakra Trace Linker

Chakra HDT

- JSON format
- Construct Dependencies

Chakra Converter

Chakra ET

- Protobuf format
- Refine / Verify ops

Merging Traces

- Merge Host and Device traces into **Chakra HDT (JSON)**

```
[docker]$ ./merge.sh
```

demo3/merge.sh:

```
python3 -m chakra.et_converter.et_converter \
    --input_type="PyTorch" \
    --input_filename="${SCRIPT_DIR}/etplus_traces/etplus_0.json" \
    --output_filename="${SCRIPT_DIR}/chakra_traces/et.0.et"
```

Converting into Charka ET

- Convert Chakra HDT (JSON) into Chakra ET (Protobuf)

```
[docker]$ ./convert.sh
```

demo3/merge.sh:

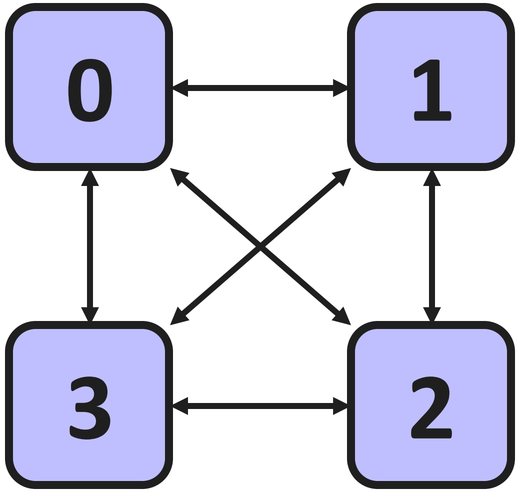
```
python3 -m chakra.et_converter.et_converter \
    --input_type="PyTorch" \
    --input_filename="${SCRIPT_DIR}/etplus_traces/etplus_0.json" \
    --output_filename="${SCRIPT_DIR}/chakra_traces/et.0.et"
```

Outline

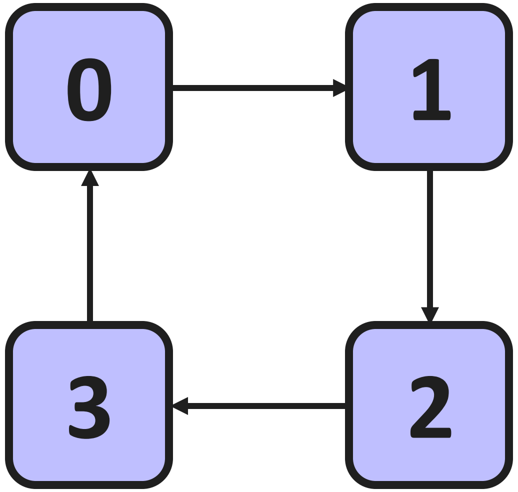
- Prerequisites
- Part 1: Generating traces with Chakra
- **Part 2: Running ASTRA-Sim with various configurations**
  - **(ASTRA-Sim Demo 1) System Layer**
  - **(ASTRA-Sim Demo 2) Network Layer - Analytical Backend**
  - **(ASTRA-Sim Demo 3) Network Layer – NS3 Backend**

System Layer – Collective algorithms

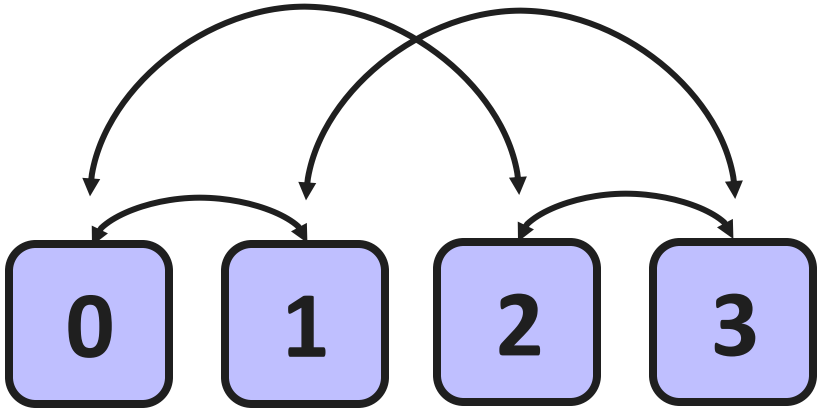
- ASTRA-Sim supports configuring different collective algorithms.



