Georgia Tech | School of Electrical and Computer Engineering
College of Engineering

SYNERGY
http://synergy.ece.gatech.edu

ASTRA-SIM
https://astra-sim.github.io

CHAKRA
https://github.com/mlcommons/chakra

Chakra + ASTRA-sim: An ecosystem for advancing benchmarking and co-design for diverse AI supercomputing platforms

Tushar Krishna
Associate Professor, School of ECE
Georgia Institute of Technology
tushar@ece.gatech.edu
https://tusharkrishna.ece.gatech.edu/

ModSim Workshop
August 13, 2025

Acknowledgments

Georgia Tech

Will Won

Jinsun Yoo

Changhai Man

Joongun Park

Meta

Saeed Rashidi

AMD

Brad Beckmann

NVIDIA

Taekyung Heo

Srinivas Sridharan

ML Commons

OPEN Compute Project®

Semiconductor Research Corporation

KEYSIGHT TECHNOLOGIES

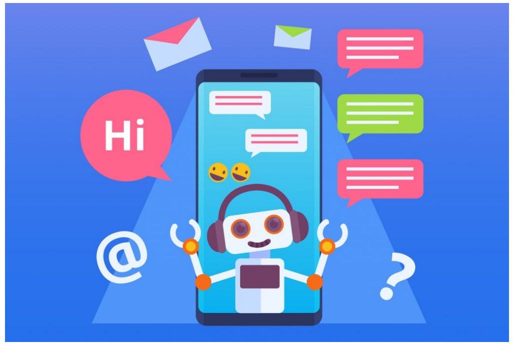
intel

Hewlett Packard Enterprise

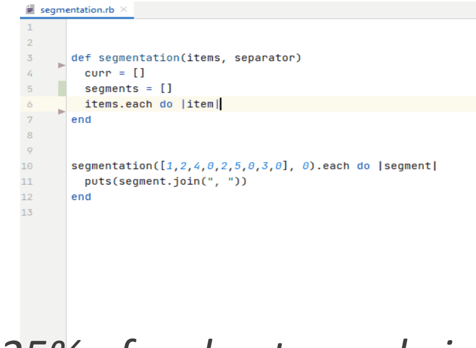
+ many more

AI is pervasive today!

Chatbots

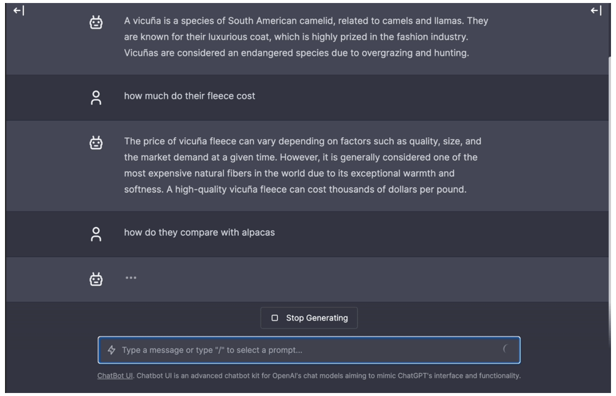


Code Generation

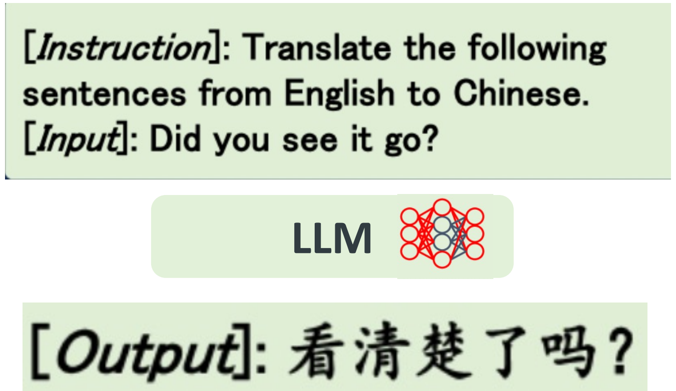


"25% of code at google in last quarter was AI generated "

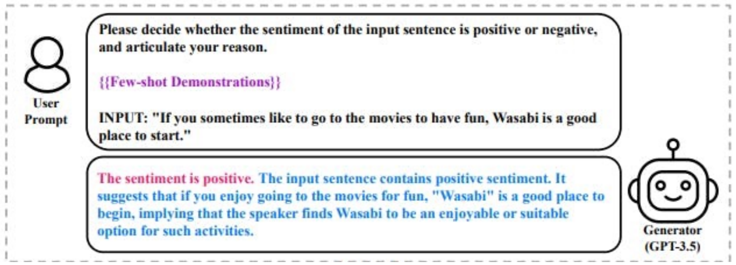
Text Generation



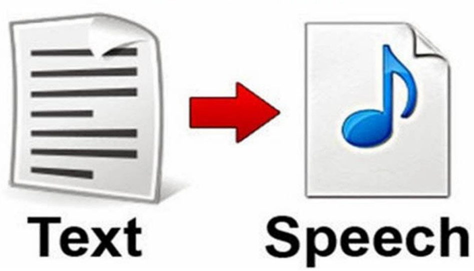
Language Translation



Sentiment Analysis

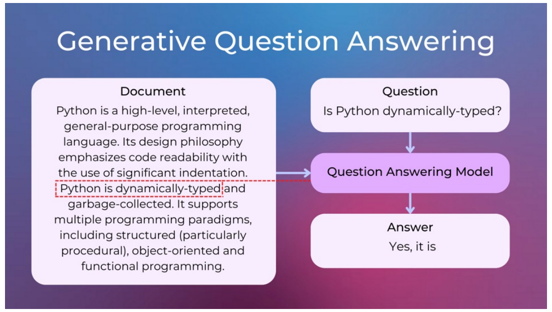


Text to Speech



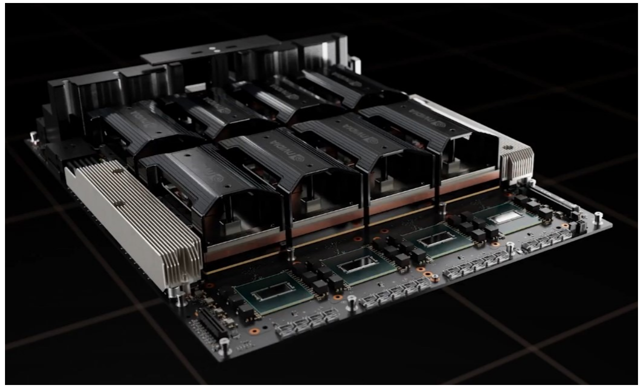
Recommendations

Question Answering

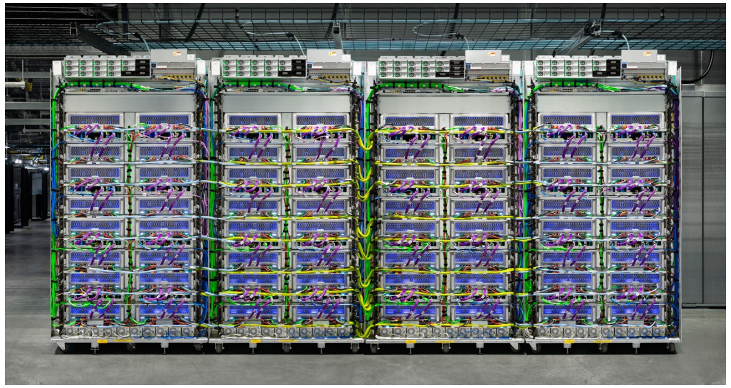


Algorithmic view of AI (Datasets and Models)

Computer Architect’s view of AI



NVIDIA
HGX

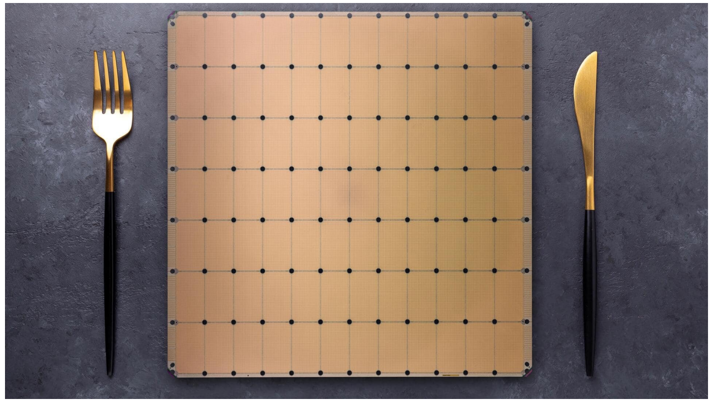


Google
Cloud TPU

SambaNova
SN40L

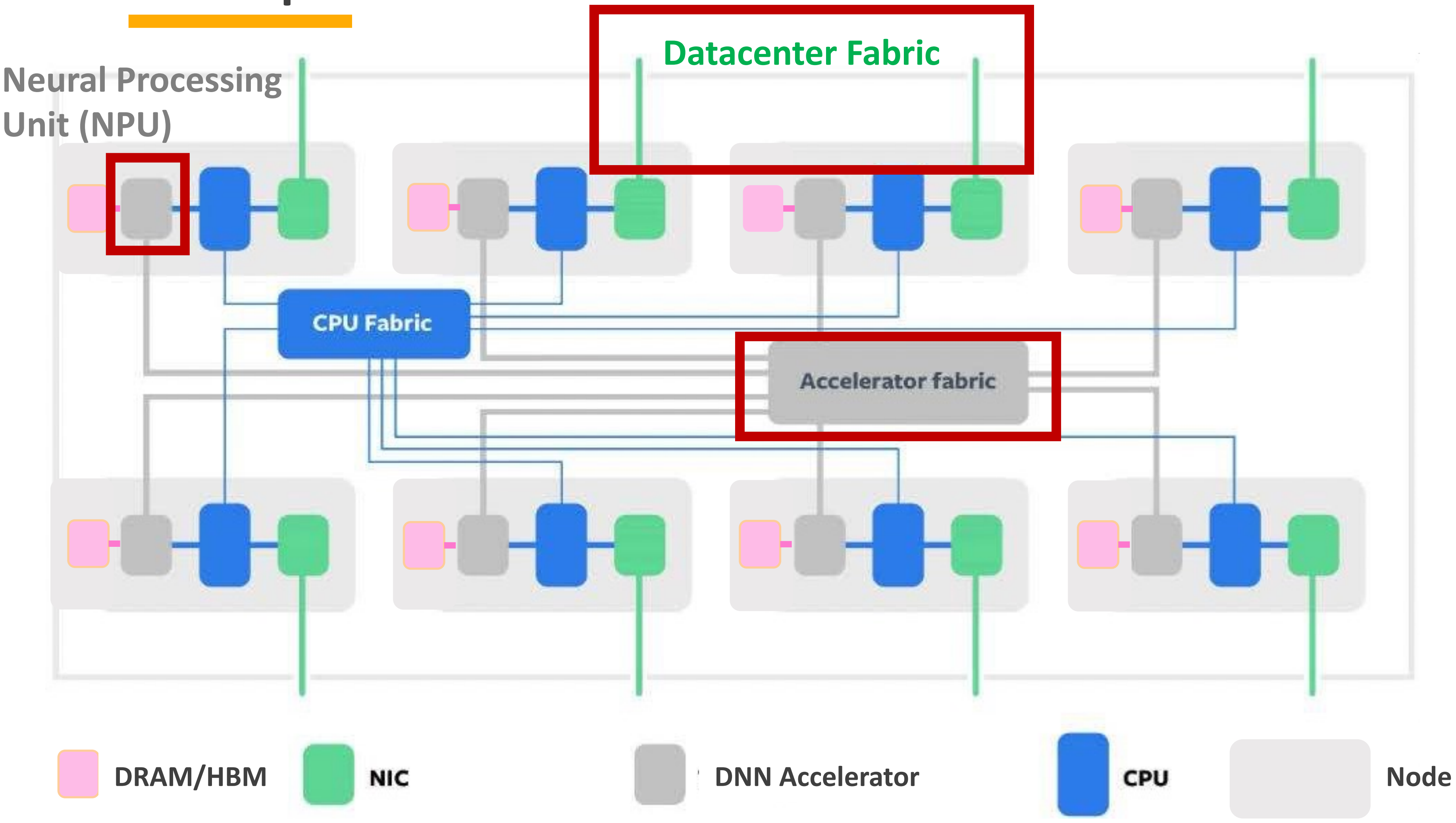
AMD
Instinct Platforms

Intel
Gaudi



Cerebras
Andromeda

Computer Architect's view of AI



- ✓ Customized accelerators for AI (aka NPUs)
- ✓ Customized network fabrics to scale the AI task across multiple accelerators

Why?

*Figure modified from "**Zion: Facebook Next- Generation Large Memory Training Platform", Misha Smelyanskiy, Hot Chips 31"***

“Large” Language Models

Hundreds of ZettaFLOPs of compute

Model (Company)	Company	#Parameters [Billion]	Model Footprint (Assuming 2B/Param)	Training Footprint (Assuming 16B/Param*)
Clause 3 Opus	Anthropic	**2,000**	4.00 TB	32.00 TB
GPT-4	OpenAI	**1,760**	3.52 TB	28.16 TB
Gemini 1.5 Pro	Google	**1,500**	3.00 TB	24.00 TB
Samba-1	SambaNova	**1,400**	2.80 TB	22.40 TB
Cerebras-1T	Cerebras	**1,000**	2.00 TB	16.00 TB
Grok-3	xAI	**928**	1.86 TB	14.85 TB
DeepSeep-R1	DeepSeek-AI	**685**	1.37 TB	10.96 TB
PaLM	Google	**540**	1.08 TB	8.64 TB

https://lifearchitect.ai/models/

**Assuming Mixed-precision Adam Optimizer (See Microsoft ZeRO)*

AI is a distributed systems problem!

Consider GPT-4

- **Total parameters** — ~1.8 trillion (over 10x more than GPT-3)
- **Architecture** — Uses a **mixture of experts (MoE)** model to improve scalability
- **Training compute** — Trained on ~25,000 Nvidia A100 GPUs over 90-100 days
- **Training data** — Trained on a dataset of ~13 trillion tokens
- **Inference compute** — Runs on clusters of 128 A100 GPUs for efficient deployment
- **Context length** — Supports up to 32,000 tokens of context

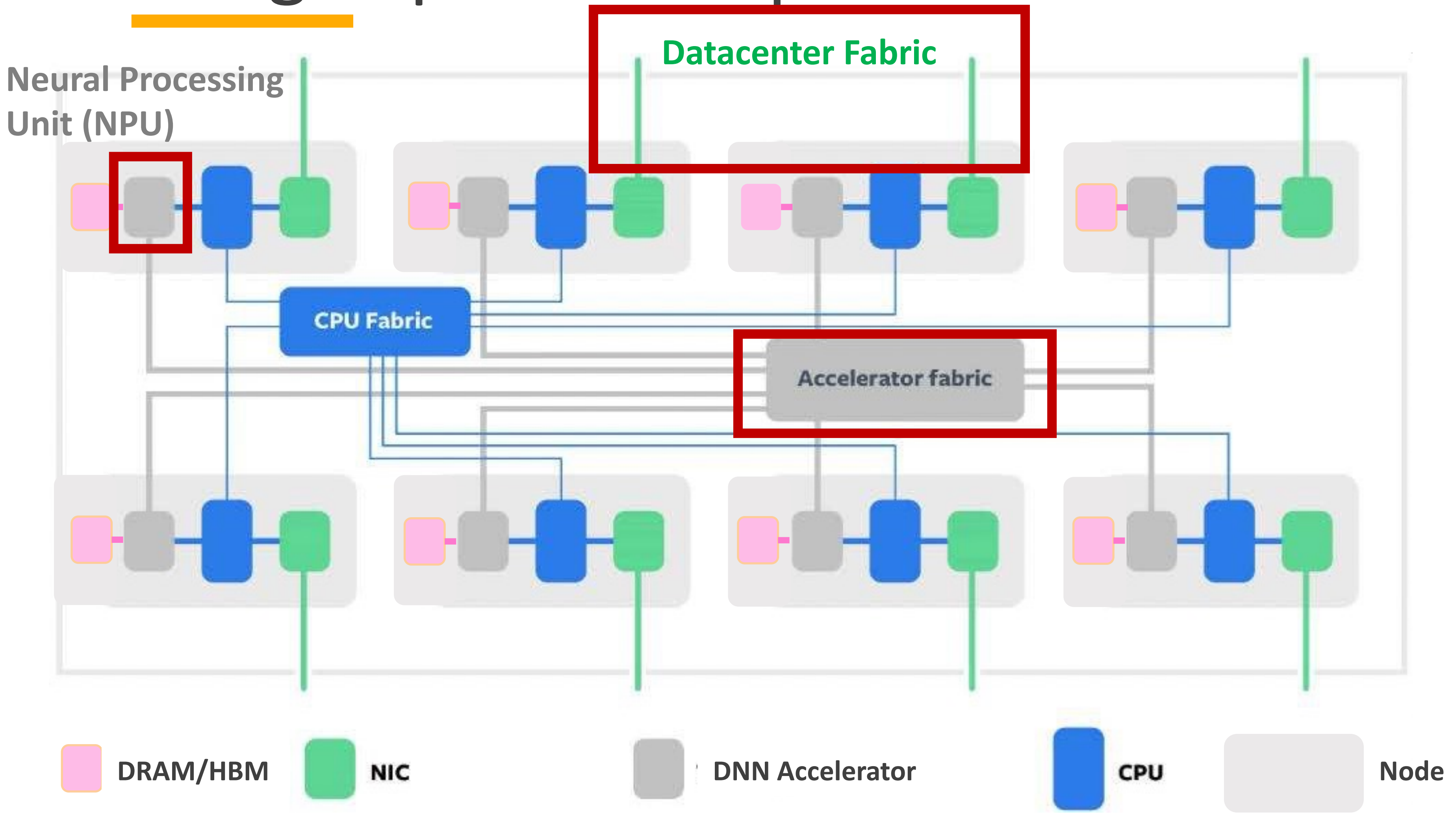
Outline

- Design Space of AI Platforms
- Chakra Workload Execution Traces
- ASTRA-sim Layers and APIs
- External Engagements
- Conclusion

