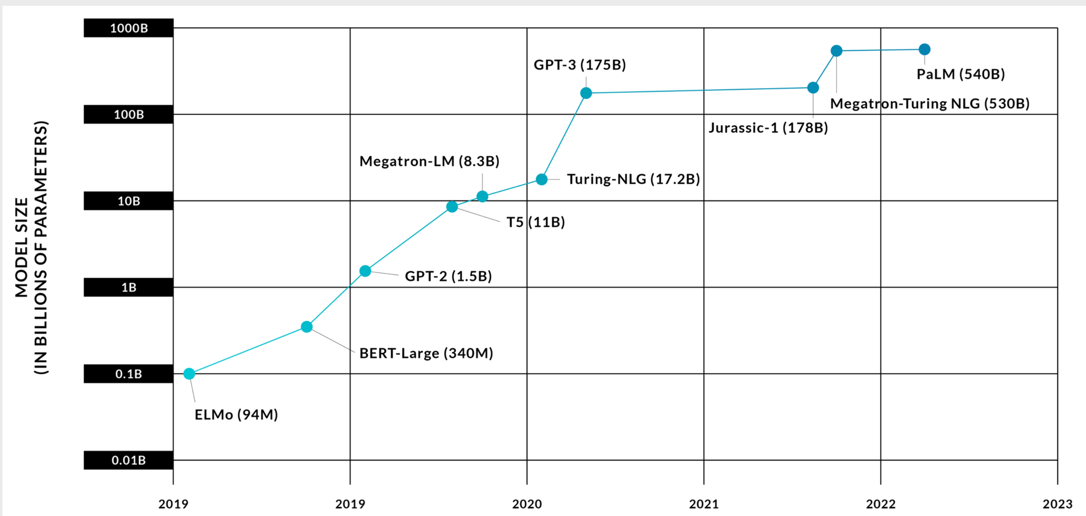
ASTRA-sim2.0: Modeling Hierarchical Networks and Disaggregated Systems for Large-model Training at Scale

William Won[1], Taekyung Heo[2], Saeed Rashidi[3], Srinivas Sridharan[2], Sudarshan Srinivasan[4], Tushar Krishna[1]
[1]Georgia Institute of Technology, [2]NVIDIA, [3]HP Labs, [4]Intel

Georgia Tech College of Computing
Center for Research into Novel Computing Hierarchies

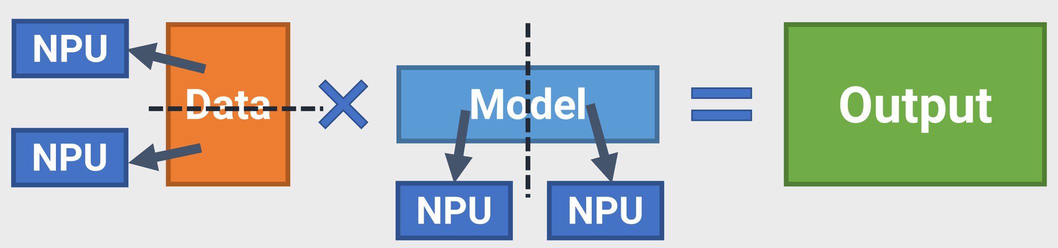
Distributed ML

Machine Learning (ML) models and training data are scaling quickly
- 2× / 3.4 months

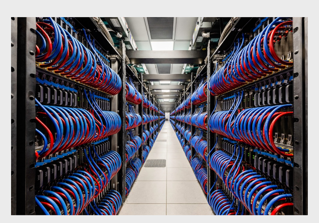


Single-NPU execution is impractical
- 355 years to train GPT-3 on NVIDIA V100

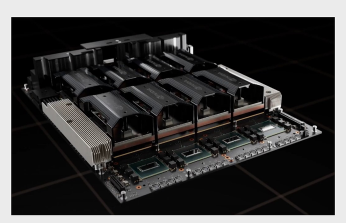
Distributed ML is necessitated
- Model Parallel: Shard model to fit on NPU
- Data Parallel: Shard input data to speedup



