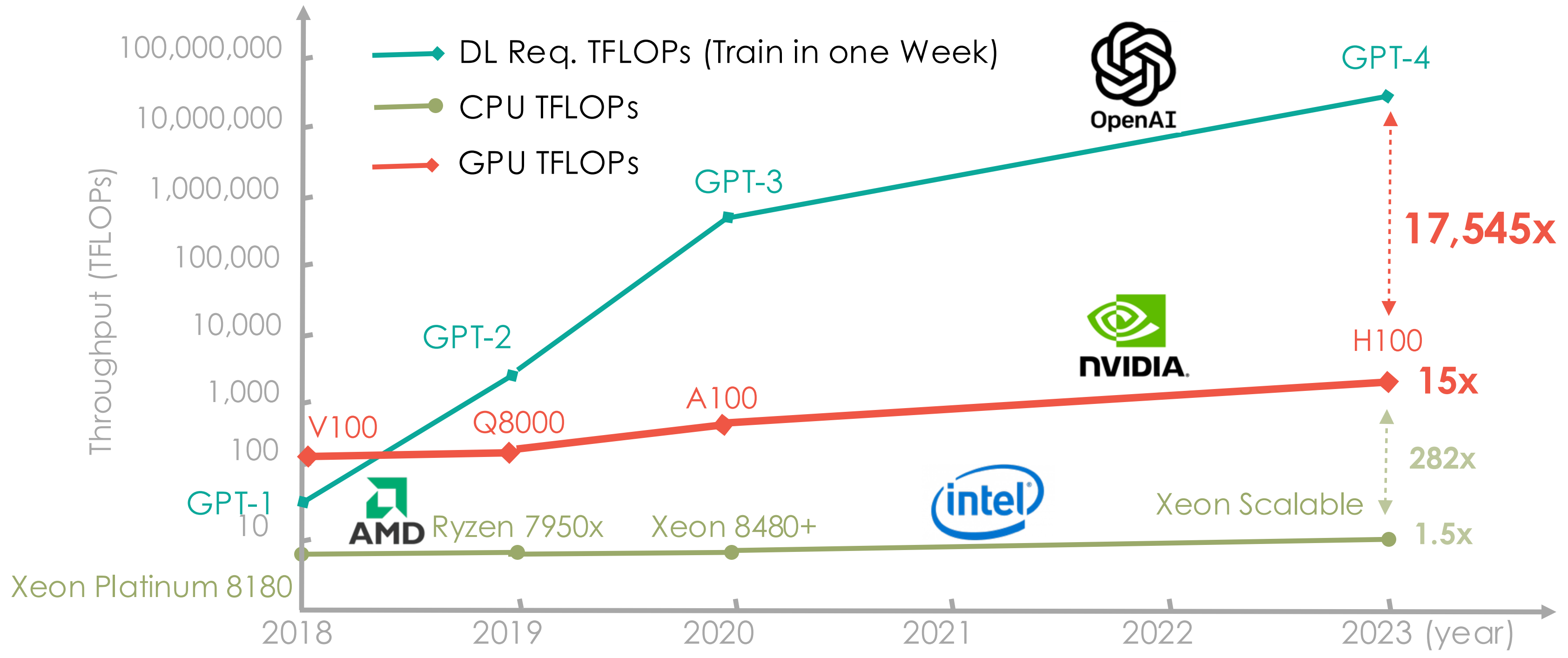
Systematic Approaches for Efficient and Scalable Deep Learning

Yuke Wang@Rice CS

The Trend of DL Algorithm and Hardware

❖ Recap of DL algorithms and hardware performance scaling.



Huge Potential with GPUs! But it still has a **Large Gap!**

Deep Learning Drives Computing Innovations

❖ Overview of my prior Ph.D. Research.

Algorithm Innovation

ICML'20,
CIKM'21,
AAAI'21

Model Design,
Model Pruning
(2018-19)

Hardware Innovation

TCAD'21a,
TCAD'21b,
CCGrid'21

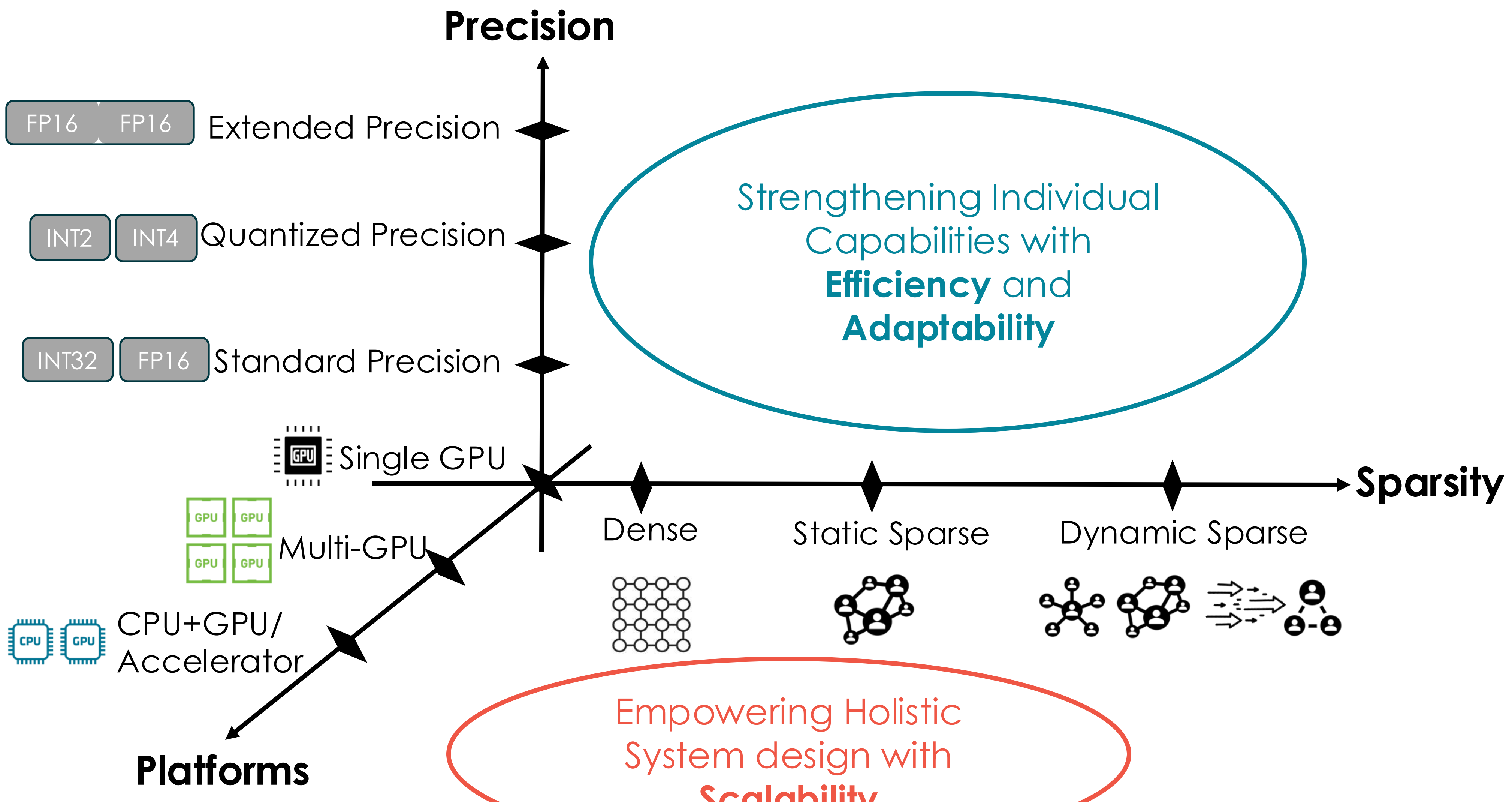
Hardware
Accelerators
(2020-21)