Outline

- **Design Space of AI Platforms**
- Chakra Workload Execution Traces
- ASTRA-sim Layers and APIs
- External Engagements
- Conclusion

Design Space of AI platforms

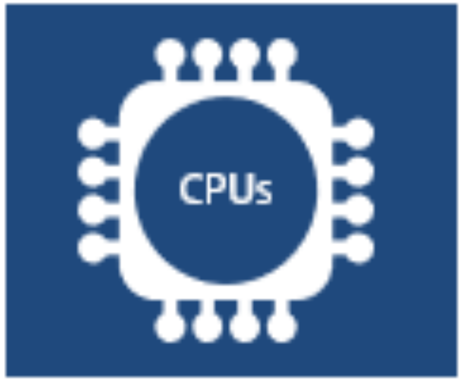


- ✓ Customized accelerators for AI (aka NPUs)
- ✓ Customized network fabrics to scale the AI task across multiple accelerators

*Figure modified from **"Zion: Facebook Next- Generation Large Memory Training Platform", Misha Smelyanskiy, Hot Chips 31"***

Diverse Compute

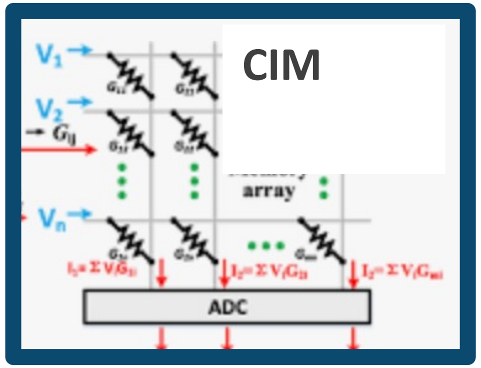
Intel
AMD



NVIDIA
AMD

Microsoft

EnChargeAI

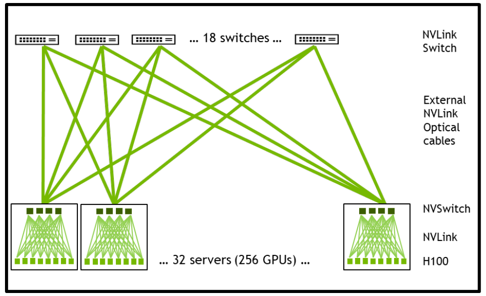


Google
Amazon
Meta
Microsoft
Groq
Cerebras
Rebellions

Diverse Networks

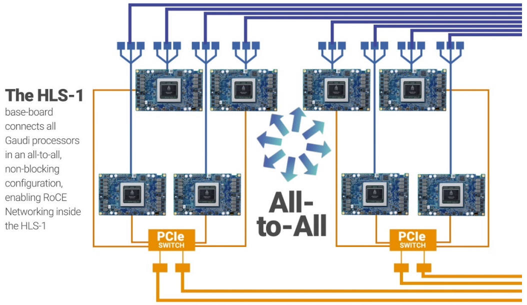
Scale-up → Scale-out

NVIDIA



NVswitch → Infiniband

Intel

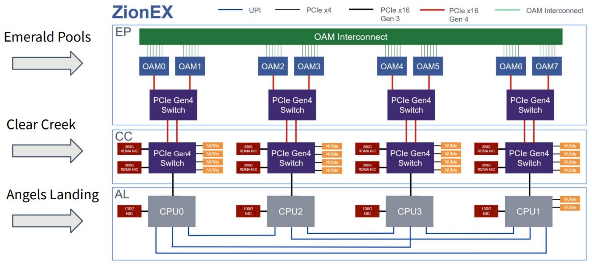


Custom NICs → RoCE

AMD

Infiniti → Infiniti

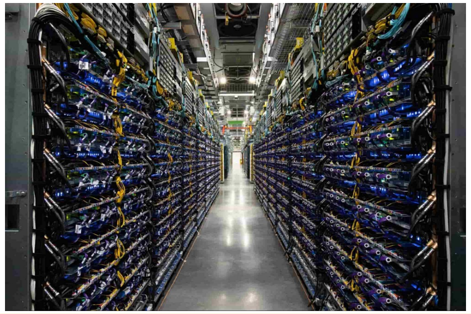
Meta



NVlink → RoCE

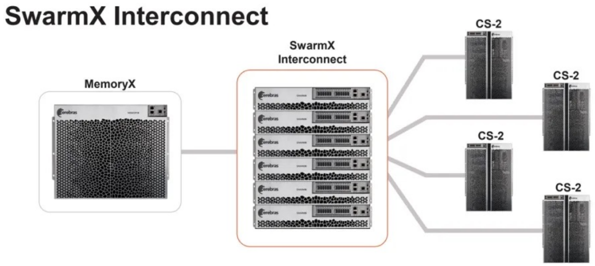
"Classic" Technologies

Google



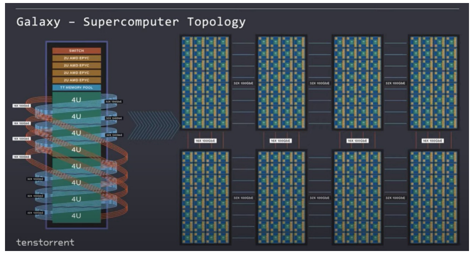
3D Electrical Torus → Optical

Cerebras



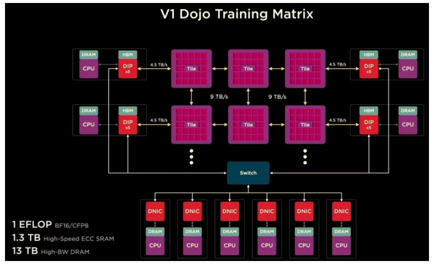
Wafer-scale → SwarmX Tree

Tensorrent



On-package Mesh → off-chip mesh

Tesla

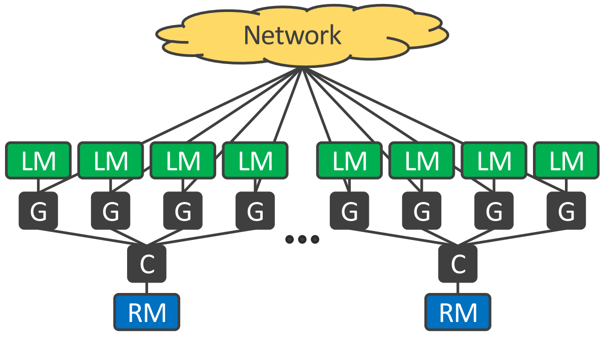


On-package Mesh → Ethernet

"Emerging" Technologies

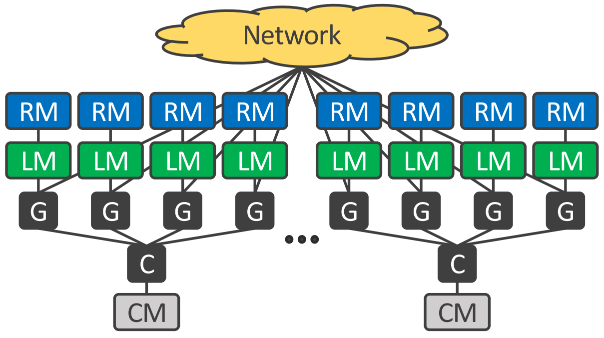
Diverse Memory Systems

C CPU | G GPU | OM On-chip spad memory | LM Local main memory | RM Remote memory | CM CPU main memory



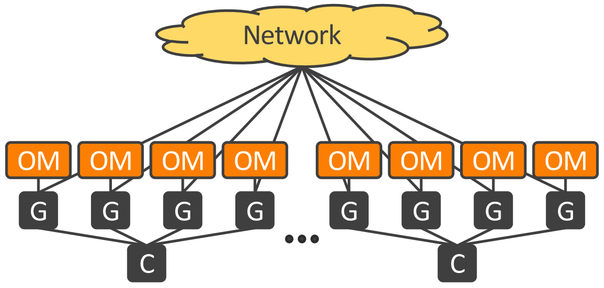
① Per-node memory expansion

Intel Habana, NVIDIA DGX2



② Per-GPU memory expansion

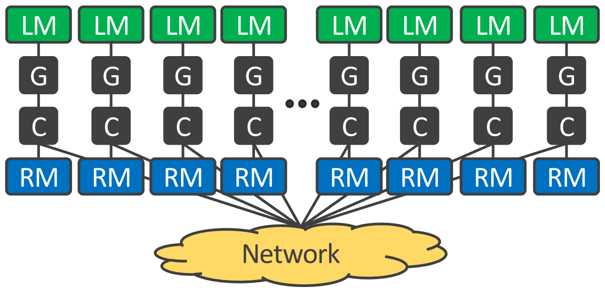
Astera Labs



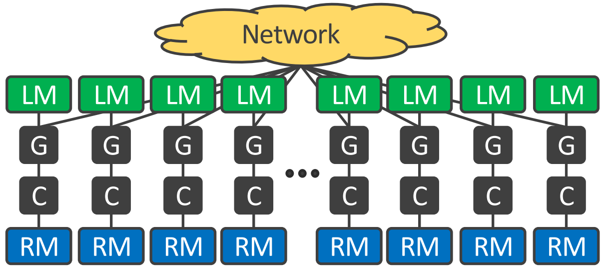
(5) On-chip (SRAM) memory

Groq

Cerebras CS-3



③ Integ. chip mem. exp with CPU network

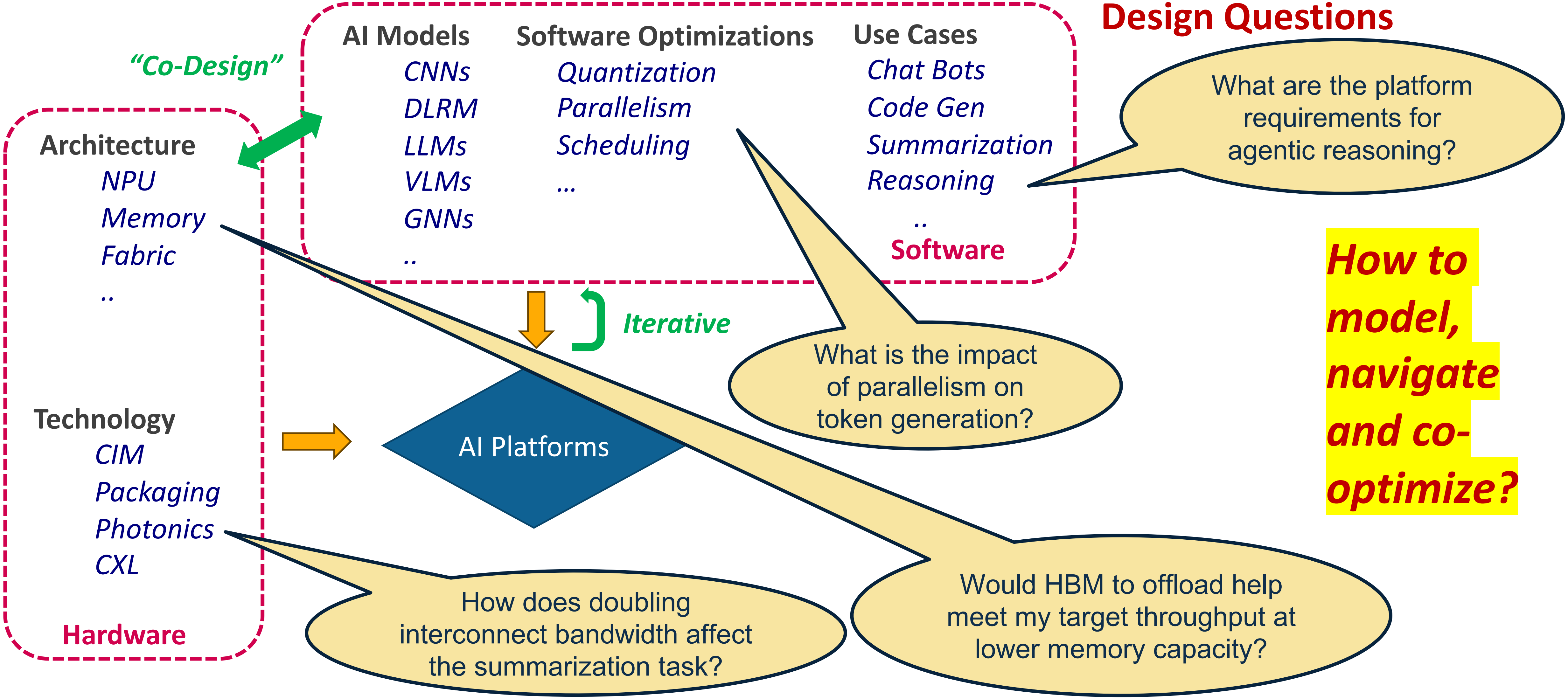


④ Integ. chip mem. exp with GPU network

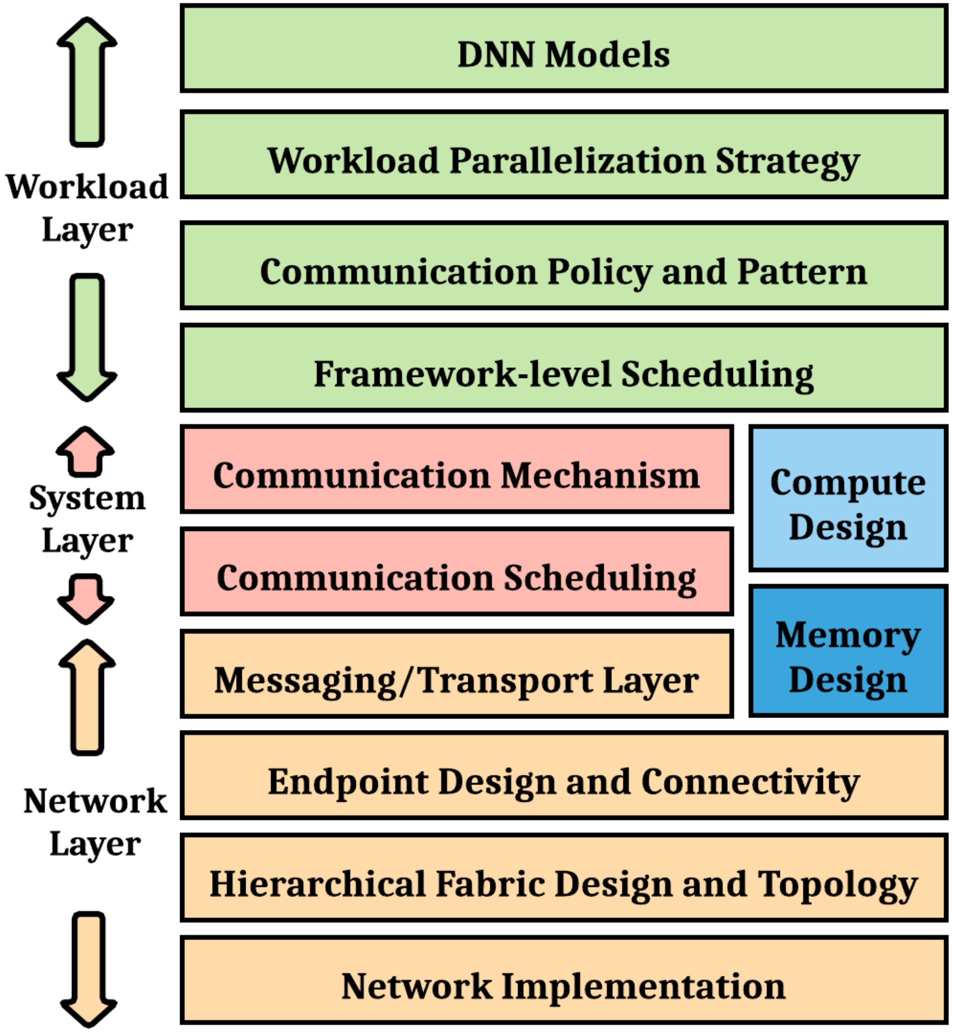
NVIDIA Grace Hopper

Cross-coupled Design-Space

Examples of Platform Design Questions



HW/SW Co-Design Space



- → DLRM, ResNet-50, Transformer, GNMT
- → Data, Model, Platform Agnostic Hybrid, Platform-aware Hybrid, Pipelined Parallelism
- → Distributed worker, Parameter Server
- → Sync/Async, Blocking/Non-blocking
- → Topology-aware Collectives, Send/Recv, RPC
- → Dataflow, Microarchitecture, Flexibility, Sparsity Support
- → LIFO, FIFO, Fusion
- → TCP, RDMA (+ GPUDirect RDMA)
- → # links, BW per link, architecture (chip/package/board), NIC offload, compression
- → Flat vs. Hierarchical, 2D/3D/4D Torus (TPU), Switch (DGX2), Fully-Connected, Hyper-Cube Mesh
- → Buffering, Flow-control, Arbitration, Congestion Mgmt

How to model, navigate and co-optimize?