State-of-the-art ML Clusters

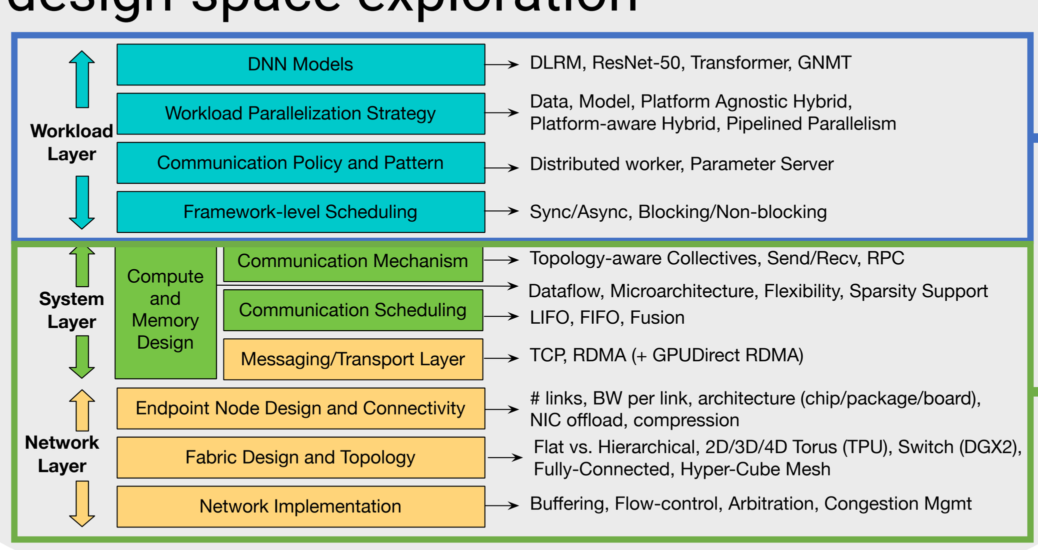

Intel Aurora


Google TPUv4


NVIDIA HGX-H100

Design Space

Design-space of Distributed ML is vast and complex
- Workload, System, and Network Layers
- Needs a methodology for swift design-space exploration



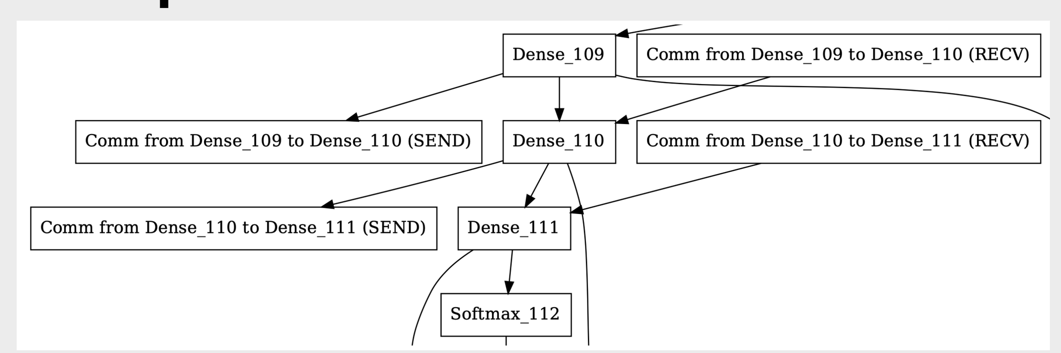
Chakra Execution Trace

Chakra ET: Represents arbitrary distributed ML workloads
- In Directed Acyclic Graph (DAG) format