Direct



Ring



Halving-Doubling

System Layer – Changing Collectives

demo1/inputs/direct_sys.json

```
"scheduling-policy": "LIFO",
"endpoint-delay": 10,
"active-chunks-per-dimension": 1,
"preferred-dataset-splits": 4,
"all-reduce-implementation": ["direct"],
"all-gather-implementation": ["direct"],
"reduce-scatter-implementation": ["direct"],
"all-to-all-implementation": ["direct"],
"collective-optimization": "localBWAware",
"local-mem-bw": 50,
"boost-mode": 0
```

4 chunks per collective

Direct algorithm

Running Simulation

- Execute ASTRA-sim Simulation

```
[docker]$ cd ../astra-sim-demo/demo1
[docker]$ ./run_demo1-1.sh
```

run_demo1-1.sh:

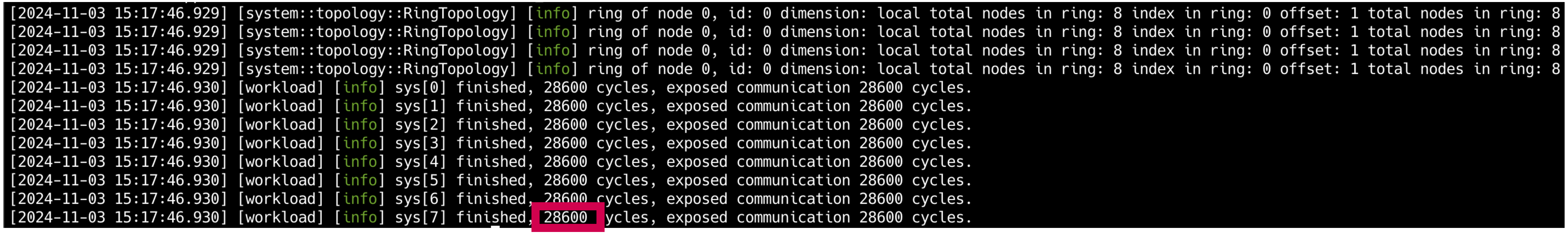
```
${ASTRA_SIM} \
    --workload-configuration=./allreduce/allreduce \
    --system-configuration=./inputs/Direct_sys.json \
    --network-configuration=./inputs/Direct_8.yml \
```

Results

- Simulate All-Reduce with Direct algorithm in ASTRA-Sim

- The result will show execution time and exposed communication time

Expected Result



```
[2024-11-03 15:17:46.929] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:17:46.929] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:17:46.929] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:17:46.929] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:17:46.930] [workload] [info] sys[0] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[1] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[2] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[3] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[4] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[5] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[6] finished, 28600 cycles, exposed communication 28600 cycles.
[2024-11-03 15:17:46.930] [workload] [info] sys[7] finished, 28600 cycles, exposed communication 28600 cycles.
```

Simulation Result: 28.6 µs

System Layer – Changing Collectives

demo1/inputs/Ring_sys.json

```
"scheduling-policy": "LIFO",
"endpoint-delay": 10,
"active-chunks-per-dimension": 1,
"preferred-dataset-splits": 4,
"all-reduce-implementation": ["ring"],
"all-gather-implementation": ["ring"],
"reduce-scatter-implementation": ["ring"],
"all-to-all-implementation": ["ring"],
"collective-optimization": "localBWAware",
"local-mem-bw": 50,
"boost-mode": 0
```

4 chunks per collective

Ring algorithm

Running Simulation

- Execute ASTRA-sim Simulation

```
[docker]$ ./run_demo1-2.sh
```

run_demo1-1.sh:

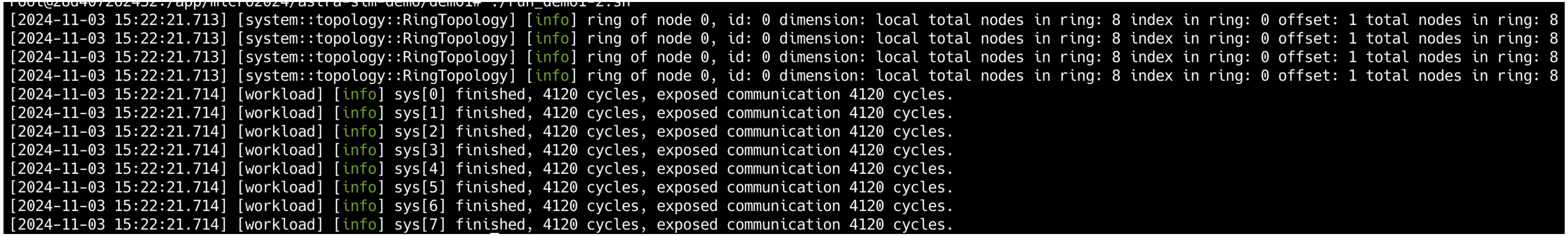
```
${ASTRA_SIM} \
    --workload-configuration=./allreduce/allreduce \
    --system-configuration=./inputs/Ring_sys.json \
    --network-configuration=./inputs/Ring_8.yml \
```