Figure Courtesy: Srinivas Sridharan (NVIDIA)

Introducing Chakra and ASTRA-sim

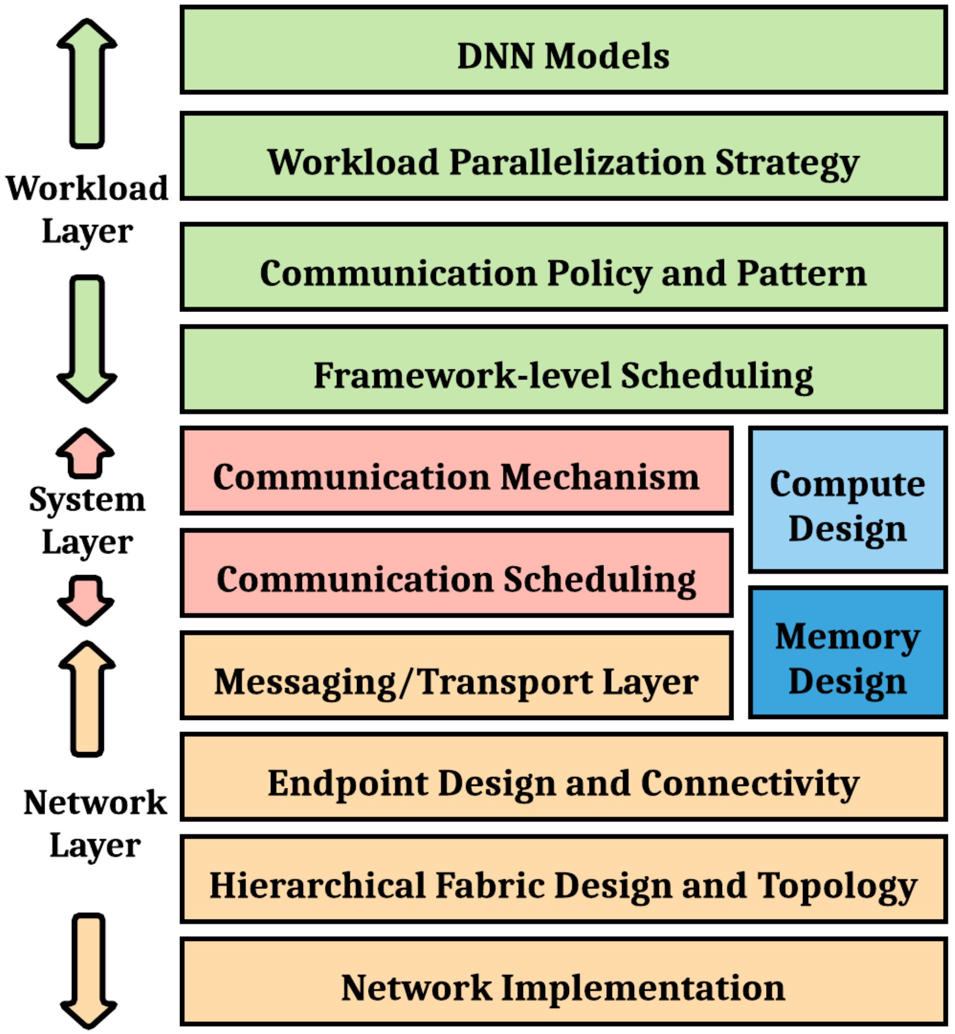


Figure Courtesy: Srinivas Sridharan (NVIDIA)

Chakra Execution Graph/Trace: an open graph-based representation of AI/ML workload execution

ASTRA-sim: Distributed AI system simulator

Outline

- Design Space of AI Platforms
- **Chakra Workload Execution Traces**
- ASTRA-sim Layers and APIs
- External Engagements
- Conclusion

HW/SW Co-Design Space

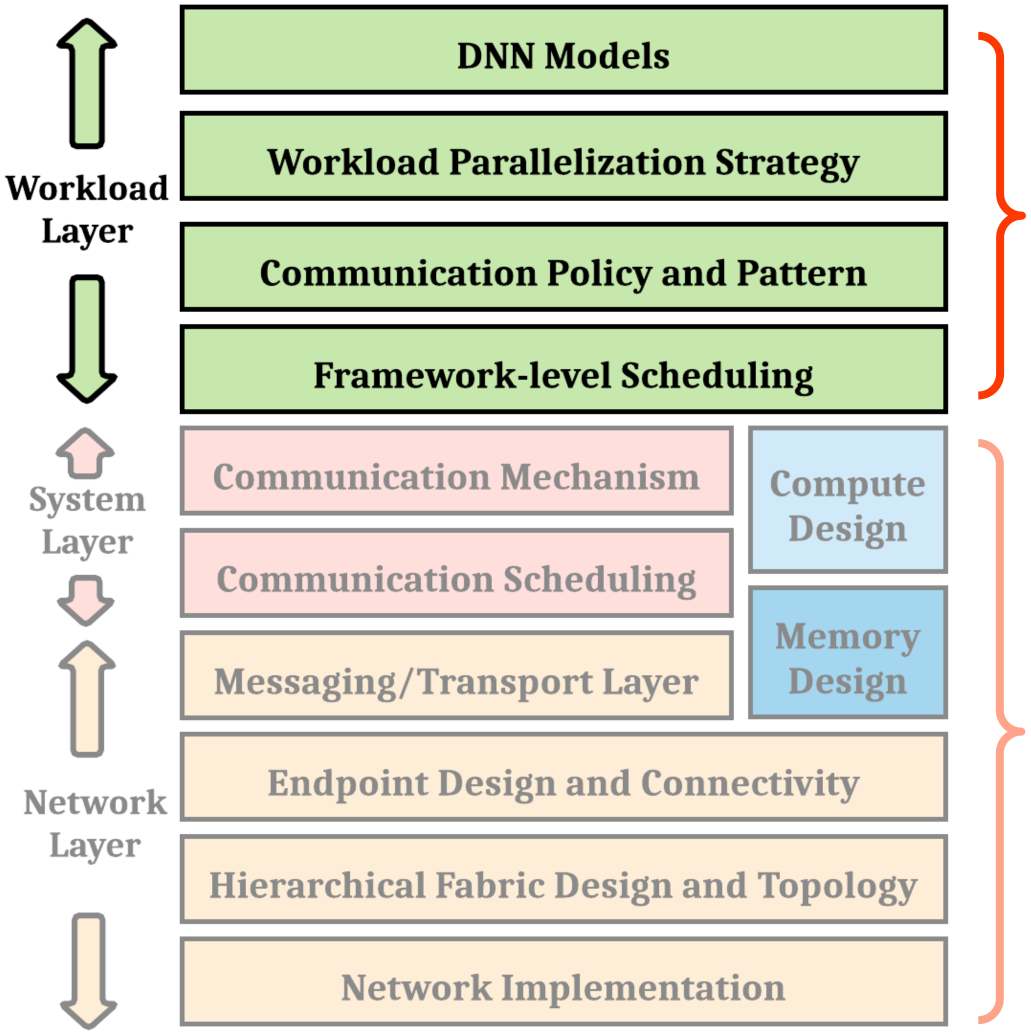
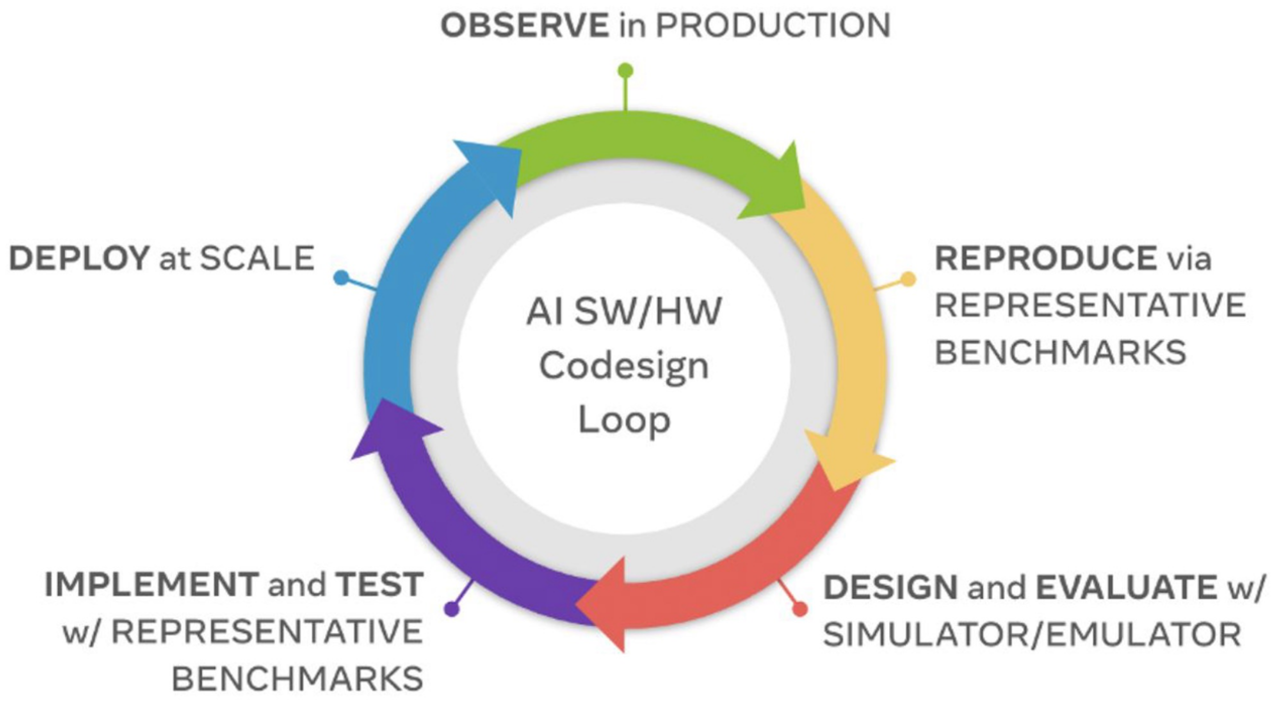


Figure Courtesy: Srinivas Sridharan (NVIDIA)

Chakra Execution Graph/Trace: an open graph-based representation of AI/ML workload execution