Software Innovation

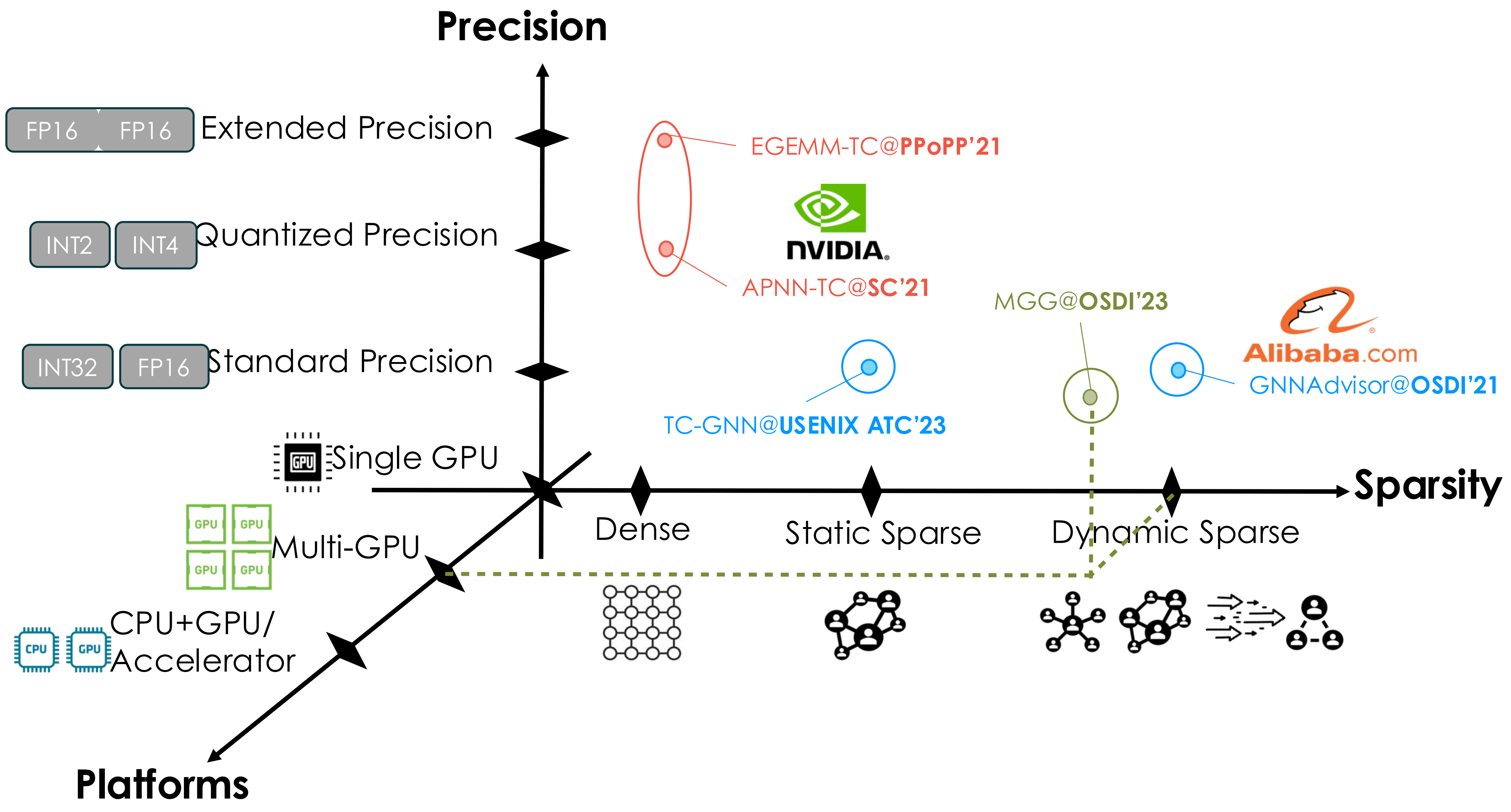
OSDI'21, SC'21
PPoPP'22, OSDI'23,
USENIX ATC'23

Runtime Systems,
Compiler
(2021-now)