Results

- Simulate All-Reduce with Ring algorithm in ASTRA-Sim

- The result will show execution time and exposed communication time

Expected Result



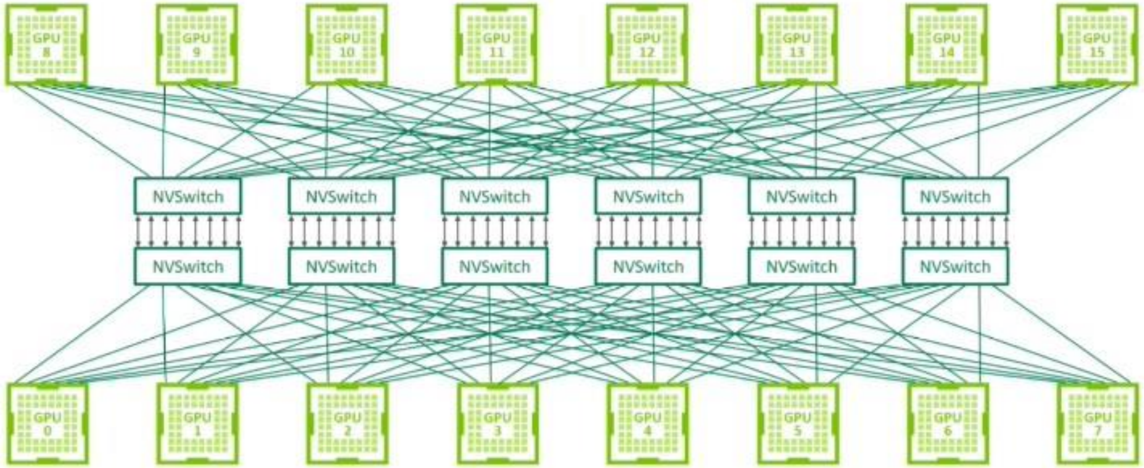
```
root@28d407262432:/app/micro2024/astra-sim-demo/demo1# ./run_demo1-2.sh
[2024-11-03 15:22:21.713] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:22:21.713] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:22:21.713] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:22:21.713] [system::topology::RingTopology] [info] ring of node 0, id: 0 dimension: local total nodes in ring: 8 index in ring: 0 offset: 1 total nodes in ring: 8
[2024-11-03 15:22:21.714] [workload] [info] sys[0] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[1] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[2] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[3] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[4] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[5] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[6] finished, 4120 cycles, exposed communication 4120 cycles.
[2024-11-03 15:22:21.714] [workload] [info] sys[7] finished, 4120 cycles, exposed communication 4120 cycles.
```

Network Layer – Analytical Backend

- Modeling **DGX H100**
- **2-level Switch** interconnect topology with **32 GPUs**
- **500 ns** (latency), **450 GB/s** (NVLink), **50 GB/s** (Infiniband)

demo2/inputs/Switch_8.yml:

```
topology: [ Switch, Switch ]
npus_count: [ 8, 4 ] # 32 GPUs
bandwidth: [ 450, 50 ]  # GB/s
latency: [ 500.0, 500.0 ]  # ns
```



Source: https://www.nextplatform.com/2024/05/30/key-hyperscalers-and-chip-makers-gang-up-on-nvidias-nvswitch-interconnect/