ASTRA-sim: Distributed AI system simulator

Chakra: Motivation

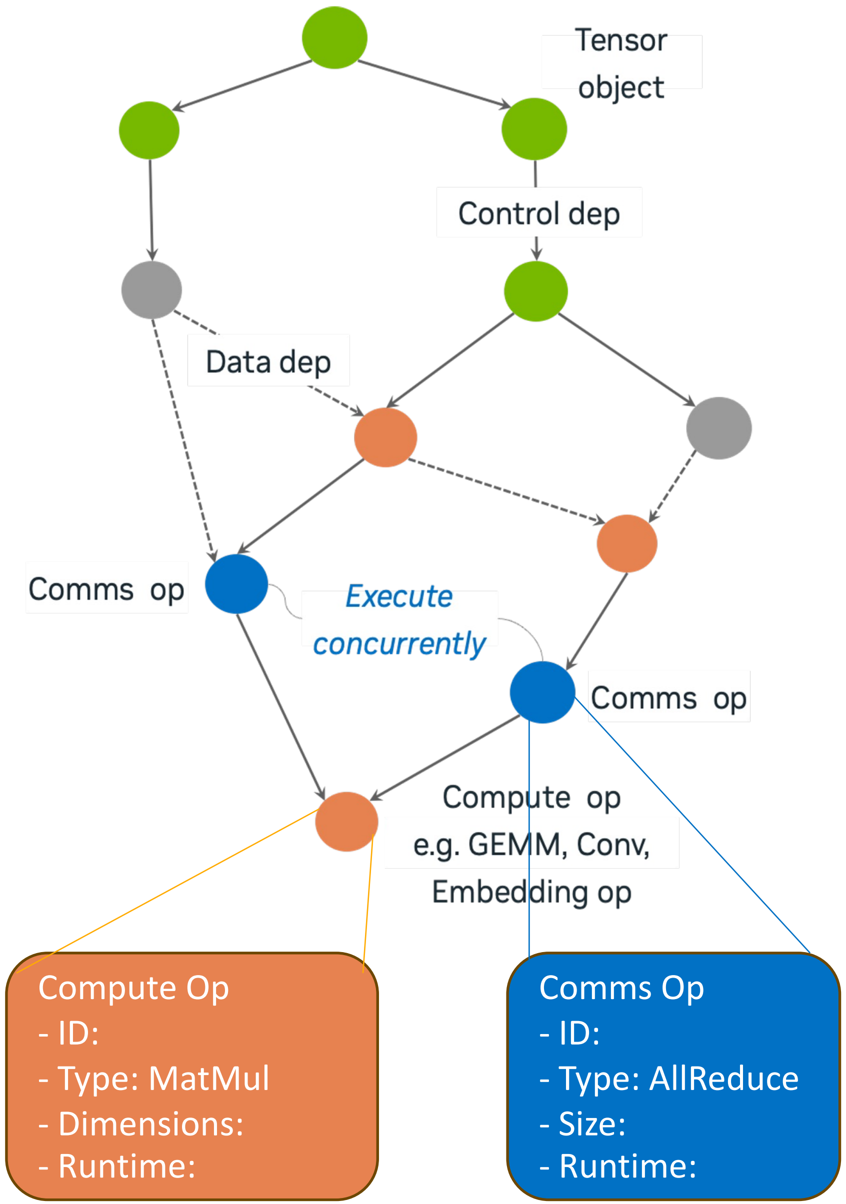


Motivation

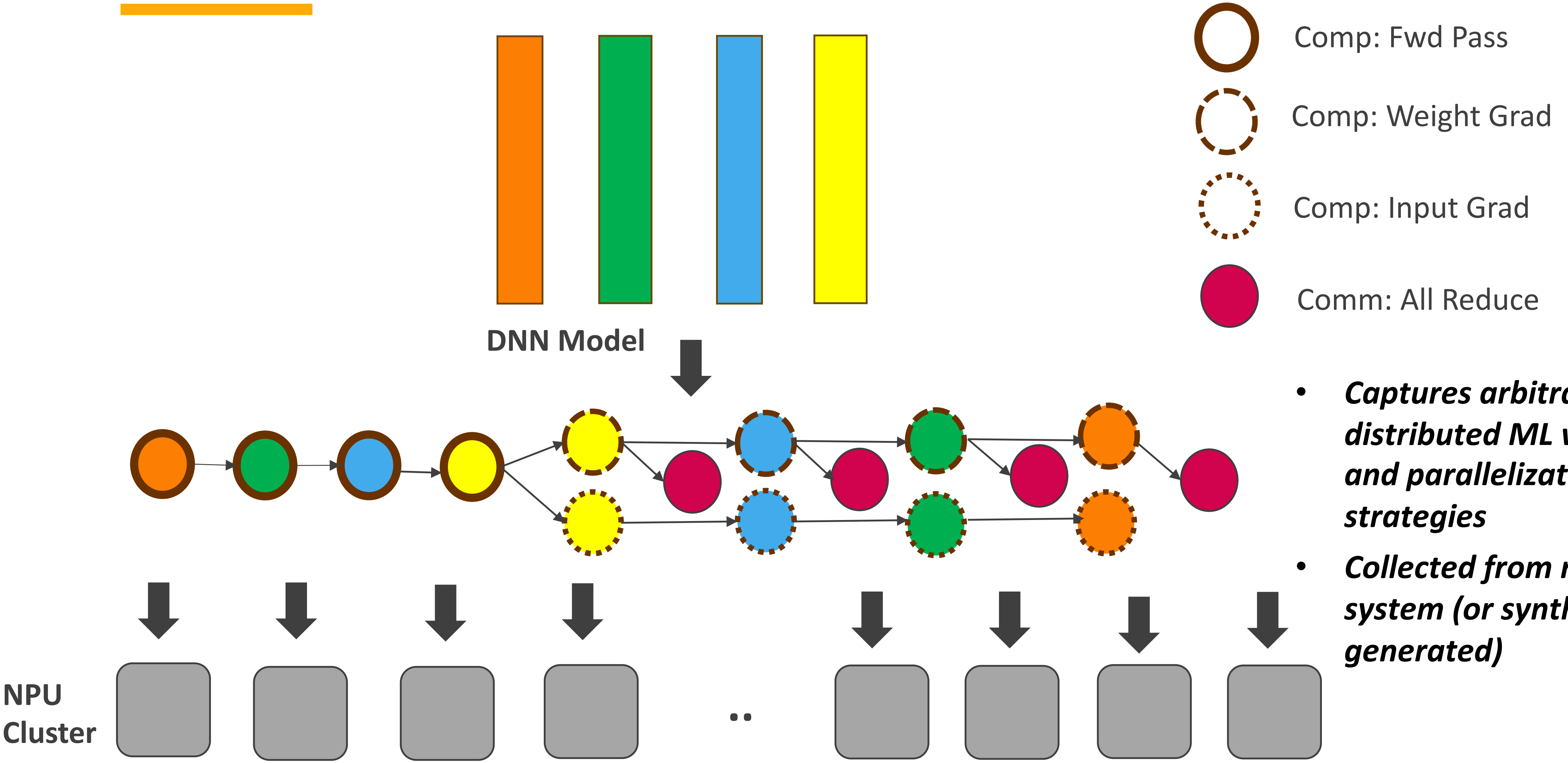
- High-cost of running full workload benchmarks
- Requires cross-domain full-stack expertise
- Difficult to isolate specific HW/SW bottlenecks
- Difficult to isolate compute, memory, network behavior
- Cannot keep up with the pace of AI innovation
- Hard to obfuscate proprietary AI model details
- Hard to reproduce without support infrastructure

Chakra Execution Graphs

- **Hierarchical DAG**
  - **Nodes**
    - Primitive operators: compute, comms, memory
    - Tensor objects: shape, size, device (local/remote)
    - Timing and resource constraints
  - **Edges**
    - Data dependency
    - Control dependency (e.g. call stack)
  - **Higher-level abstractions (e.g., components)**
    - Comprises of other components or primitive ops
- **Benefits**
  - Isolate comms and compute operators
  - Operator, dependencies, and timing for replay, simulation, and analysis
  - Flexible to represent both workloads and collective implementations
  - Graph transformations to obscure sensitive IP

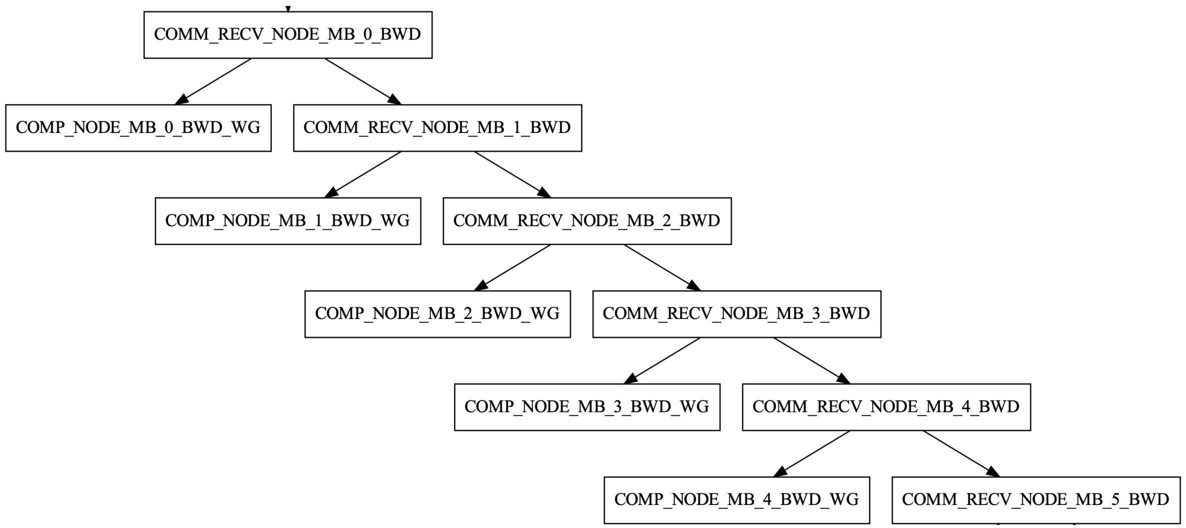


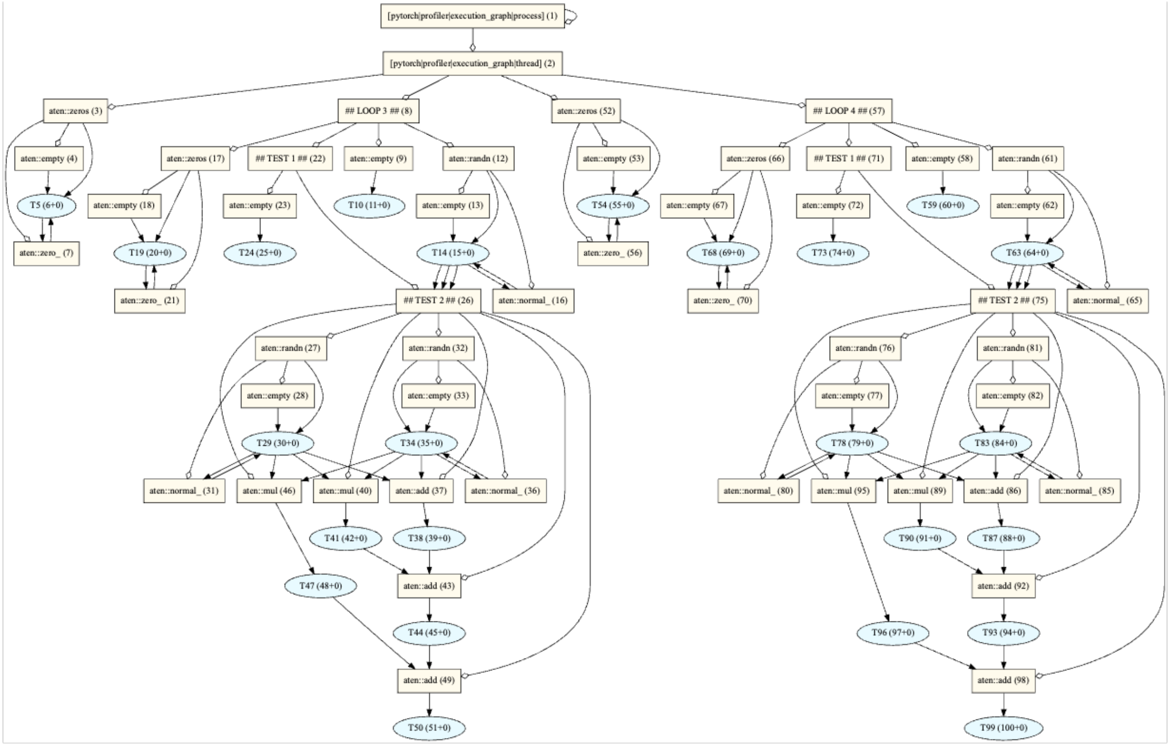
Chakra Execution Graphs



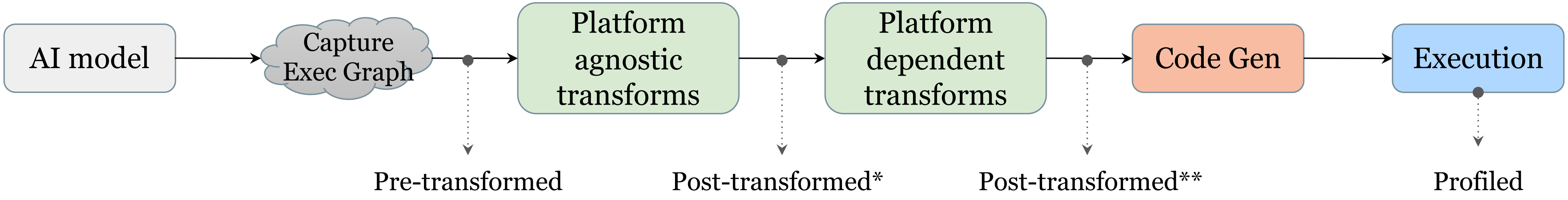
- ***Captures arbitrary distributed ML workload and parallelization strategies***
- ***Collected from real-system (or synthetically generated)***

Sample Chakra Traces from Real systems



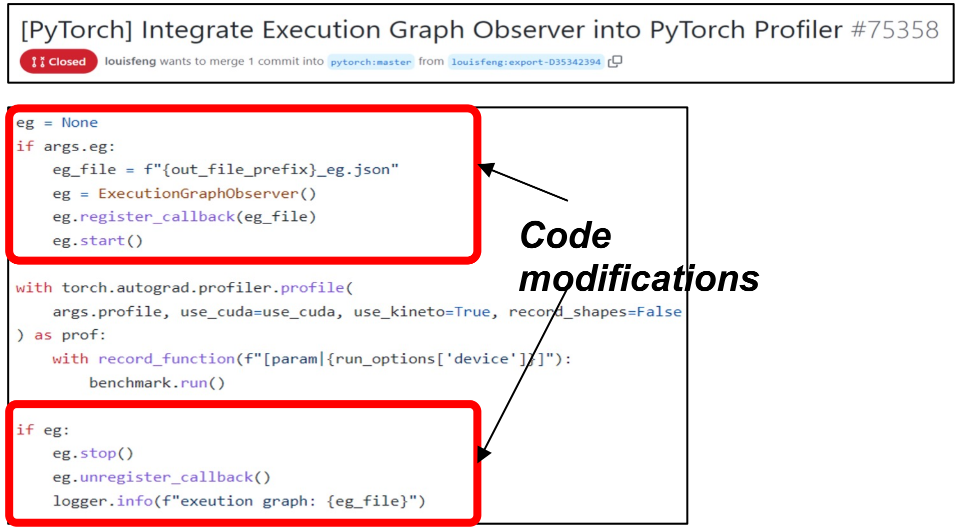


Chakra Traces: Collection and Synthesis



Type of Execution Traces

1. **Pre-transformed:** original model
2. **Post-transformed#:** optimized graph (may or may not be platform dependent) *through PyTorch2.0 FXgraphs*
3. **Profiled Traces:** graph executed on a specific platform
4. **Synthesized:** via analytical or statistical models

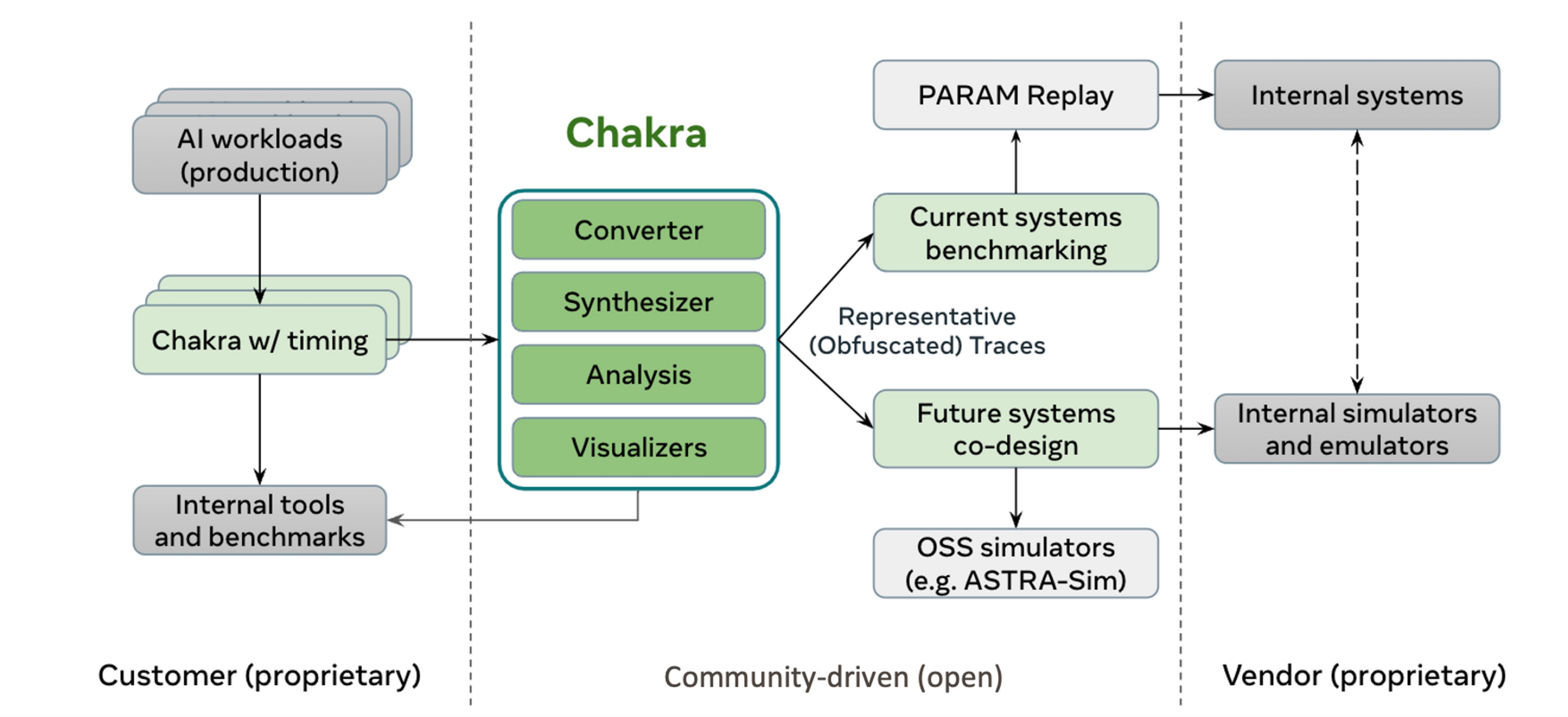



```
eg = None
if args.eg:
    eg_file = f"{out_file_prefix}_eg.json"
    eg = ExecutionGraphObserver()
    eg.register_callback(eg_file)
    eg.start()

with torch.autograd.profiler.profile(
    args.profile, use_cuda=use_cuda, use_kineto=True, record_shapes=False
) as prof:
    with record_function(f"[param|{run_options['device']}]"):
        benchmark.run()

if eg:
    eg.stop()
    eg.unregister_callback()
    logger.info(f"exeution graph: {eg_file}")
```

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

ICLR 2025 paper (GT, Meta, NV): https://arxiv.org/abs/2411.02322

Chakra Ecosystem and End-to-End Flow



Outline

- Design Space of AI Platforms
- Chakra Workload Execution Traces
- **ASTRA-sim Layers and APIs**
- External Engagements
- Conclusion