My Prior PhD Research Recap



My Prior Research Recap



Precision

Sparsity

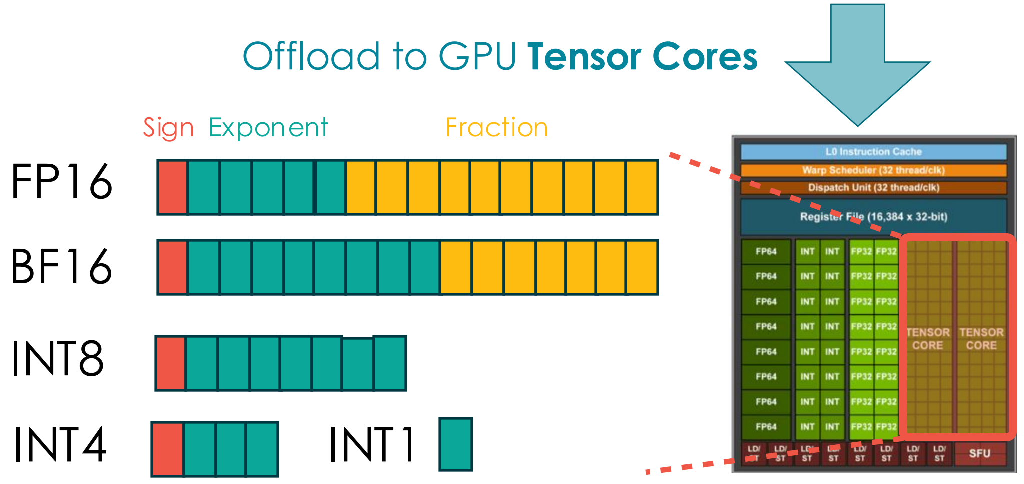
Scalability

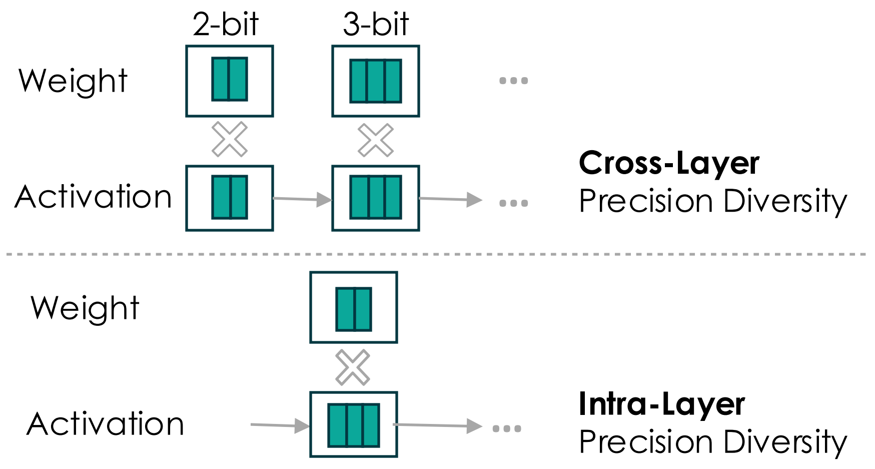
Diverse Precision Demands for DL Applications

❖**Low-precision** quantized deep-learning applications.

Quantized Deep Learning	Precision Requirements
QNNs [JMLR'18]	**1-bit Weight, 2-bit Activation** for Vision Model, **3-bit Weight, 4-bit Activation** for Language Model.
SGQuant [ICTAI'20]	Graph Attention Model: **2-bit** Neighbor **Attention**, **4-bit** Neighbor **Aggregation.**
LLM.int8() [NeurIPS'22]	**8-bit** Quantization for Transformers.
…	…

Offload to GPU **Tensor Cores**


Sign
Exponent
Fraction
FP16
BF16
INT8
INT4
INT1
L0 Instruction Cache
Warp Scheduler (32 thread/clk)
Dispatch Unit (32 thread/clk)
Register File (16,384 x 32-bit)
FP64
INT
FP32
TENSOR CORE
LD/ST
SFU

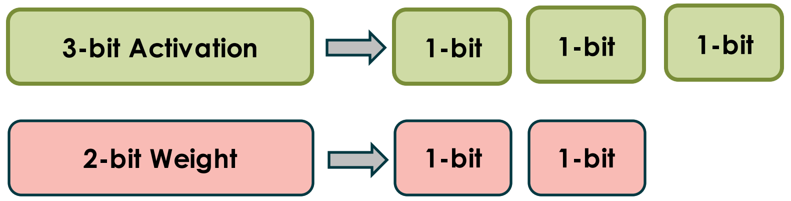

2-bit
3-bit
Weight
Activation
…
Cross-Layer Precision Diversity
Weight
Activation
…
Intra-Layer Precision Diversity

Low-precision deep-learning applications **can** leverage low-precision GPU Tensor cores, but suffer from **low efficiency**.

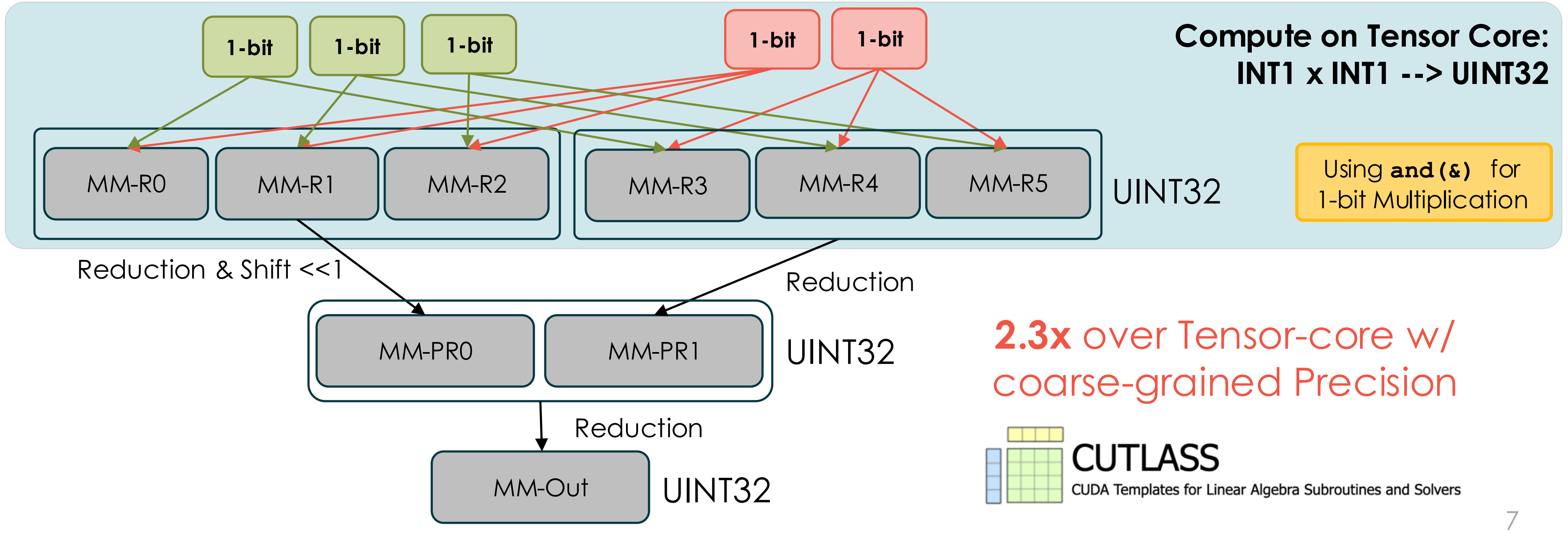
Bit Composition for Quantized Deep Learning [SC'21]

❖ Insight: **Quantized deep learning** can be **composed** with the **binary (1-bit) precision**.