Running Simulation

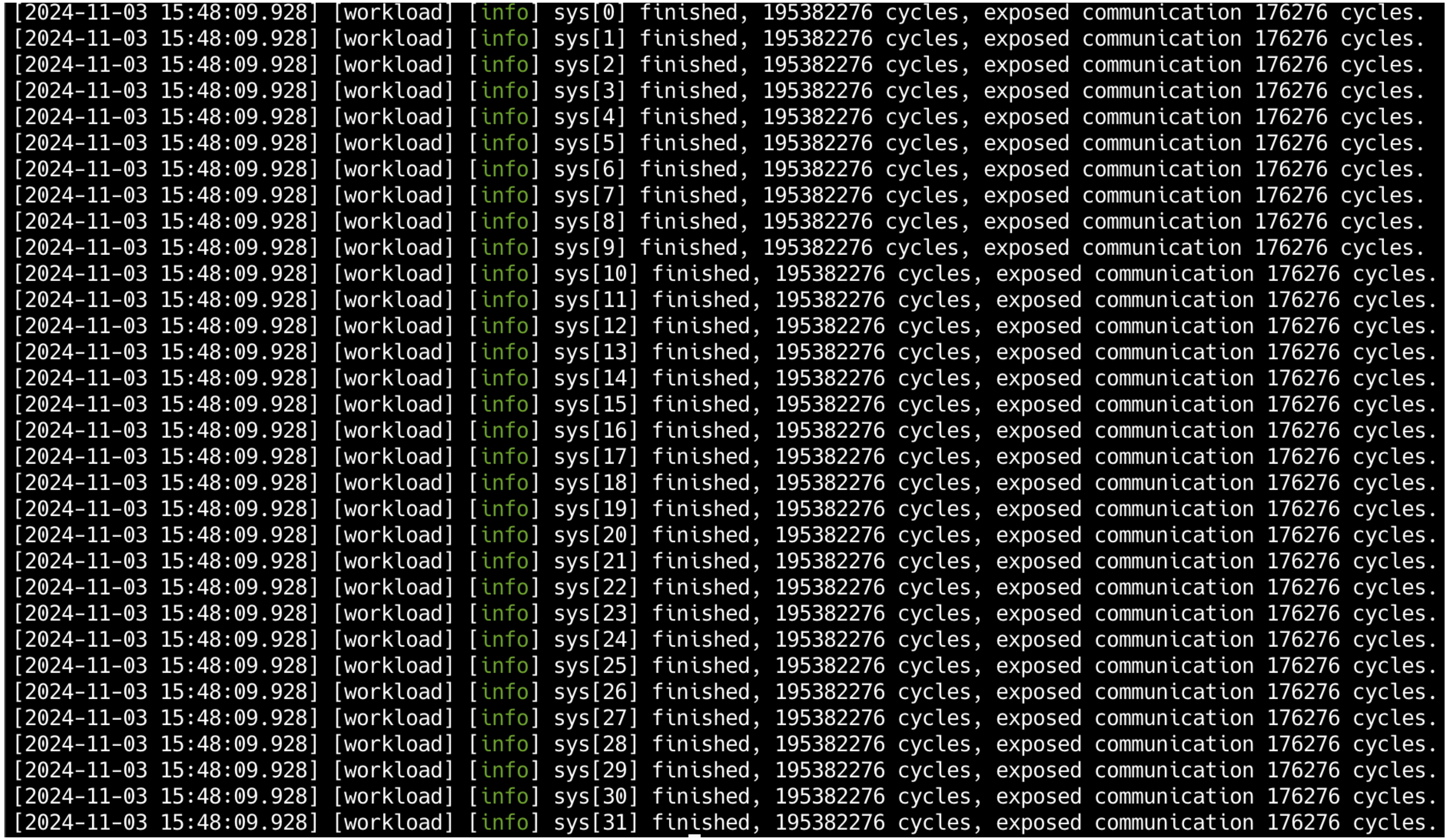
- Execute ASTRA-sim Simulation

```
[docker]$ ./run_demo2-1.sh
```

run_demo1-1.sh:

```
${ASTRA_SIM} \
    --workload-configuration=./workload/MLP_ModelParallel \
    --system-configuration=./inputs/DGX_H100.json \
    --network-configuration=./inputs/DGX_H100-32.yml \
```

Demo 1-1: Simulation Result



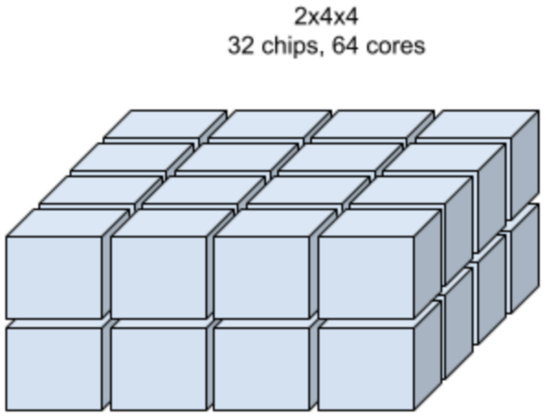
```
[2024-11-03 15:48:09.928] [workload] [info] sys[0] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[1] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[2] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[3] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[4] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[5] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[6] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[7] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[8] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[9] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[10] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[11] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[12] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[13] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[14] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[15] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[16] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[17] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[18] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[19] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[20] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[21] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[22] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[23] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[24] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[25] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[26] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[27] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[28] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[29] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[30] finished, 195382276 cycles, exposed communication 176276 cycles.
[2024-11-03 15:48:09.928] [workload] [info] sys[31] finished, 195382276 cycles, exposed communication 176276 cycles.
```