HW/SW Co-Design Space

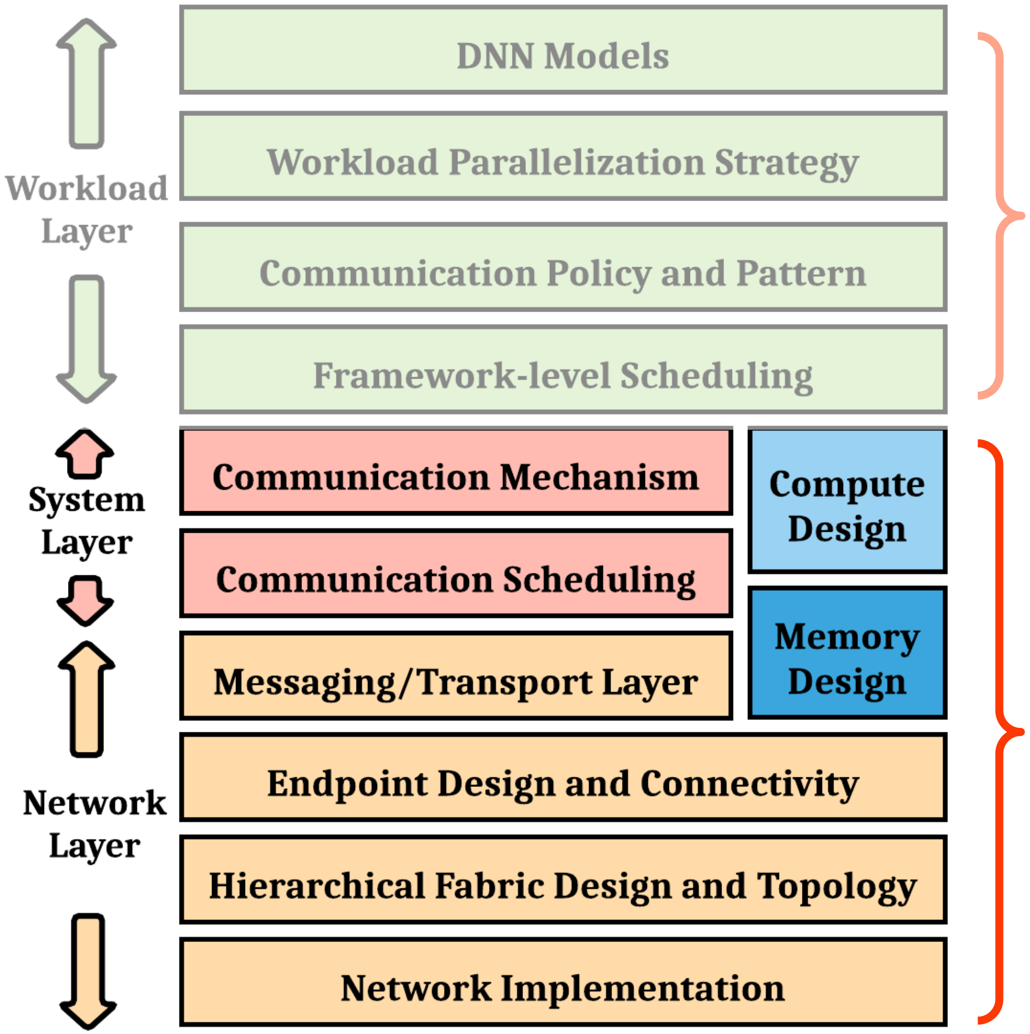
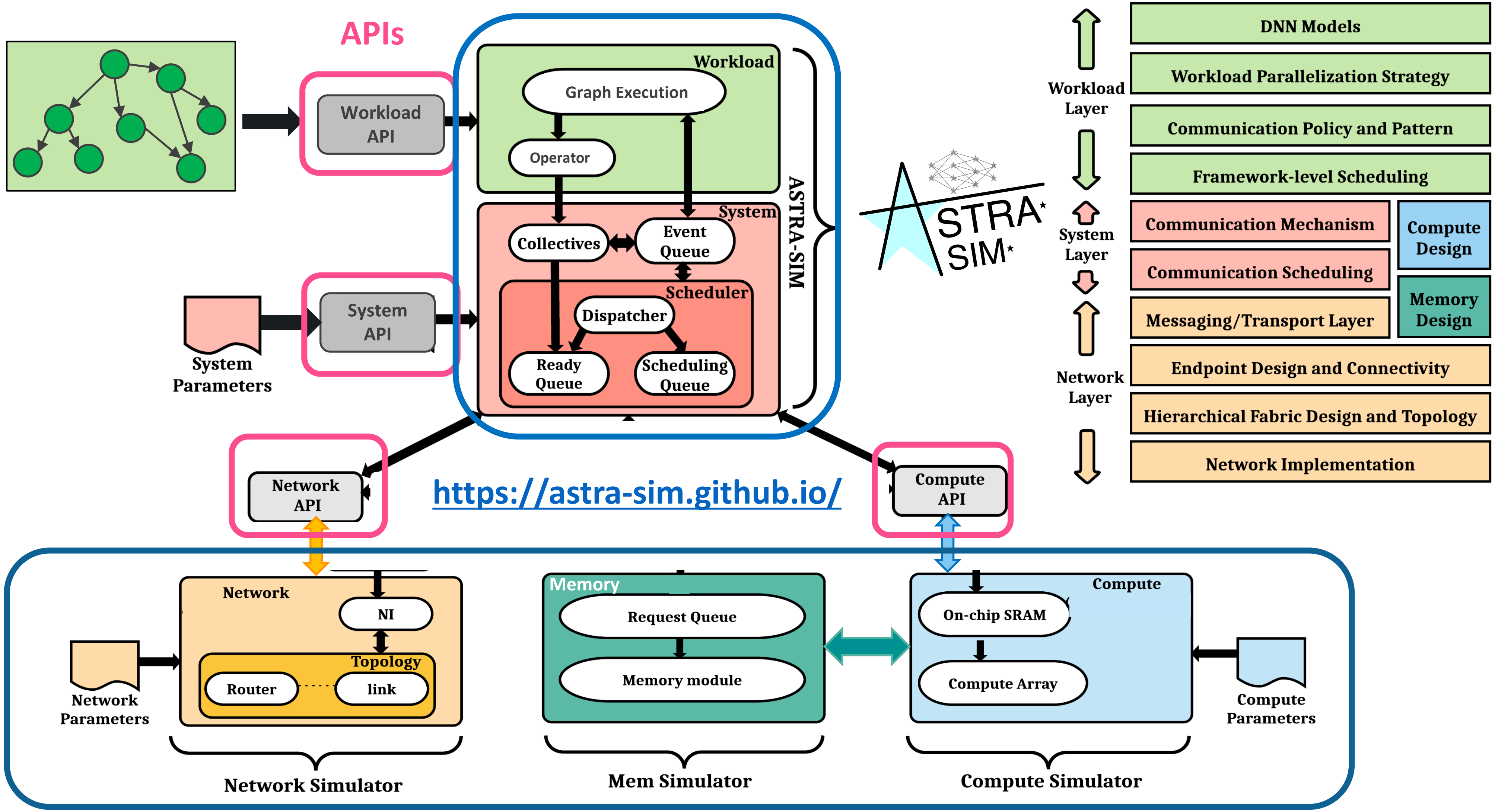


Figure Courtesy: Srinivas Sridharan (NVIDIA)

Chakra Execution Graph/Trace: an open graph-based representation of AI/ML workload

ASTRA-sim: Distributed AI system simulator

Overview of ASTRA-sim



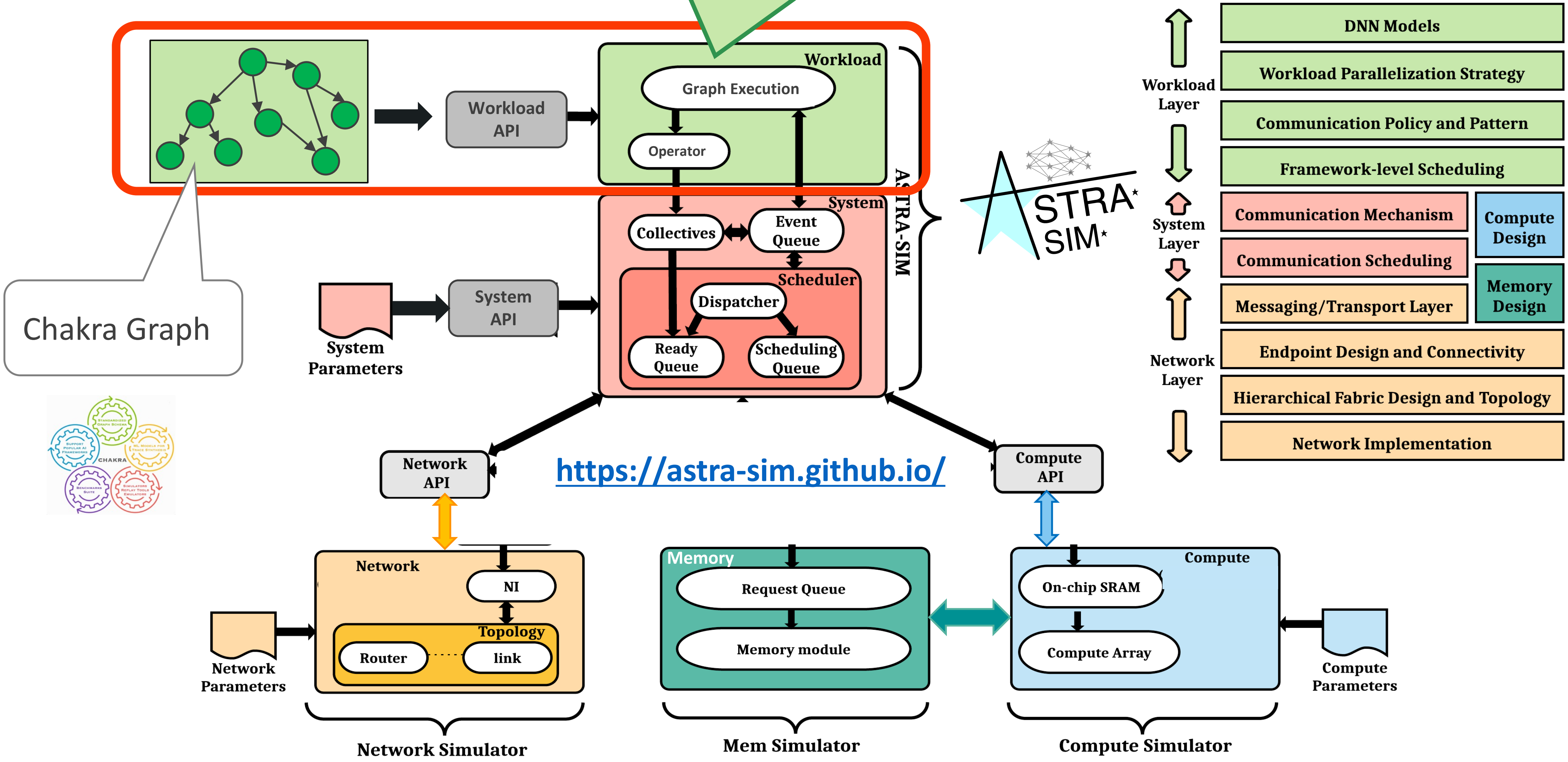
Custom Simulators (varying degrees of fidelity)

ASTRA-sim: Design Principles

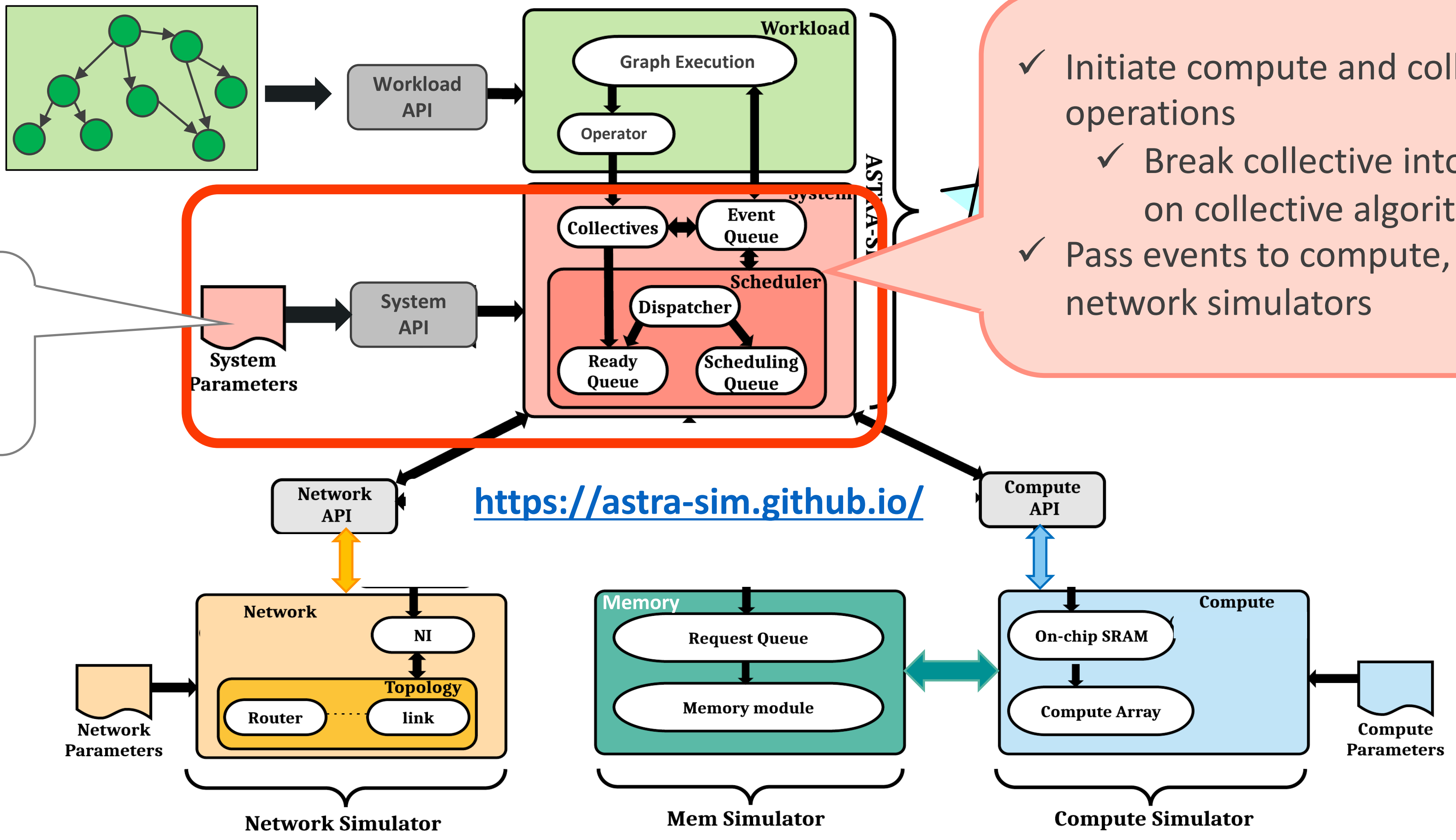
- User determines the model/simulator for compute/network/memory depending on the level of detail and simulation time they want
- **Key enabler:** APIs for plugging in diverse external open/proprietary tools (i.e., composable simulators)
- **Reference Implementation:** http://github.com/astra-sim/astra-sim
- **Website:** https://astra-sim.github.io/
- **Tutorials:** https://astra-sim.github.io/tutorials