Example of **2-bit** and **3-bit** Precision in Quantized DNN Computing.



Multiplication between 3-bit Activation and 2-bit Weight



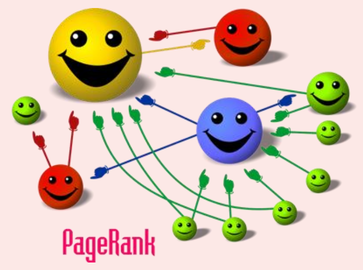
2.3x over Tensor-core w/ coarse-grained Precision

CUTLASS
CUDA Templates for Linear Algebra Subroutines and Solvers

A Typical Paradigm of Graph Deep Learning

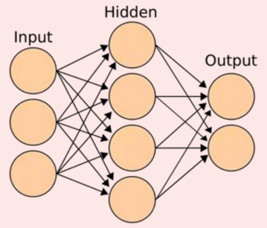
Operation view

GNN Layer =



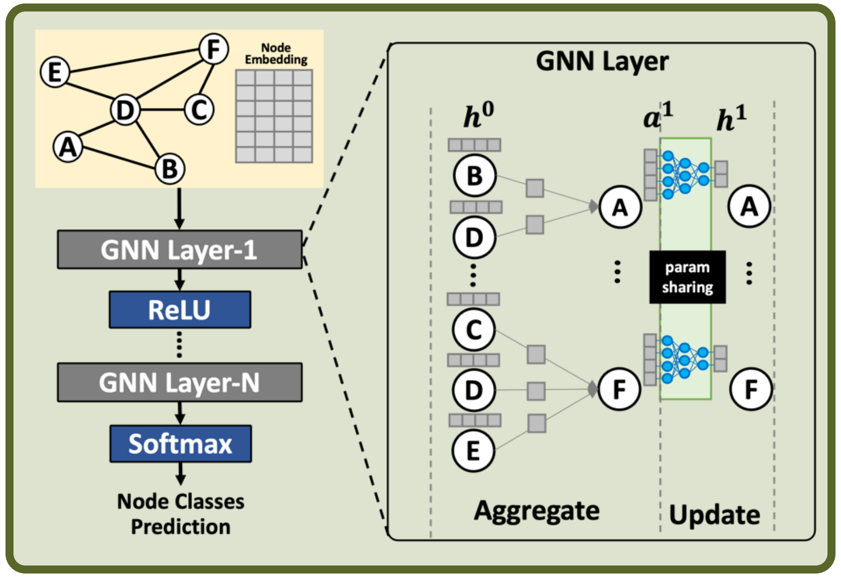
Graph Ops

+

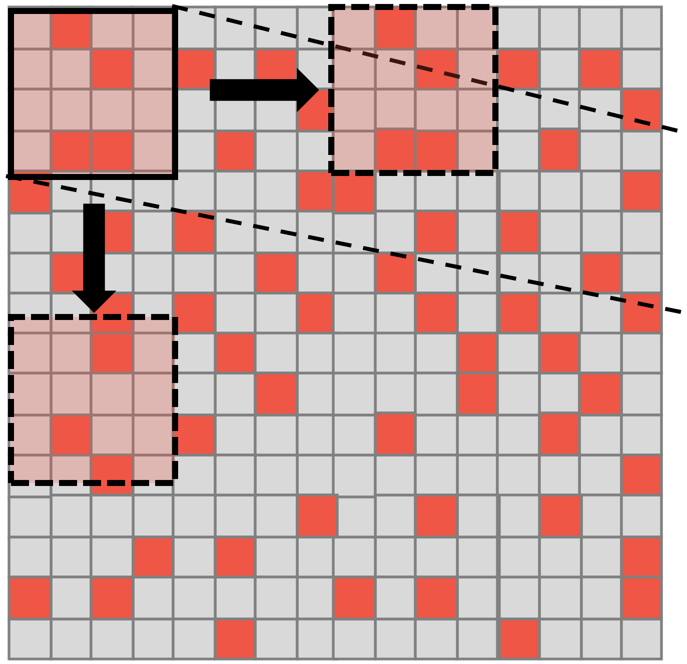


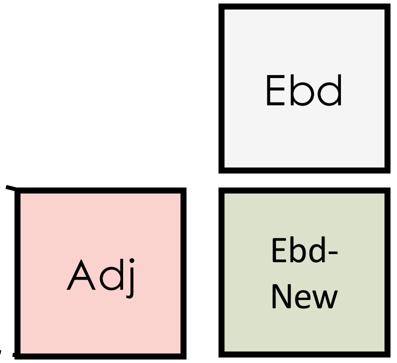
DNN Ops

Model view



Challenge of Mapping Sparse Computing to Dense Units


Sparse Adjacent Matrix of Graph



>> A100/H100 (80GB)

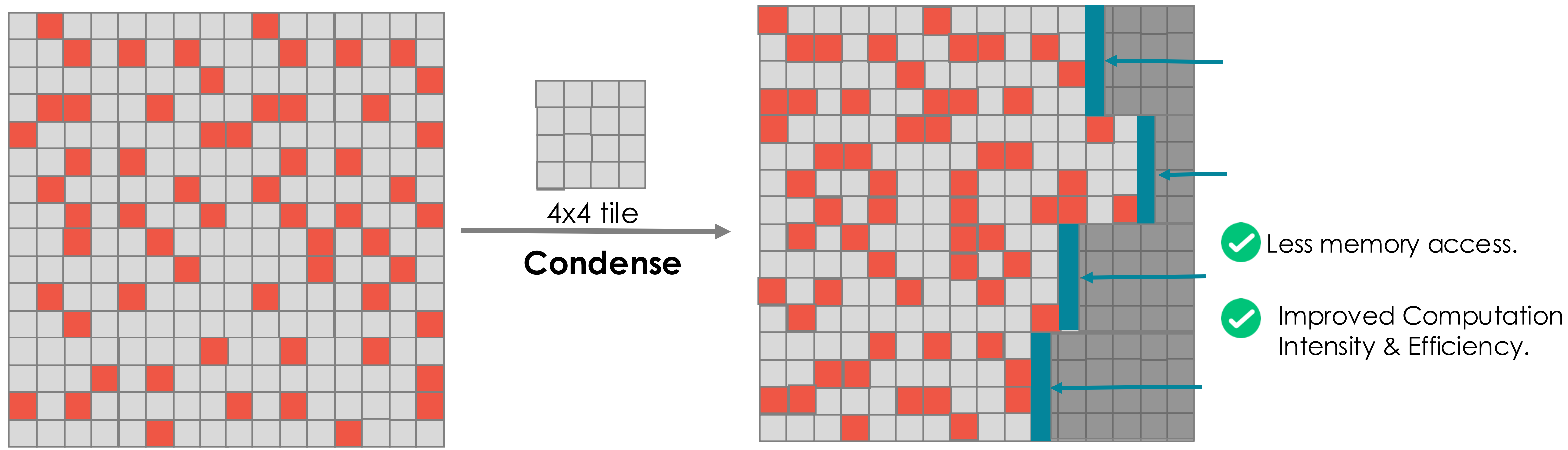
Dataset	# Nodes	# Edges	Memory	Eff.Comp
OVCR-8H	1,890,931	3,946,402	14302.48 GB	0.36%
Yeast	1,714,644	3,636,546	11760.02 GB	0.32%
DD	334,925	1,686,092	448.70 GB	0.03%