Network Layer – Analytical Backend

- Modeling **TPUv4**
- **3D torus** interconnect topology with **32 TPUs**
- **500 ns** (latency), **50 GB/s** (bandwidth)

demo2/inputs/Switch_8.yml:

```
topology: [ Ring, Ring, Ring ]
npus_count: [ 2, 4, 4 ]
bandwidth: [ 50, 50, 50 ]  # GB/s
latency: [ 500.0, 500.0, 500.0] # ns
```



TPUv4
3D torus

Running Simulation

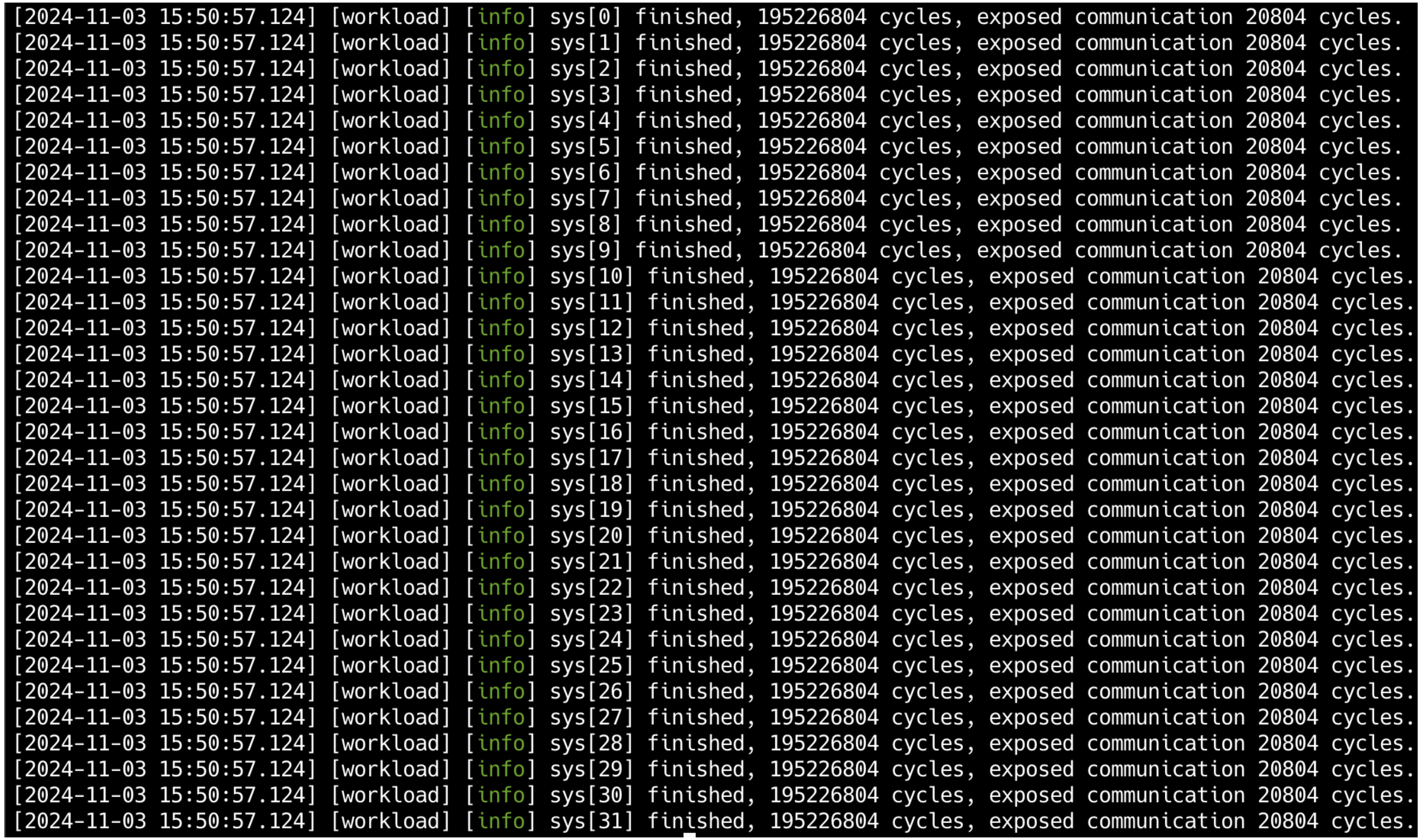
- Execute ASTRA-sim Simulation

```
[docker]$ ./run_demo2-2.sh
```

run_demo2-1.sh:

```
${ASTRA_SIM} \
    --workload-configuration=./allreduce/allreduce \
    --system-configuration=./inputs/Ring_sys.json \
    --network-configuration=./inputs/Ring_8.yml \
```