Workload Layer

- ✓ Graph parsing and execution
- ✓ Pass operators to system layer



System Layer

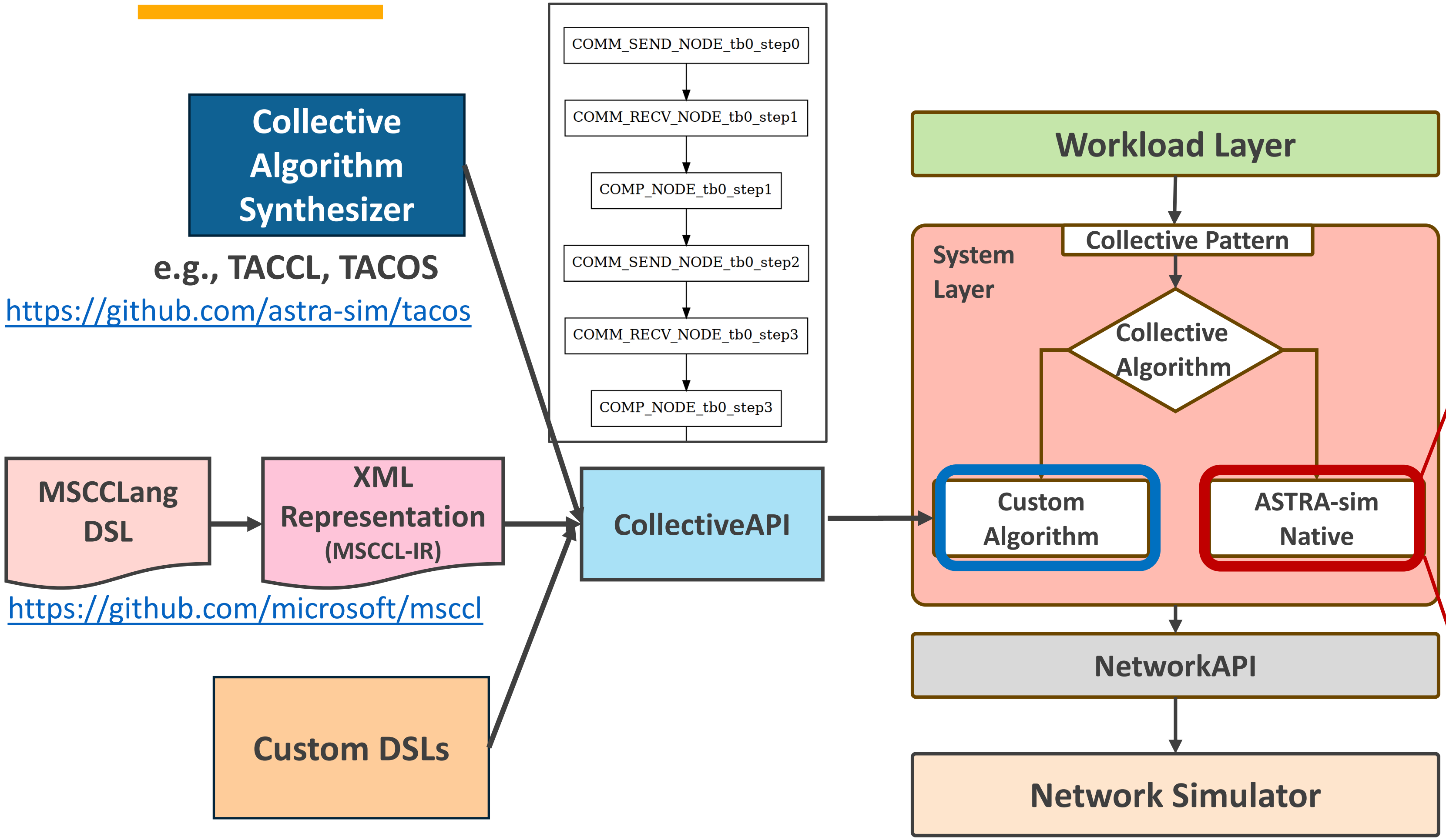


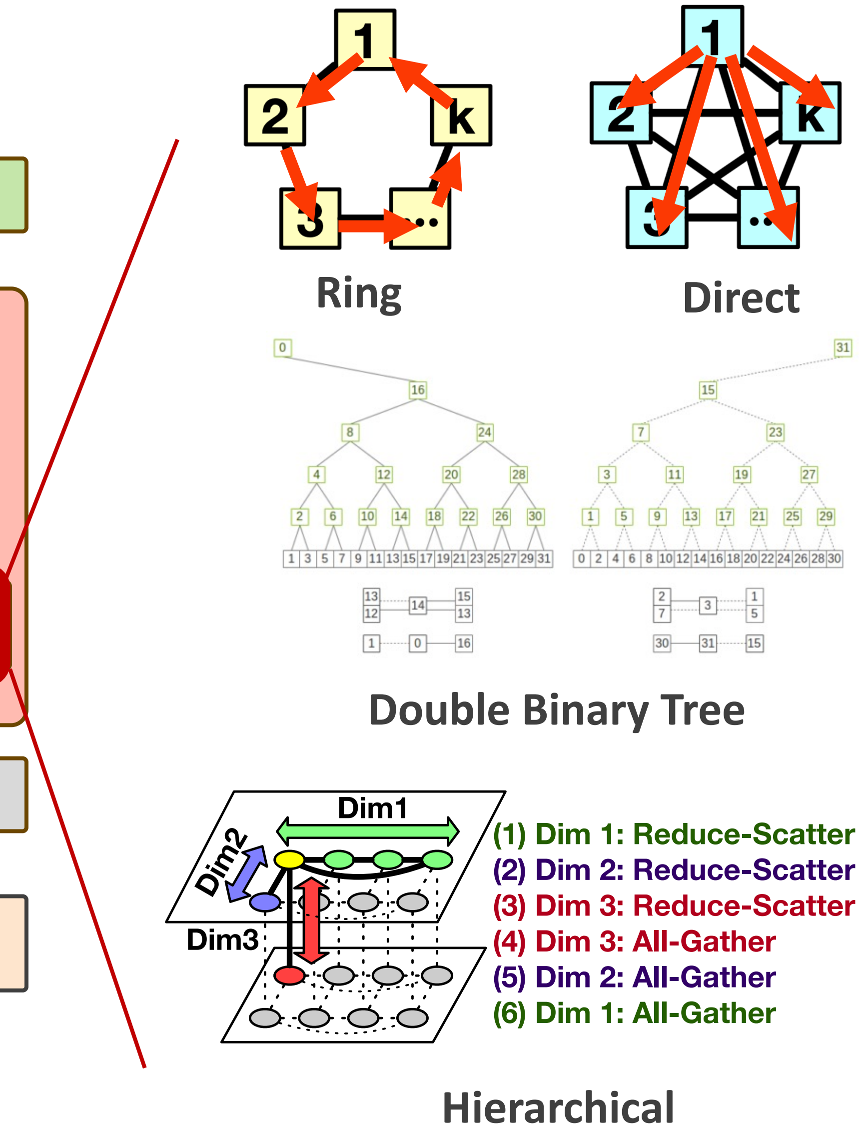
Collective Communication Algorithm & Scheduling

- ✓ Initiate compute and collective operations
  - ✓ Break collective into send-rcv based on collective algorithm
- ✓ Pass events to compute, memory and network simulators

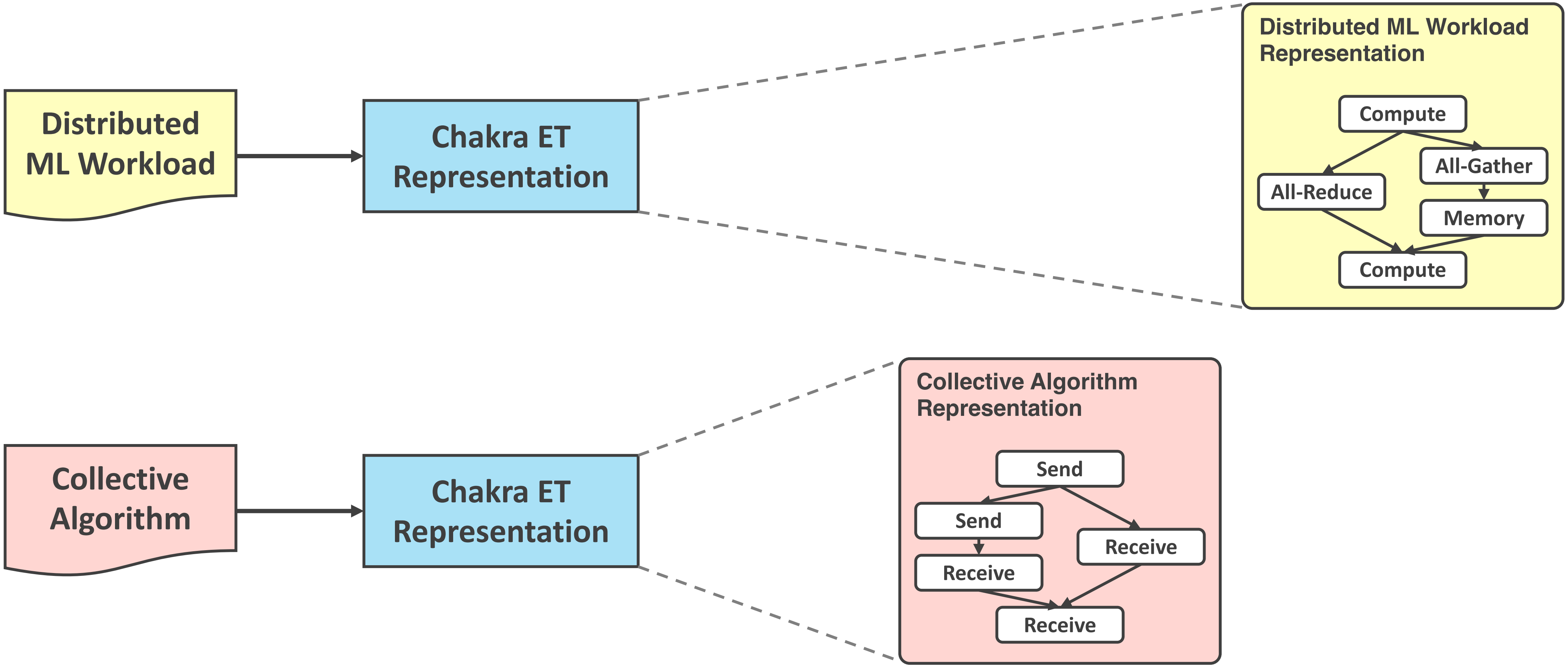
https://astra-sim.github.io/

Collective Algorithms



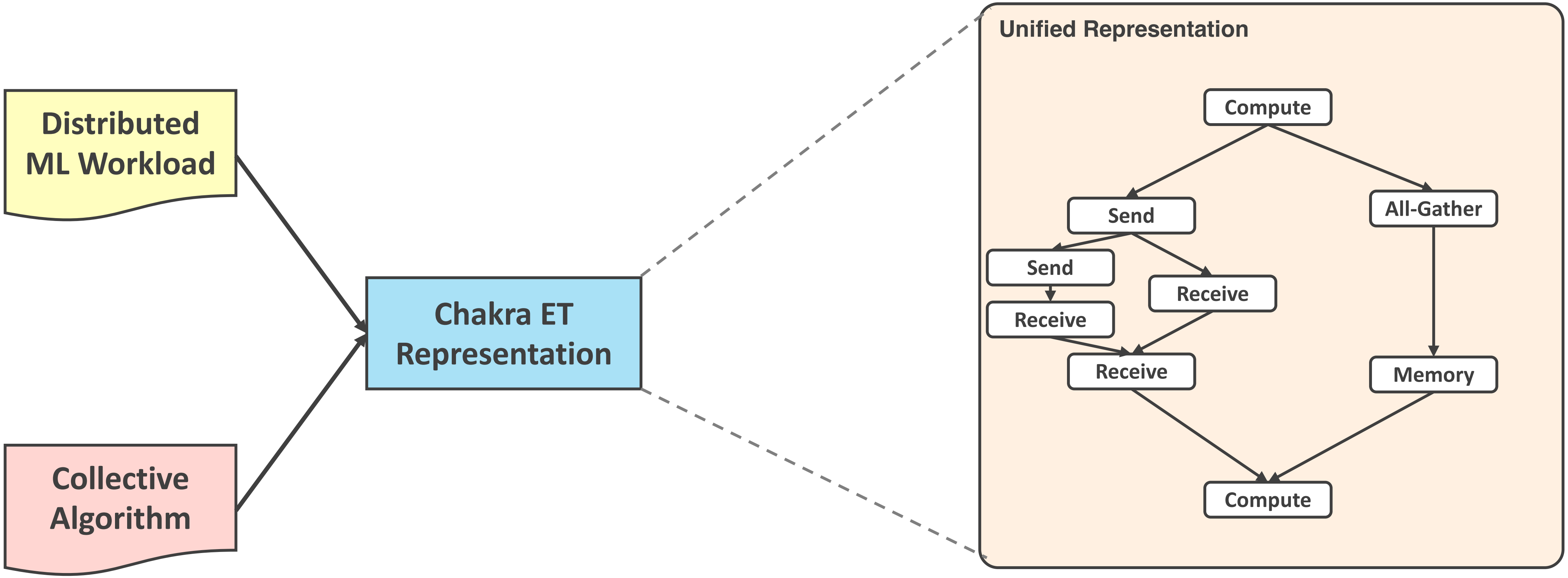


Collective API (uses Chakra format)



Jinsun Yoo, William Won, Meghan Cowan, Nan Jiang, Benjamin Klenk, Srinivas Sridharan, and Tushar Krishna, **"Towards a Standardized Representation for Deep Learning Collective Algorithms"**, *In Proc. of the 31st IEEE Hot Interconnects Symposium (**HotI**), Aug 2024*
https://arxiv.org/abs/2408.11008

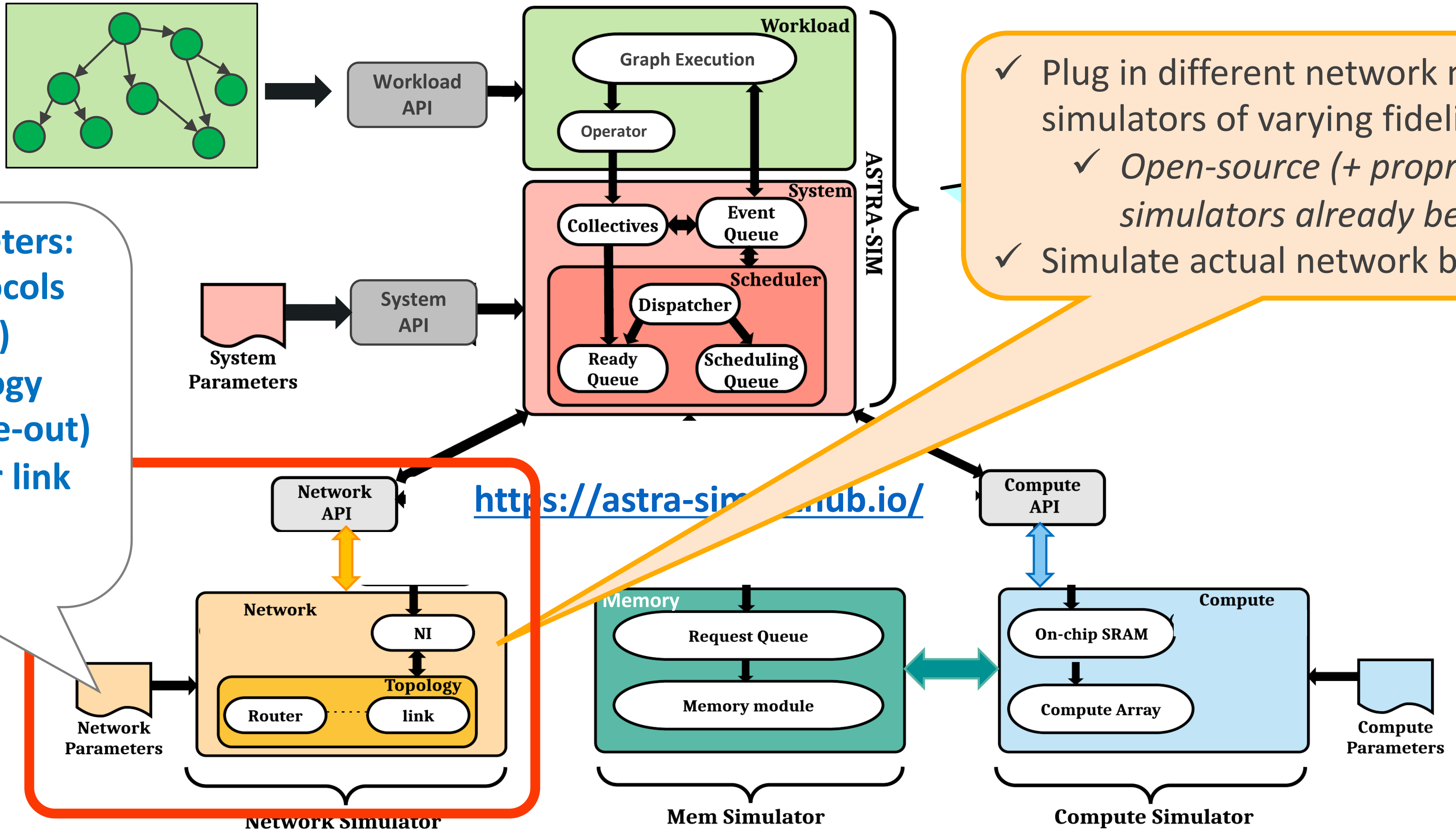
Vision: Compute-Collective Co-Optimization



Common representation using **Chakra ET** format
→ enable fine-grained co-optimization

Network Layer

Network parameters:
- Transport protocols (TCP, RDMA, etc.)
- Network topology (scale-up → scale-out)
- BW/latency per link
- Buffering and Arbitration



- ✓ Plug in different network models / simulators of varying fidelity
  - ✓ *Open-source (+ proprietary) simulators already being used*
- ✓ Simulate actual network behavior

Network Layer

- Simulates **actual network behavior** (send/recv)
- Supports **multiple** network models/simulators through NetworkAPI
  - Enabling the simulation of various scales/fidelity
  - We currently support **4 network simulators** implementing NetworkAPI

Model/Simulator	Purpose	Notable Feature
analytical	analytical equation-based simulation	fast simulation, hierarchical topologies
congestion-aware	congestion-aware analytical simulation	+ congestion (queueing) modeling
gem5-Garnet	on-chip/scale-up network simulation	packetization, flow control, congestion
HTsim	inter-cluster simulation	TCP-IP, congestion-control, UEC use cases
ns-3	inter-cluster simulation	RDMA, congestion-control
Real Network*	Transmit packets through real network	Measured network performance