Largely **Wasted** Computation and Memory Access

Direct mapping suffers from **extra high memory consumption** and **extremely low computing efficiency**.

TC-GNN: Order-Invariant Transformation [ATC'23]

- Irregularly-scattered elements can be **condensed** to benefit high-performance dense GPU units.



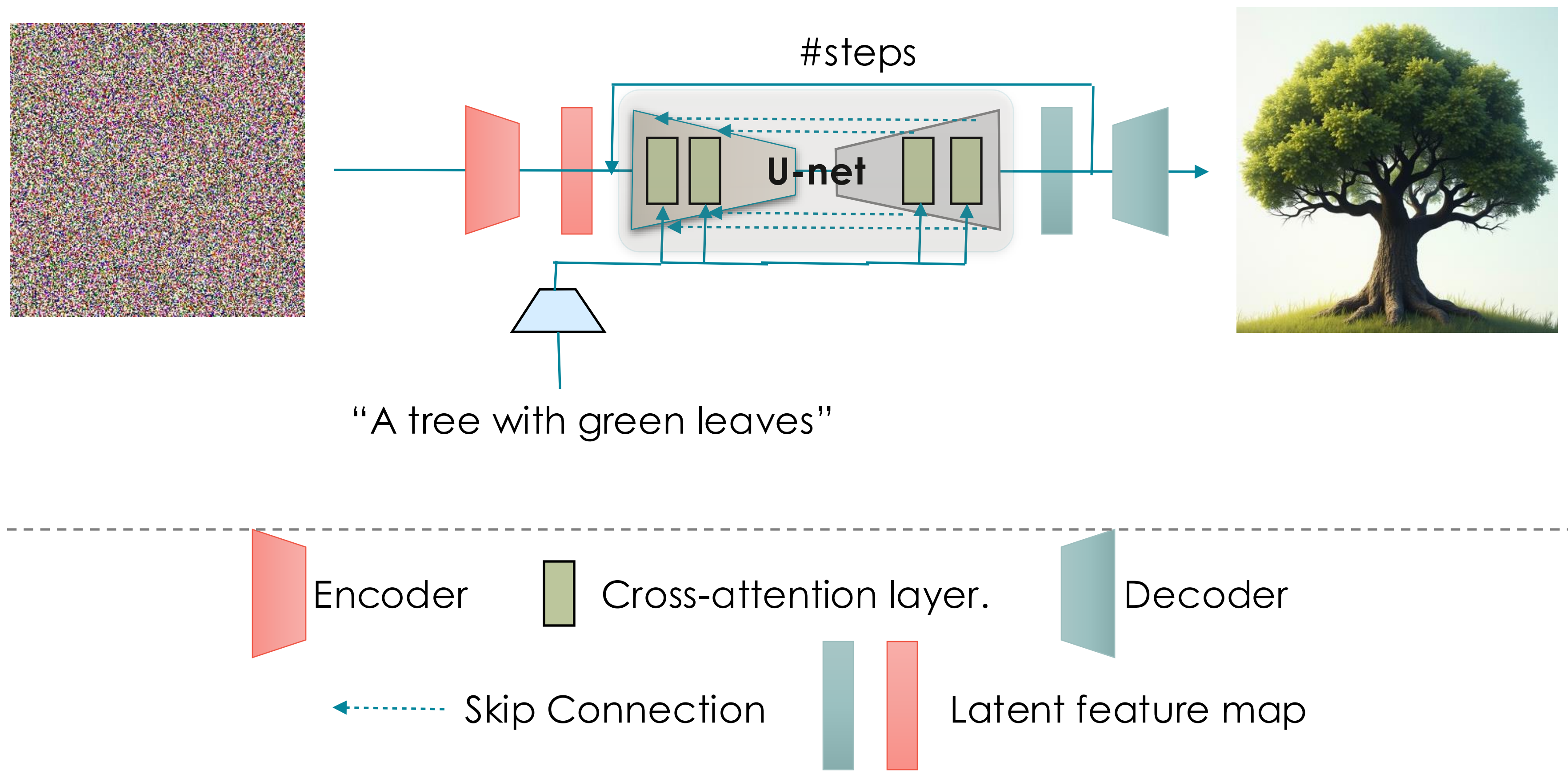
Original Sparse Adjacent Matrix of Graph

Condensed Adjacent Matrix for Tensor Core

1.50x ~ 6.70x over DGL operators (cuSPARSE).
Incorporated by SparseTIR in **TVM** Project.