Running Simulation



```
[2024-11-03 15:50:57.124] [workload] [info] sys[0] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[1] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[2] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[3] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[4] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[5] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[6] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[7] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[8] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[9] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[10] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[11] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[12] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[13] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[14] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[15] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[16] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[17] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[18] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[19] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[20] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[21] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[22] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[23] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[24] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[25] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[26] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[27] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[28] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[29] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[30] finished, 195226804 cycles, exposed communication 20804 cycles.
[2024-11-03 15:50:57.124] [workload] [info] sys[31] finished, 195226804 cycles, exposed communication 20804 cycles.
```

Build and Install

- ASTRA-Sim can use NS-3 as network backend
- Compile ASTRA-sim with the analytical ns-3 backend

```
[docker]$ ./compile_astra_sim.sh
```

Running Simulation: 1D Ring across Fat-Tree

- Execute ASTRA-sim Simulation on NS3

```
[docker]$ cd./run_demo3-1.sh
```

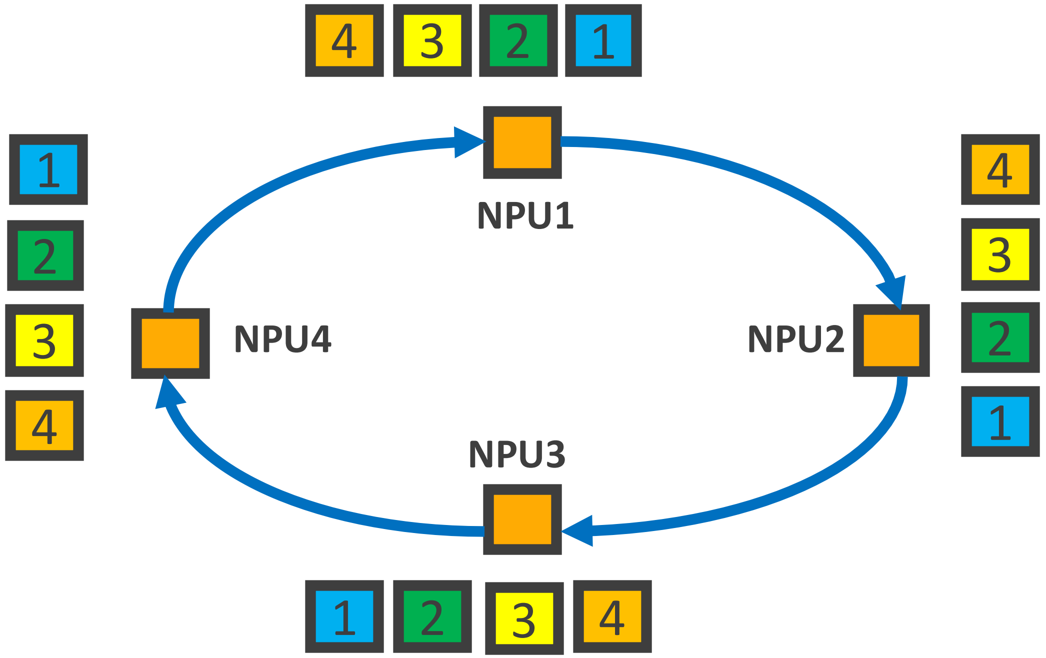
run_demo3-1.sh:

```
cd ${NS3_DIR}
./ns3.42-AstraSimNetwork-default\
    --workload-configuration=./allreduce/allreduce \
    --system-configuration=./inputs/Ring_sys.json \
    --network-configuration=../../../ns-3/scratch/config.txt \
    --logical-topology=./inputs/128nodes_1D.json
```

Note, For NS-3, we have a new option, '**logical-topology**'