Fidelity Speed

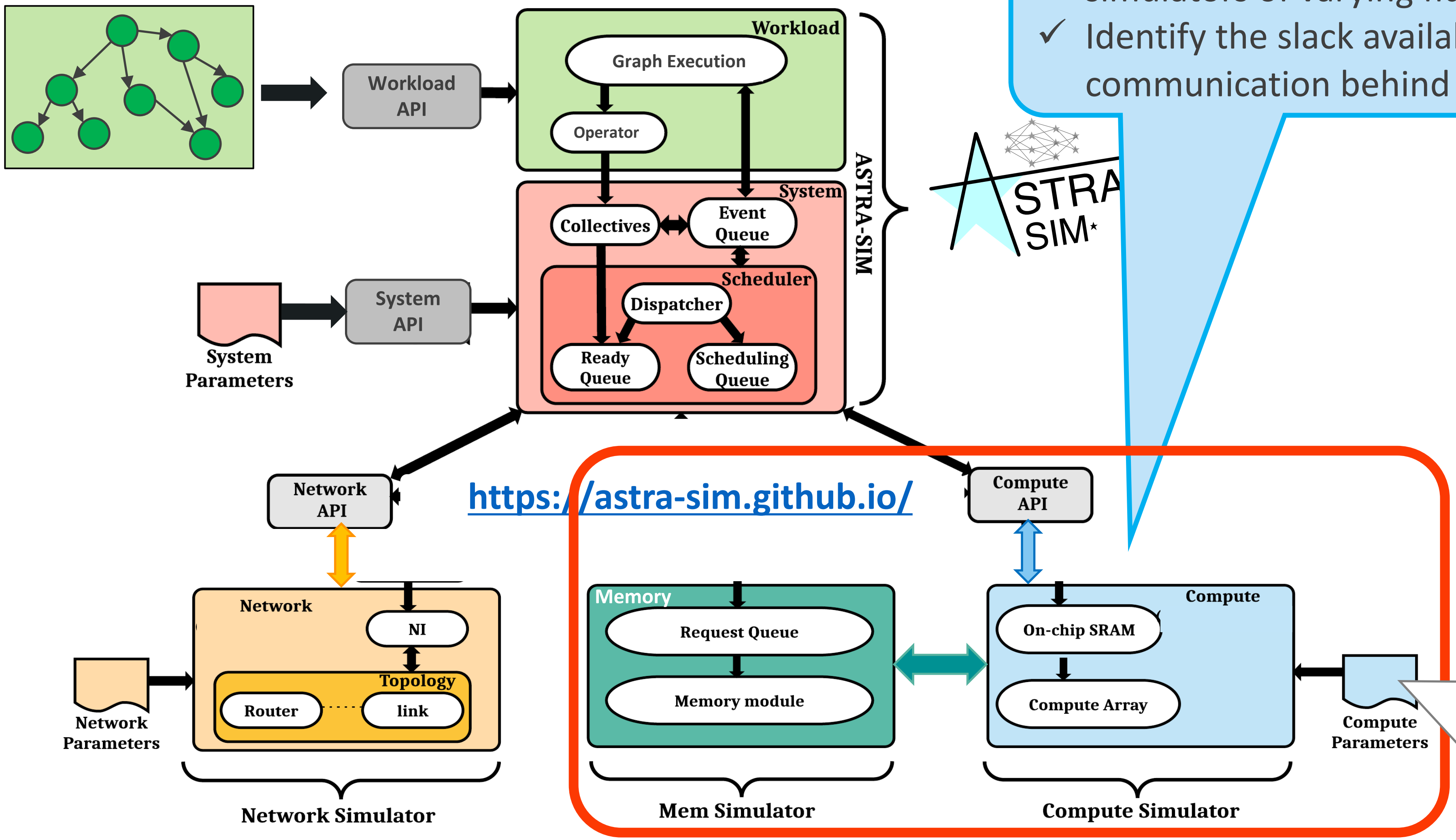
Depracated

*In progress

Additional proprietary network simulators implementing the NetworkAPI being used by AMD, Alibaba, HPE Labs, Keysight, some startups ..

Compute Layer

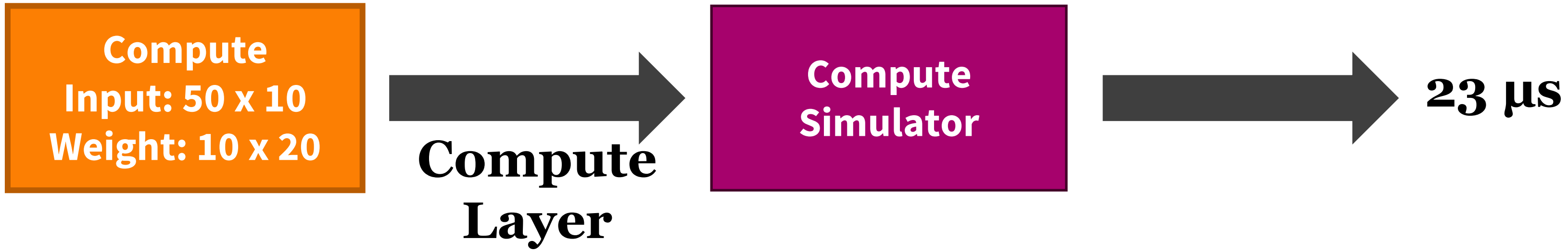
- ✓ Plug in different compute models / simulators of varying fidelity
- ✓ Identify the slack available to hide communication behind compute



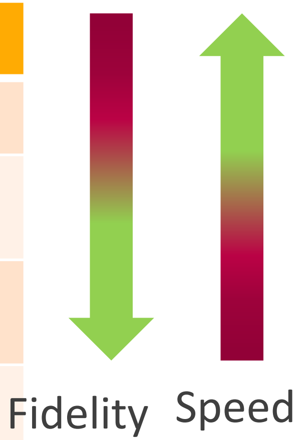
NPU parameters (FLOPS, dataflow, SRAM, Mem BW..)

Compute Layer

- **Simulates compute times required** for Chakra ET's compute node



Model	Purpose	Notable Feature
Roofline	Analytical: First-order roofline	Fast analysis for compute vs memory boundness
SCALE-sim	Cycle-accurate: systolic array and memory	Models Google-TPUv5-like SoC
Accel-sim*	GPU simulator	Can run CUDA code
Real GPU*	Run compute operator on real GPU	Measured runtime

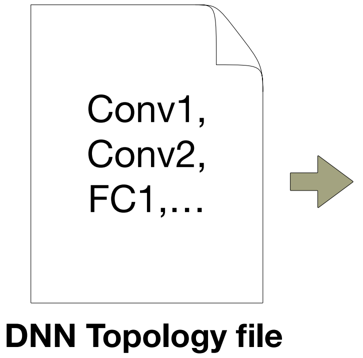


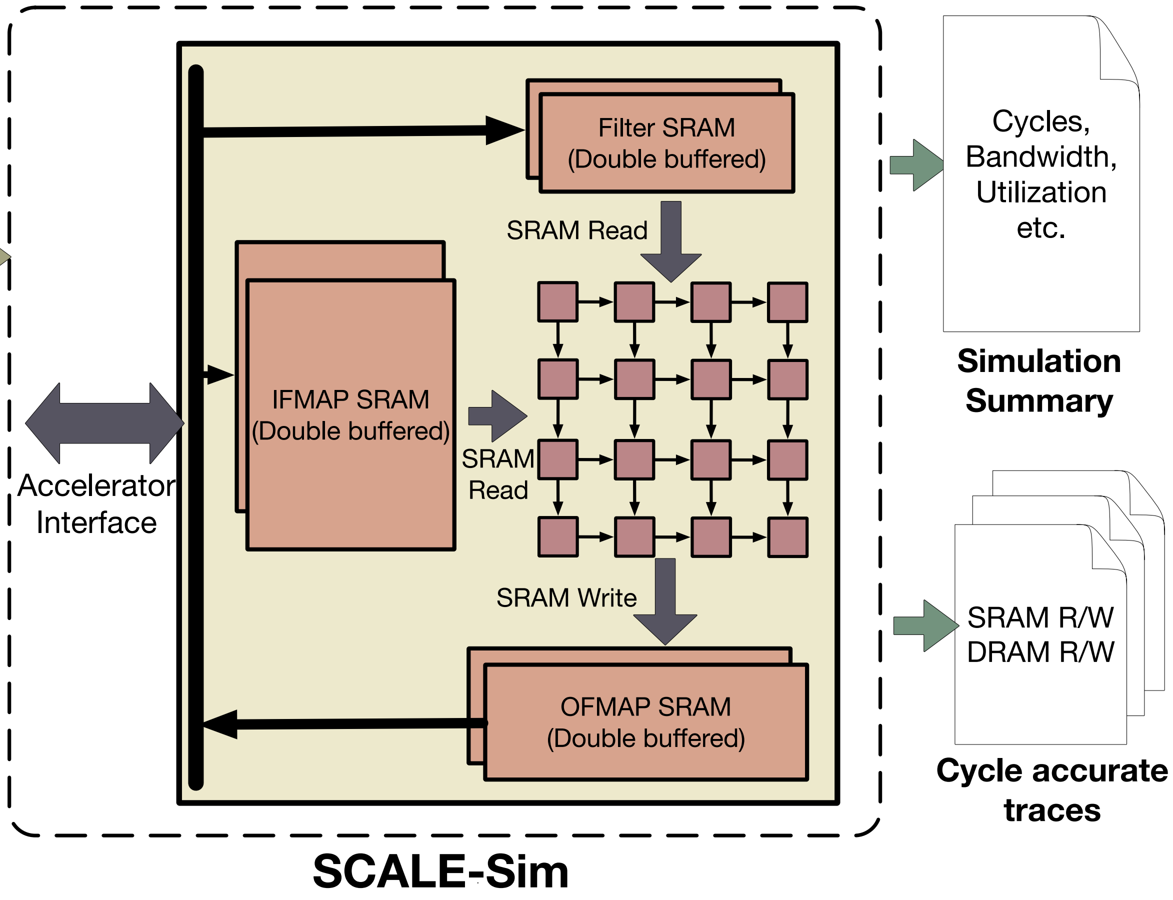
**In progress*

Sneak Peak: SCALE-Sim NPU Simulator

Parameter	Value
Array Height	32
Array Width	32
IFMAP SRAM	1024
Filter SRAM	1024
OFRAM SRAM	128
Dataflow	WS

Config file





https://github.com/scalesim-project/scale-sim

R Raj, S Banerjee, N Chandra, Z Wan, J Tong, A Samajdar, T Krishna, **"SCALE-Sim v3: A modular cycle-accurate systolic accelerator simulator for end-to-end system analysis"**, ISPASS 2025

Cycle accurate systolic array based accelerator simulator

Inputs

1. Architecture configuration
- Array dimensions
- Buffer sizes
- Dataflow (OS, WS, IS)

2. Workload hyper parameters

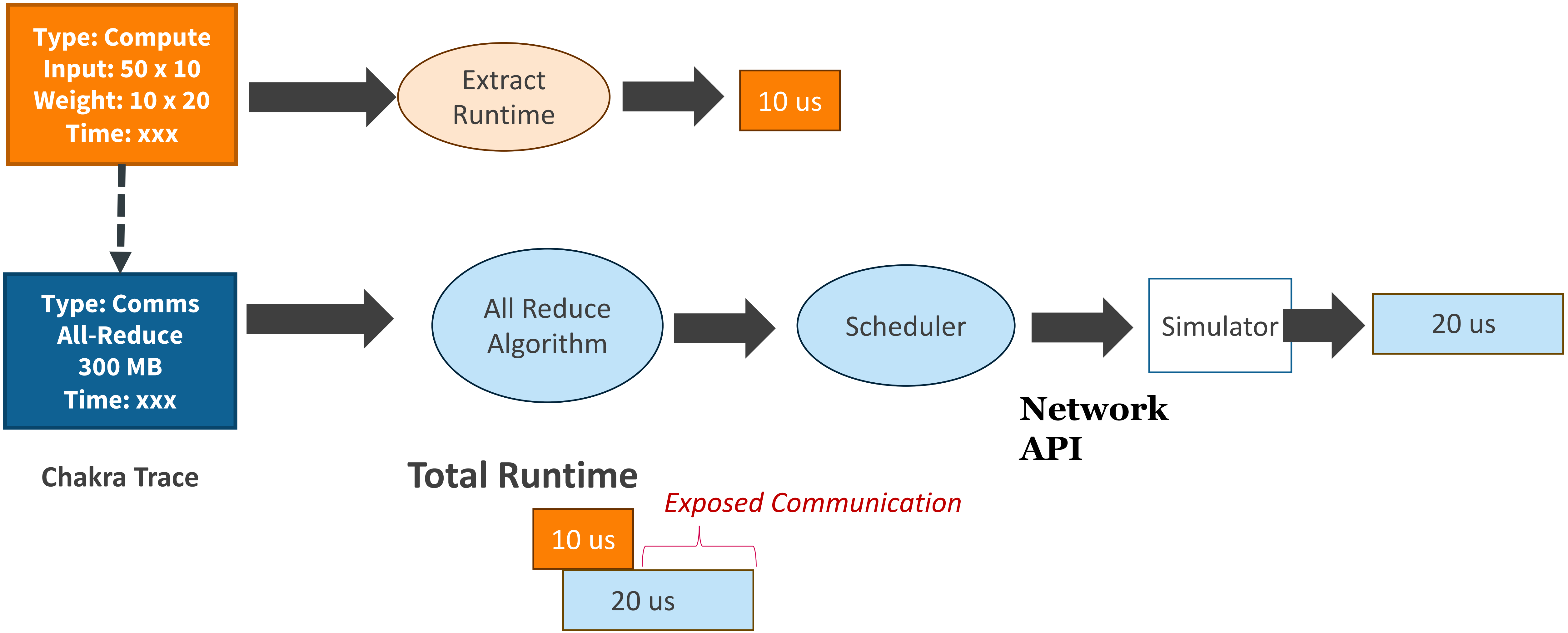
SCALE sim generates,

1. Runtime in cycles
2. Cycle accurate memory traces
3. Interface bandwidth requirements

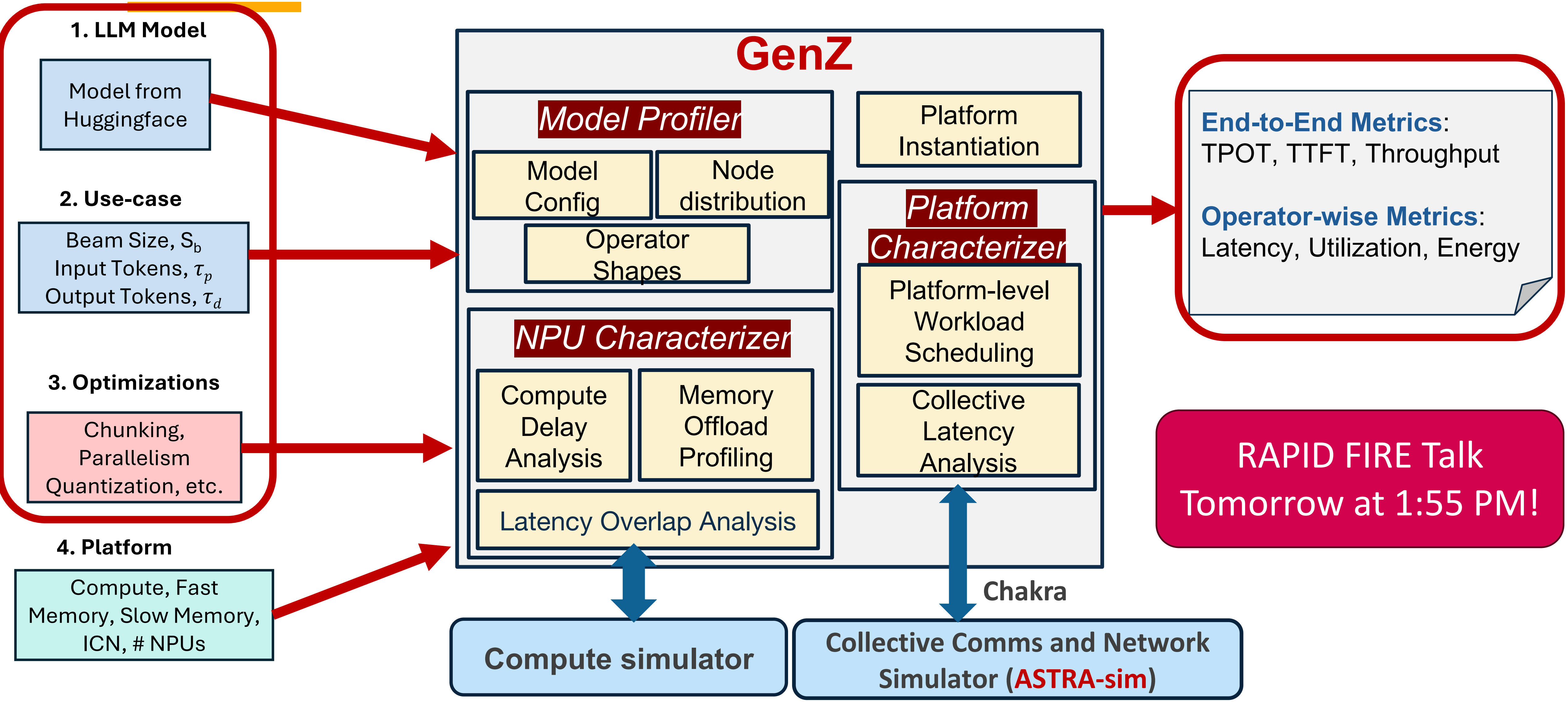
RAPID-FIRE Talk Tomorrow at 3:25 PM!