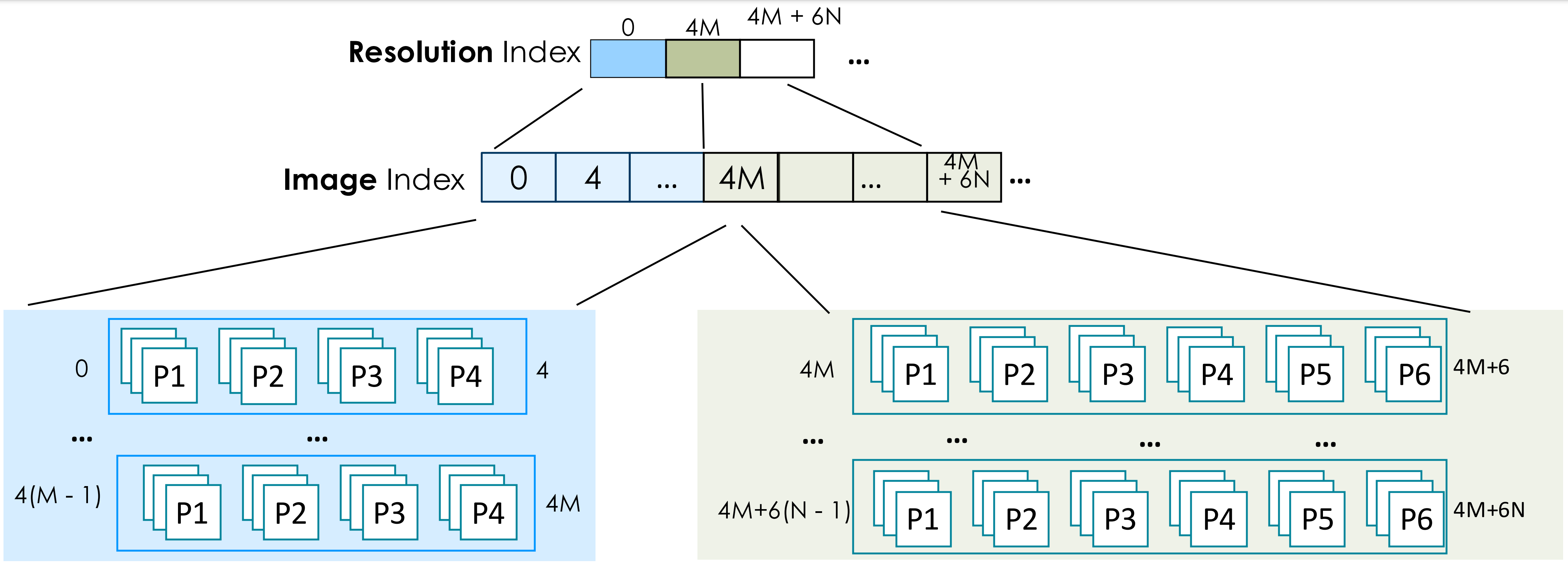
R1: Patch-based Stable Diffusion Serving

- Architecture of Stable Diffusion Model.

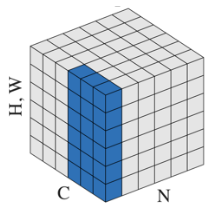


Note that Conv are omitted in Unit for simplicity.

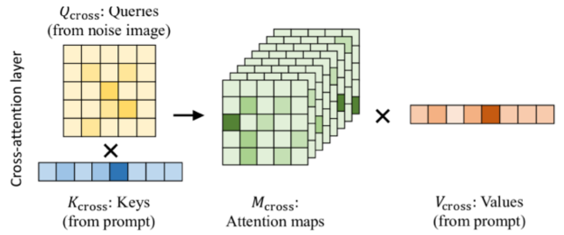
R1: Patch-based Stable Diffusion Serving



Group Norm

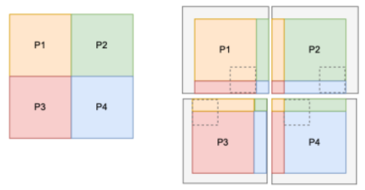


Cross Attention



Q_{cross}: Queries (from noise image)
Cross-attention layer
K_{cross}: Keys (from prompt)
M_{cross}: Attention maps
V_{cross}: Values (from prompt)

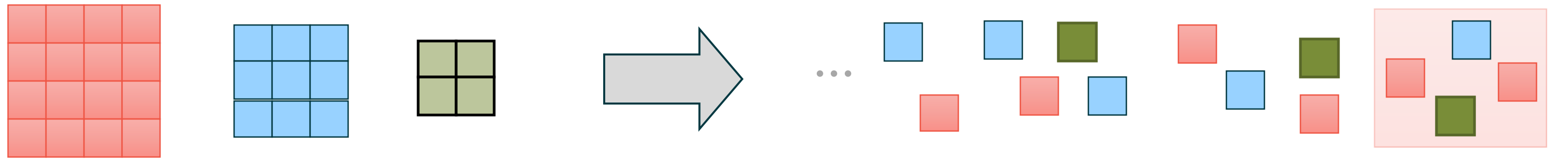
Convolution



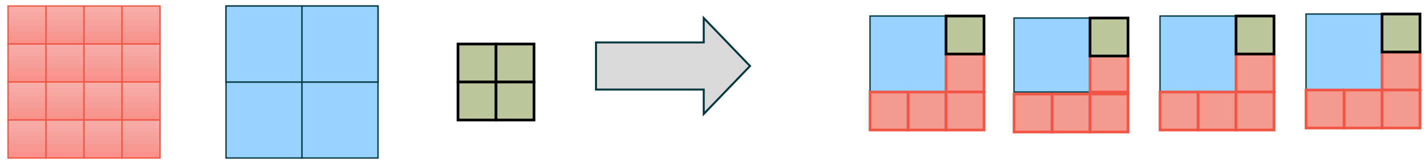
Note that the common factor of C*H*W is omitted for simplicity.

R1: Patch-based Stable Diffusion Serving

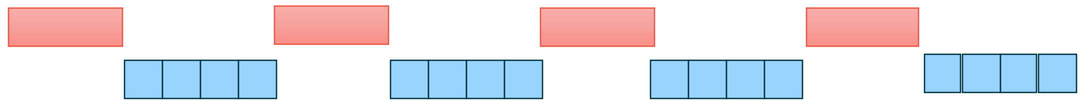
Unify the processing of different resolutions



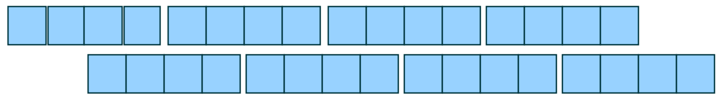
Control granularity for mixed workload composition



Unlock more fine-grained pipelining (FCFS)