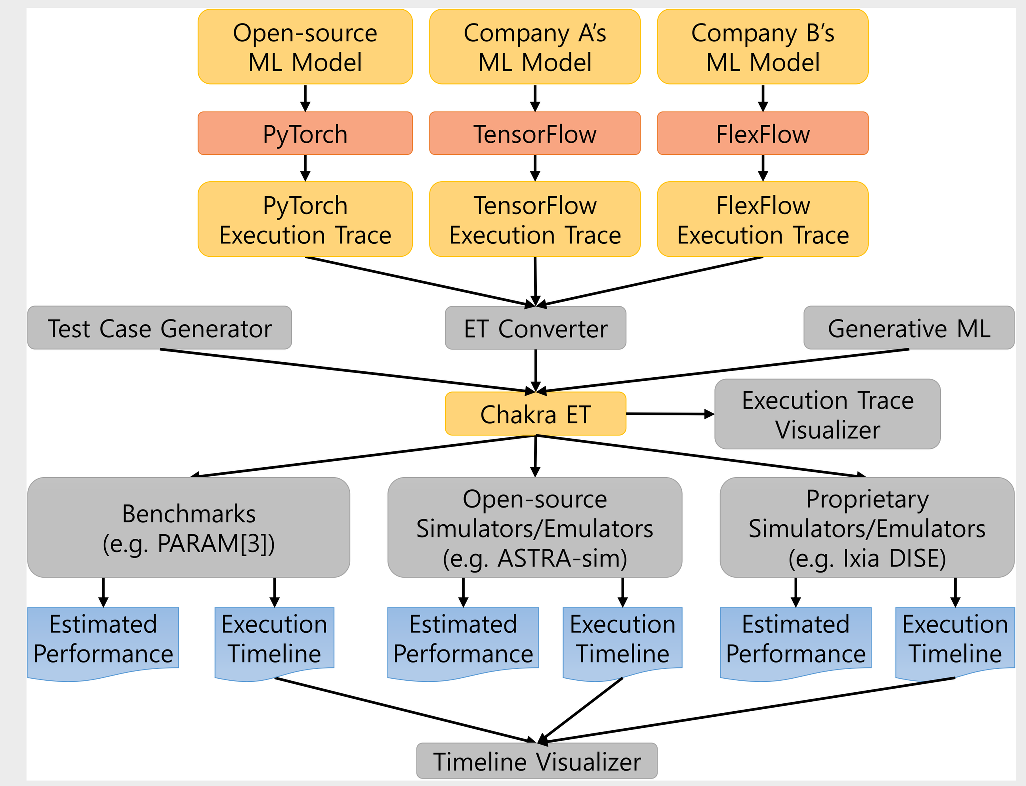
Three types of Chakra nodes
- Compute: Operand size or runtime
- Communication: Collective or Peer-to-peer
- Memory: Local or Remote memory access

Compute	Communication	Memory
GEMM 50 µs	All-Reduce 200 MB	Remote 150 MB

Example Chakra Execution Trace



Workflow of Chakra Execution Trace

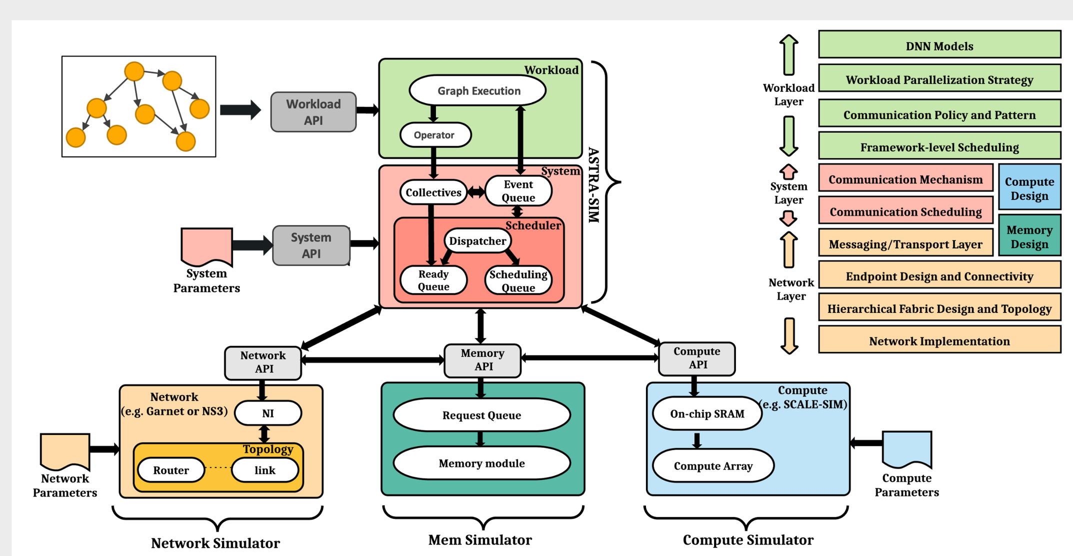


Advantages of Chakra Execution Trace
- Flexible to capture arbitrary ML schemes
- Suitable for benchmark, replay, simulation
- Obfuscates key proprietary workload data
- Facilitates the exchange of ML traces
- Being standardized via MLCommons

Chakra MLCommons *Chakra Paper* *ASTRA-sim Website* *ASTRA-sim2.0 Paper*

ASTRA-sim 2.0

ASTRA-sim2.0 Infrastructure



ASTRA-sim2.0 supports
- Simulation of Chakra Execution Traces
- Analytical network modeling at scale
- Multi-dimensional network topologies
- Memory system modeling via MemoryAPI