Example of ASTRA-sim in action

Simulating a system with new network fabric but same GPU



Sneak Peak: Performance Analysis of LLM Inference

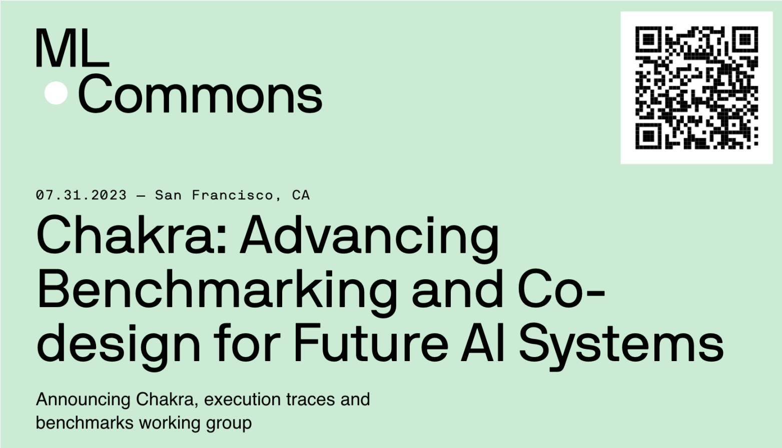


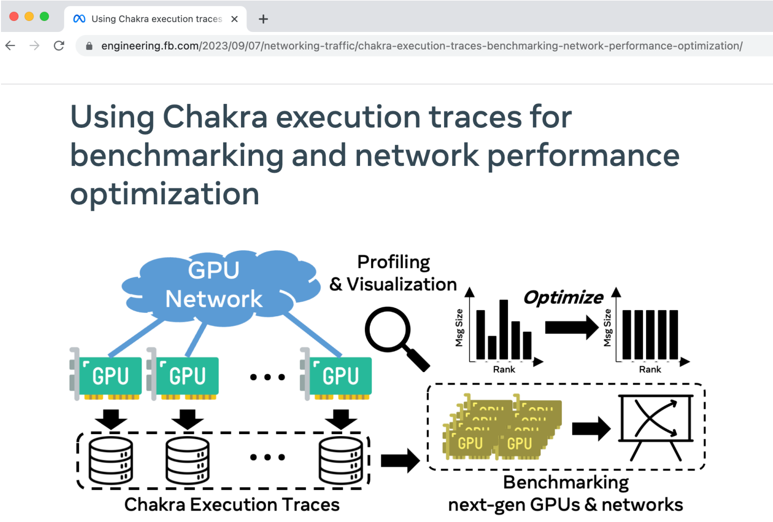
RAPID FIRE Talk Tomorrow at 1:55 PM!

Outline

- Design Space of AI Platforms
- Chakra Workload Execution Traces
- ASTRA-sim Layers and APIs
- **External Engagements**
- Conclusion

Chakra has been adopted by MLCommons





Blog Article from Meta

- **MLCommons** helps define industry-standard ML benchmarks
  - Maintainer of MLPerf
- **Chakra Working Group in MLCommons** **Please join!**
  - **Chairs:** Tushar Krishna (GT) + Srinivas Sridharan (NVIDIA)
  - **Active Members (and Users):** NVIDIA, AMD, Intel, Meta, ByteDance, HPE, Juniper, Keysight, Marvell, (+startups), ..

  https://mlcommons.org/working-groups/research/chakra/

- **Goals of Working Group**
  - Trace Format Standardization
    - Enable easier sharing between hyperscaler/cloud and vendors (with/without NDA)
    - Vendors can focus on different components (compute/memory/network)
  - Trace Collection support (PyTorch/TensorFlow/JAX)
  - Trace Replay and Simulation support tools
  - Trace Benchmark Suite Creation

ASTRA-sim Open-Source Community

Wiki

- https://astra-sim.github.io/astra-sim-docs/index.html

Github

- https://github.com/astra-sim/astra-sim

Tutorials

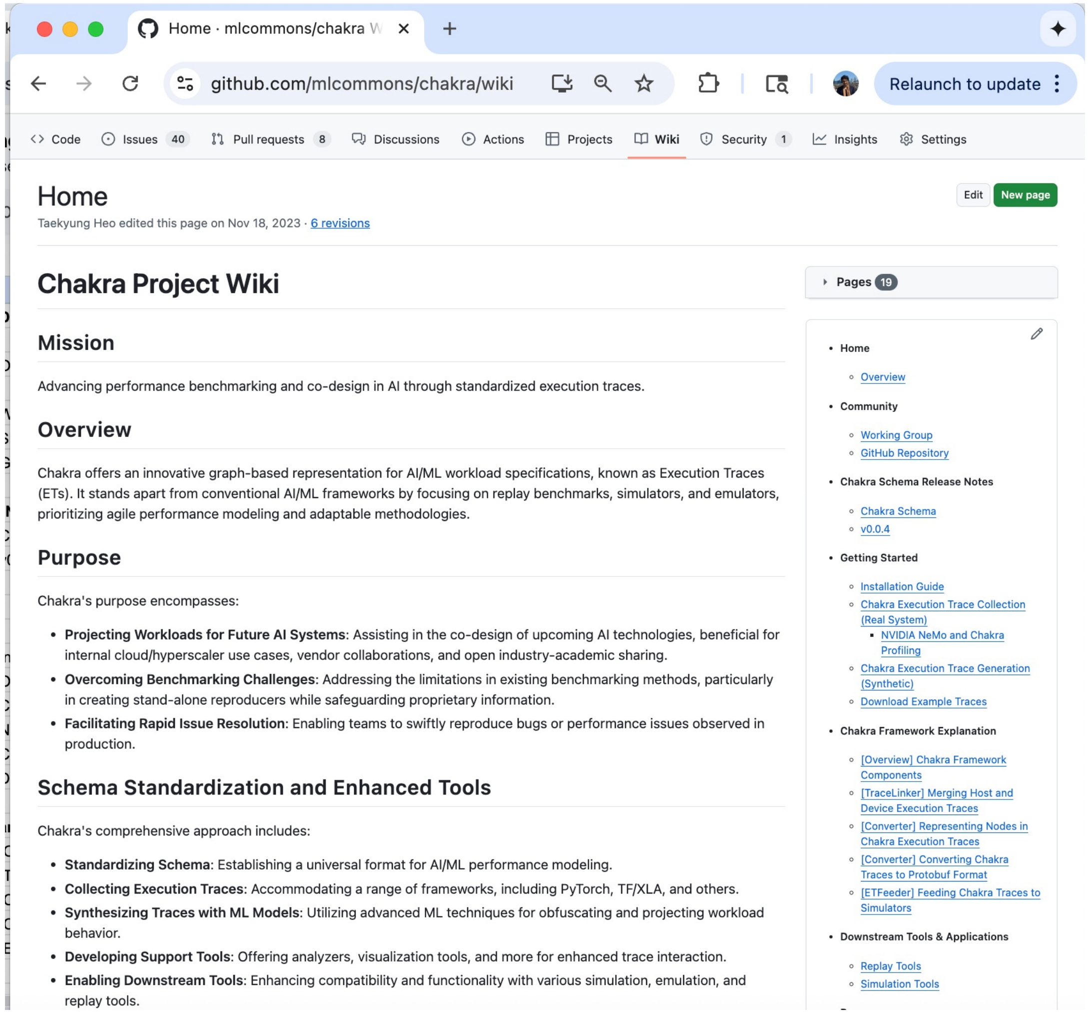
- ASPLOS (2022, 2023), ISCA (2022), MLSys (2022), HotI (2024), MICRO (2024)
- https://astra-sim.github.io/tutorials

Growing Userbase
(410 stars on github, 150 forks)

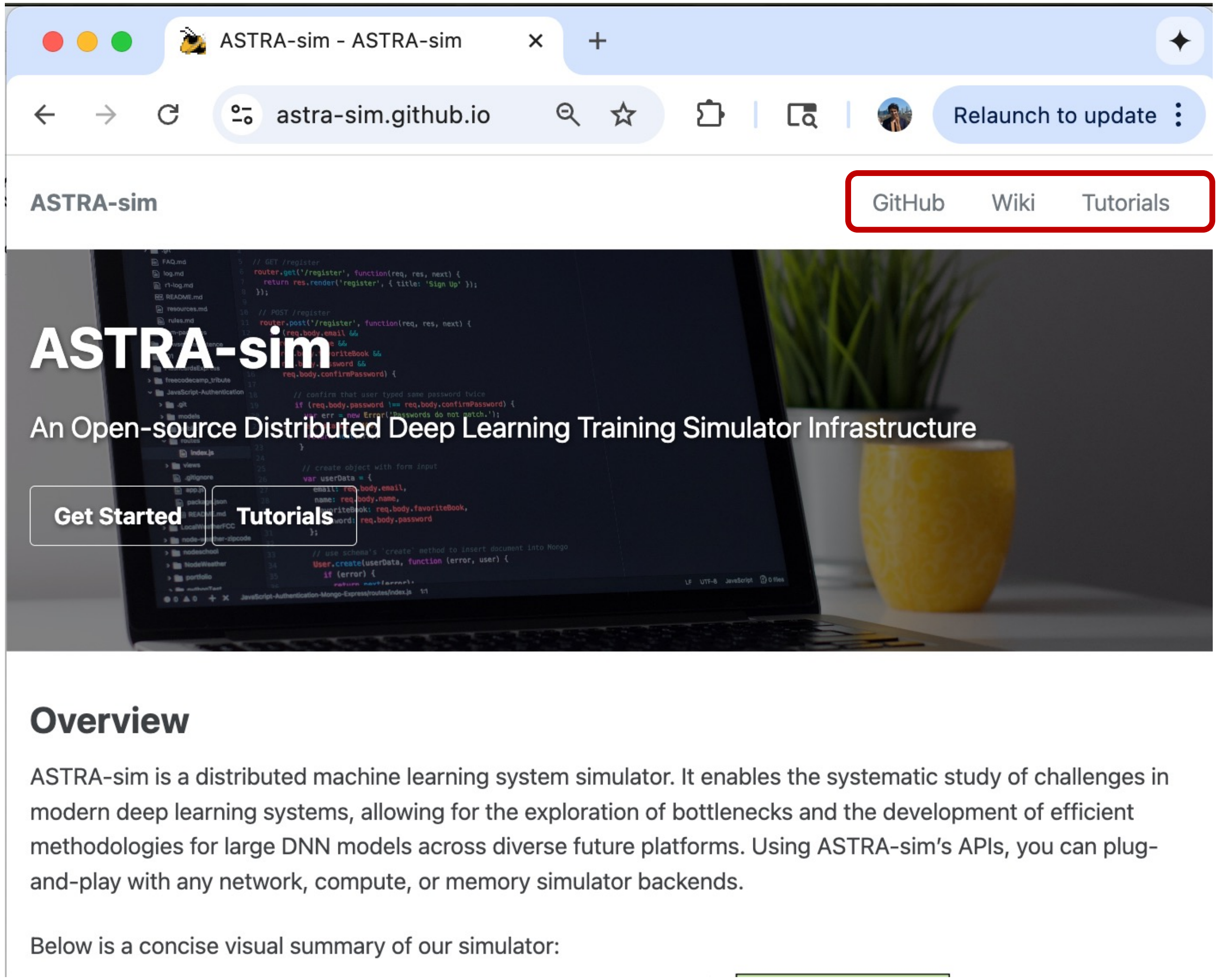
- Industry
  - Google, Meta, HPE, Alibaba, ByteDance
  - AMD*, NVIDIA, Intel, IMEC*, Qualcomm
  - Keysight*, Micron, Marvell*, Samsung
  - …
- Several startups
- Open Compute Project (OCP) WG on Co-Design
- ML Commons Chakra WG
- Many universities

**Contributing new features*

Resources



https://github.com/mlcommons/chakra/wiki



https://astra-sim.github.io/

Outline

- Design Space of AI Platforms
- Chakra Workload Execution Traces
- ASTRA-sim Layers and APIs
- External Engagements
- **Conclusion**

Summary: Chakra and ASTRA-sim

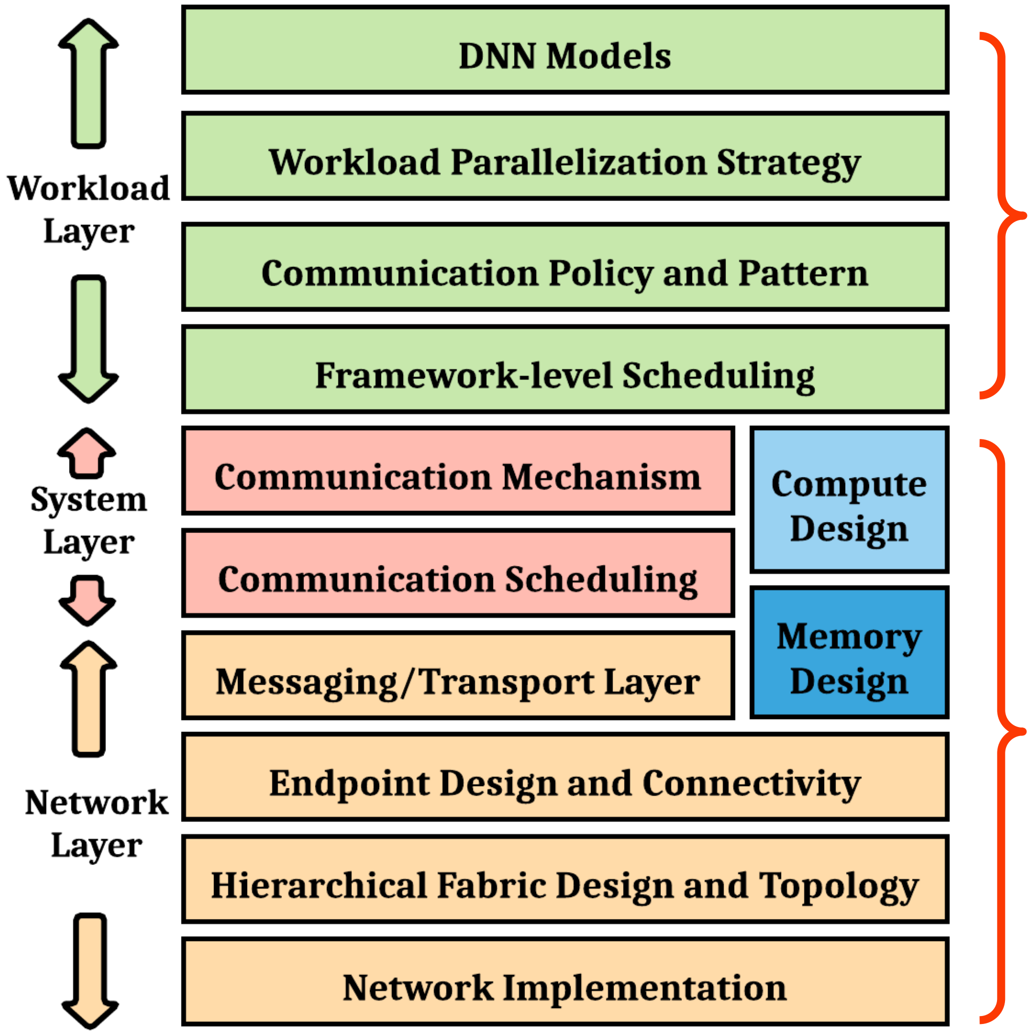


Figure Courtesy: Srinivas Sridharan (NVIDIA)

Chakra Execution Graph/Trace: an open graph-based representation of AI/ML workload execution

- enables isolation and optimization of compute, memory, communication behavior
- an ecosystem for benchmarking, performance analysis, and performance projection

ASTRA-sim: Distributed AI system simulator

- models distributed AI system co-design stack
- allows mix-and-match of performance models for compute, memory and network (API-based)

Thank you!