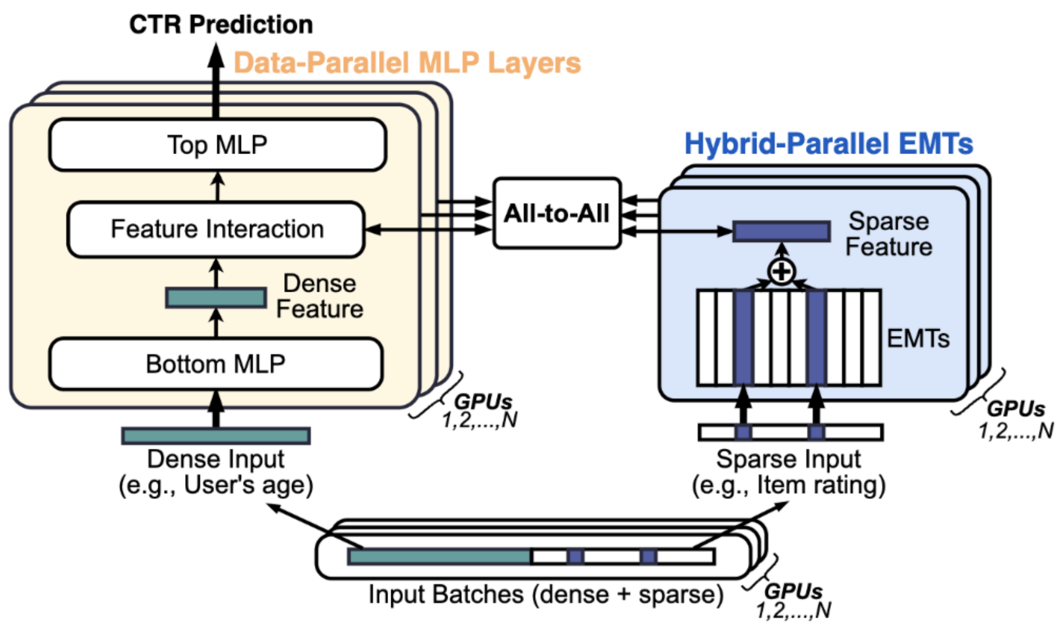
a. Resolution-mismatch for largely sequent processing.

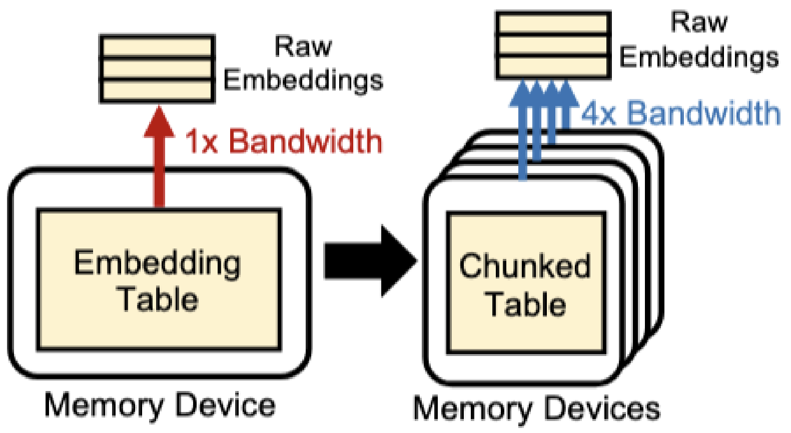


b. Decomposed Resolution for batched processing.

R2: Efficient DLRM with CXL Disaggregated Memory

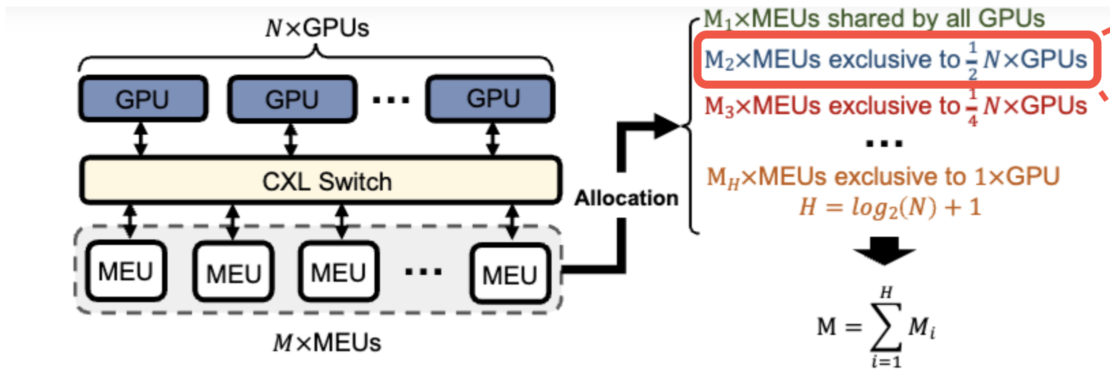
- Typical **architecture** and **configuration** of DLRMs.



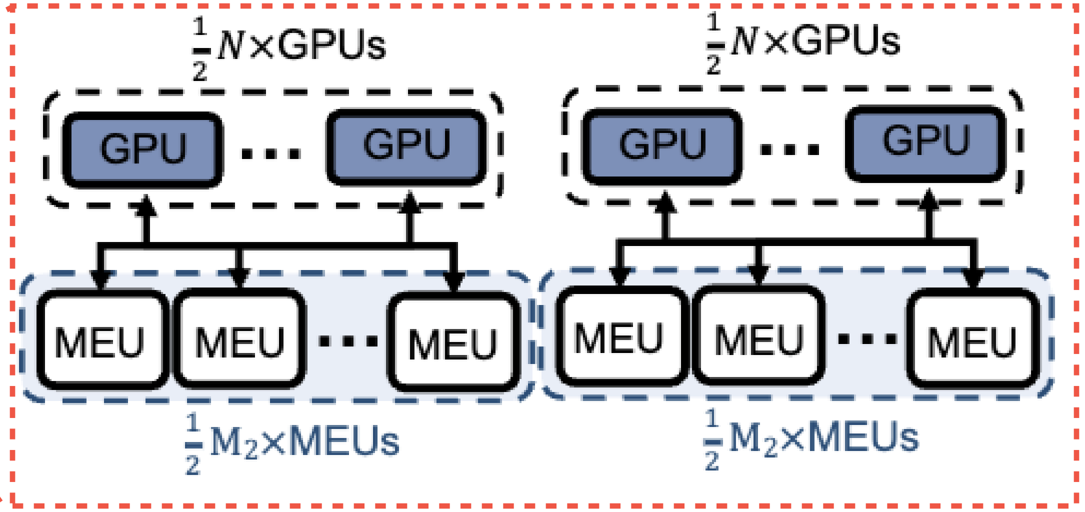


R2: Joint Optimization with Disaggregated Memory

- **Hierarchical** memory layout with **fine-grained access control** enlarges design space.