Workload API
- Simulates arbitrary Chakra ETs
- Capable of capturing flexible workloads

System API
- Defines system-level characteristics
- Collective algorithms, chunks scheduling

Network API
- Simulates actual network behaviors
- Analytical, Garnet, ns-3

Memory API
- Models memory node of Chakra ET
- Supports in-switch collectives

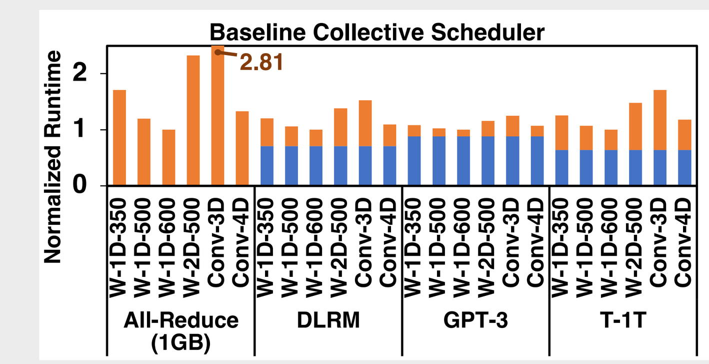
Compute API
- Models compute node of Chakra ET
- Analytical, Roofline, SCALE-sim

Case Studies

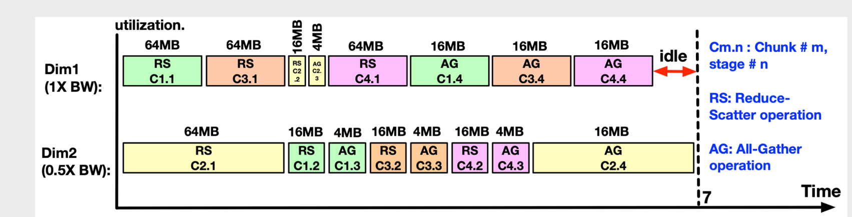
Case 1: Conventional vs. Wafer-scale
- Conventional: High-dimensional (scale-out)
- Wafer-scale: Low-dimensional (scale-up)

Topology	Shape	NPU Size	BW (GB/s)
W-1D	Switch	512	350, 500, 600
W-2D	Switch_Switch	32×16	250_250
Conv-3D	Ring_FC_Switch	16×8×4	200_100_50
Conv-4D	Ring_FC_Ring_Switch	2×8×8×4	250_200_100_50

Case 1 Result
- Wafer-scale system perform better
- Due to the multi-dimensional network overhead

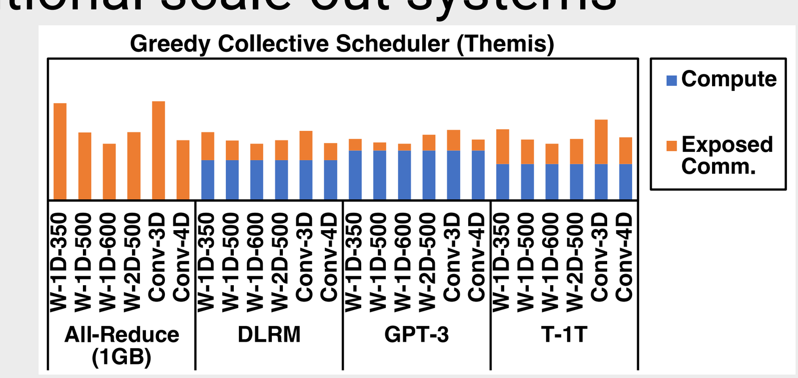


Case 2: Smart Chunk Scheduling
- Themis (ISCA '22): greedy-based chunk scheduler for multi-dimensional networks


Themis Scheduling

Themis Paper

Case 2 Result
- Themis mitigates the overhead of conventional scale-out systems



CONCLUSIONS

Needs for Distributed ML
- Models and training data are scaling
- Single-NPU execution is impractical

Needs for simulation methodology
- For swift design-space exploration
- Of emerging ML platforms

Chakra ET and ASTRA-sim2.0
- Chakra ET: standardized, graph-based distributed ML workload representation
- ASTRA-sim2.0: simulating emerging ML systems