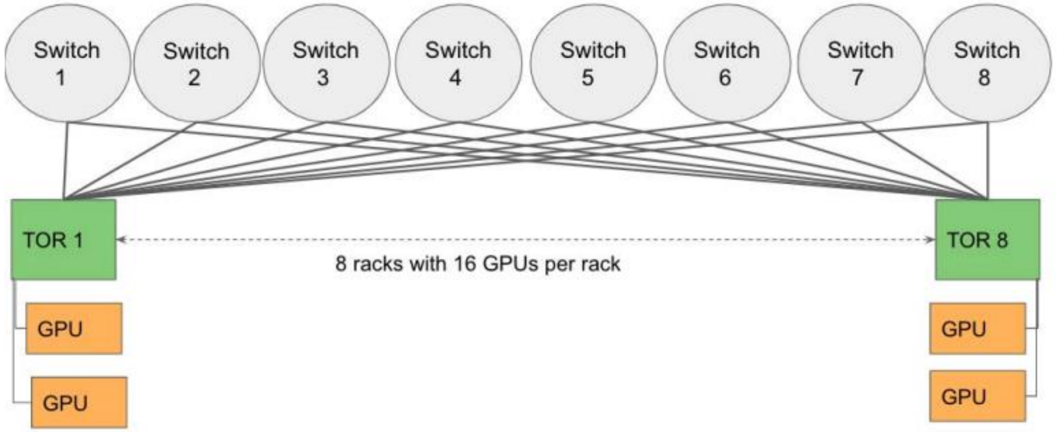
Logical Topology v. Physical Topology

Logical Topology



Which NPUs will NPU X communicate with?

Physical Topology



Actual connectivity between wires, switches, etc.

Logical Topology v. Physical Topology

Logical Topology

demo4/inputs/128_nodes_1D.json:

```
{
    "logical-dims": ["128"]
}
```

demo4/inputs/128_nodes_2D.json:

```
{
    "logical-dims": ["8", "16"]
}
```

Network(ns-3) configuration

demo4/inputs/config_clos.txt:

```
TOPOLOGY_FILE \
../8_nodes_1_switch.txt
```

Physical Topology

demo4/inputs/8_nodes_1_switch.txt:

```
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
…
```

Logical Topology v. Physical Topology

In Analytical backend, logical dimensions automatically matches physical dimension

In NS-3, we decouple the logical dimension and the physical dimension

- e.g. We could run a 1D Ring AllReduce across all 128 nodes in a physical Fat-Tree topology

  And compare with a 2D Ring AllReduce

Physical Topology Setup

demo3/inputs/8_nodes_1_switch.txt:

Total # node(NPU) + Switch

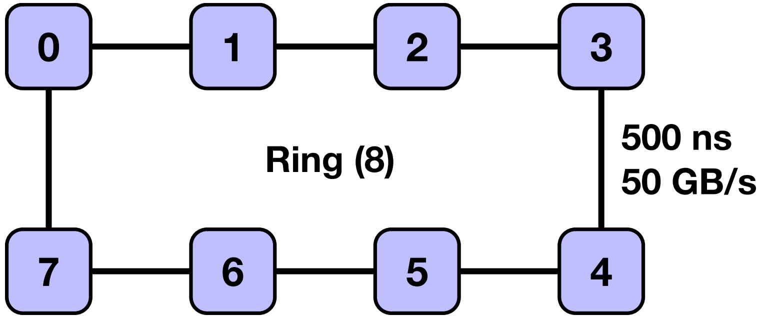
#Switches

#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

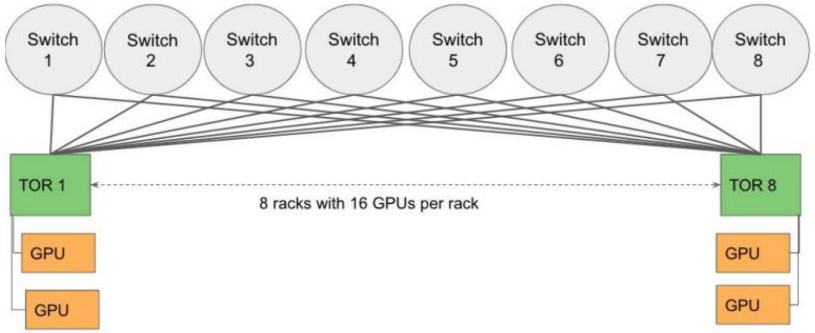
demo3/inputs/128_nodes_16_switch.txt:

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
```



Running Simulation: 2D Ring across Fat-Tree

- Execute ASTRA-sim Simulation on NS3

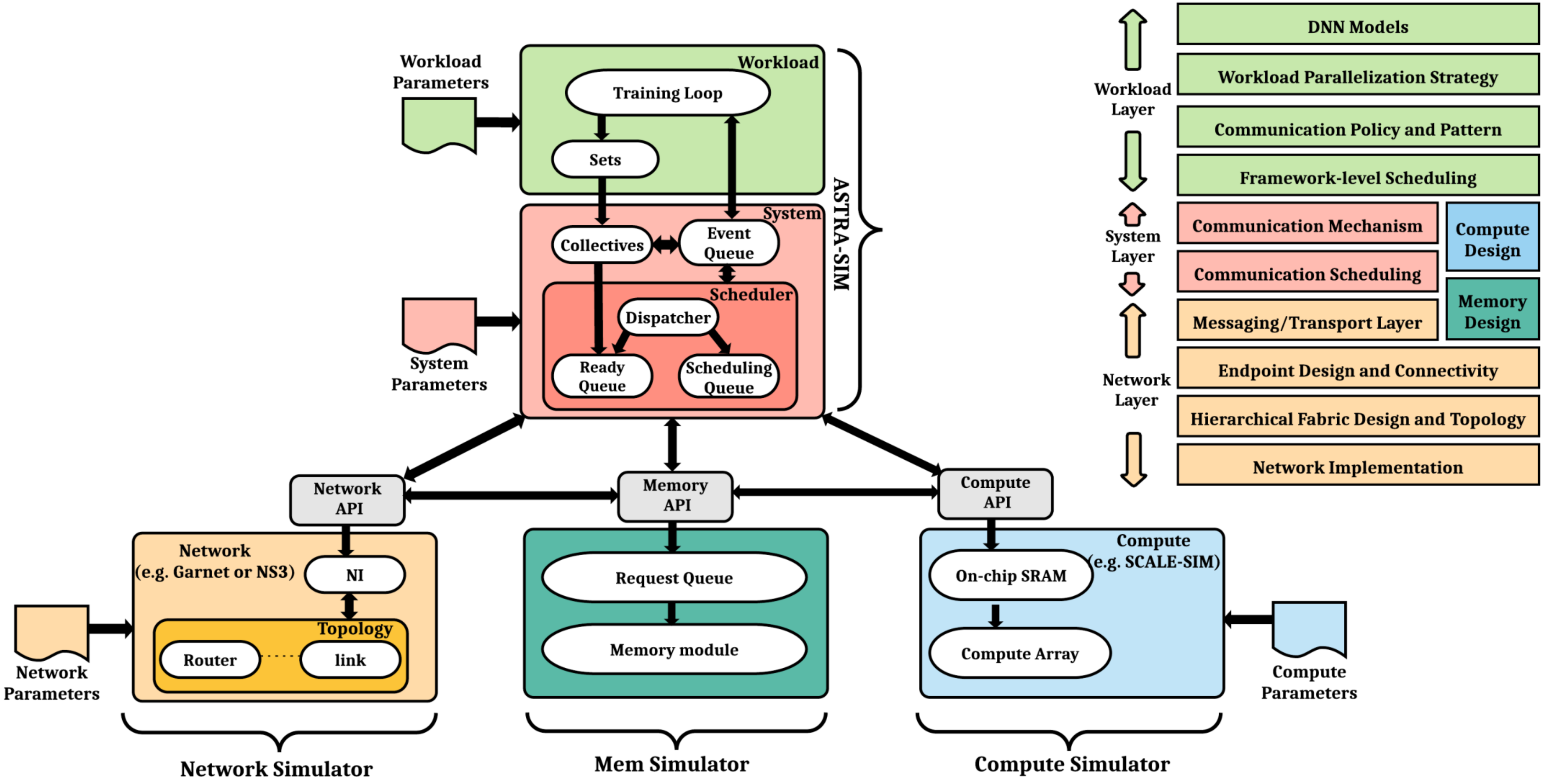
```
[docker]$ ./run_demo3-3.sh
```

run_demo3-3.sh:

```
cd ${NS3_DIR}
./ns3.42-AstraSimNetwork-default\
    --workload-configuration=./allreduce/allreduce \
    --system-configuration=./inputs/Ring_Ring_sys.json \
    --network-configuration=../../../ns-3/scratch/config_clos.txt \
    --logical-topology=./inputs/128nodes_1D.json
```

Must use relative directory in script for network config

Thank you!


Workload Parameters
Workload
Training Loop
Sets
System
Collectives
Event Queue
Scheduler
Dispatcher
Ready Queue
Scheduling Queue
System Parameters
ASTRA-SIM
Network API
Memory API
Compute API
Network (e.g. Garnet or NS3)
NI
Topology
Router
link
Network Parameters
Request Queue
Memory module
Compute (e.g. SCALE-SIM)
On-chip SRAM
Compute Array
Compute Parameters
Network Simulator
Mem Simulator
Compute Simulator
Workload Layer
System Layer
Network Layer
DNN Models
Workload Parallelization Strategy
Communication Policy and Pattern
Framework-level Scheduling
Communication Mechanism
Communication Scheduling
Messaging/Transport Layer
Compute Design
Memory Design
Endpoint Design and Connectivity
Hierarchical Fabric Design and Topology
Network Implementation