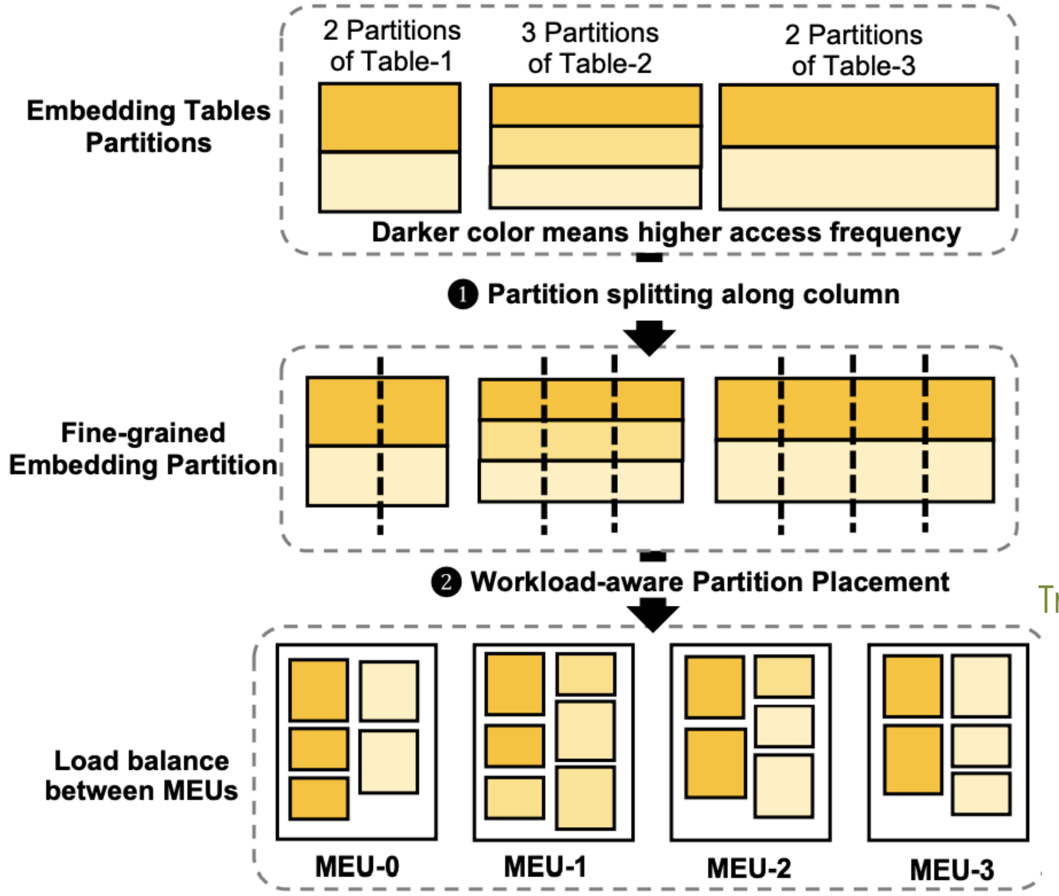

N×GPUs
GPU
GPU
GPU
CXL Switch
MEU
MEU
MEU
MEU
M×MEUs
Allocation
M_1×MEUs shared by all GPUs
M_2×MEUs exclusive to $\frac{1}{2}N$×GPUs
M_3×MEUs exclusive to $\frac{1}{4}N$×GPUs
M_H×MEUs exclusive to 1×GPU
$H = log_2(N) + 1$
$M = \sum_{i=1}^{H} M_i$

H: #Hierarchy



R2: Joint Optimization with Disaggregated Memory

❖ **Workload-aware** Embedding Table Placement.


2 Partitions of Table-1
3 Partitions of Table-2
2 Partitions of Table-3
Embedding Tables Partitions
Darker color means higher access frequency
❶ Partition splitting along column
Fine-grained Embedding Partition
❷ Workload-aware Partition Placement
Load balance between MEUs
MEU-0
MEU-1
MEU-2
MEU-3

❖ ILP formulation for joint memory and table optimization.

Memory Unit Placement

$$\sum_{j=1}^{H} m_{ij} = 1,\ m_{ij} \in \{0,1\}$$

Embedding Table Assignment

$$\sum_{j=0}^{H} t_{ij} = 1,\ t_{ij} \in \{0,1\}$$

H is the number of memory hierarchy

Memory Constraints

$$Cap_j \leq Mem_j \quad j \in \{0,1,\cdots,H\}$$

$$Cap_j = \begin{cases} N * C_G\ , & j = 0 \\ \sum_{i=1}^{M} m_{ij} * C_M * \frac{1}{2^{j-1}}\ , & j \in \{1,2,\cdots,H\} \end{cases}$$

$$Mem_j = \sum_{i=1}^{T} t_{ij} * S_i * D_i * size(float) \quad j \in \{0,1,\cdots,H\}$$

Optimize Transfer & Lookup Efficiency

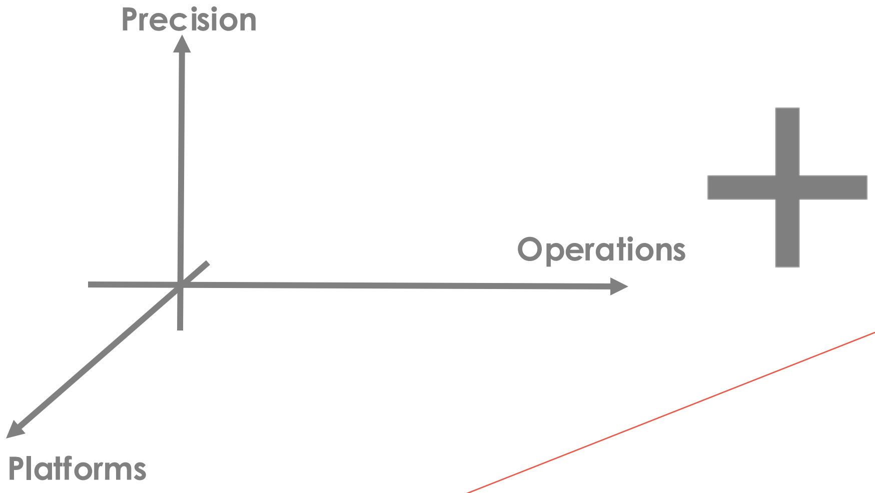
$$\boldsymbol{minimize}\ \ Max(L\_send_j + L\_lookup_j) \quad j \in \{0,1,\cdots,H\}$$

$$L\_send_j = \begin{cases} 0, & j = 0 \\ \frac{B * size(float) * \sum_{i=1}^{K} D_i}{BW_{CXL}}, & j \in \{1,2,\cdots,H\} \end{cases}$$

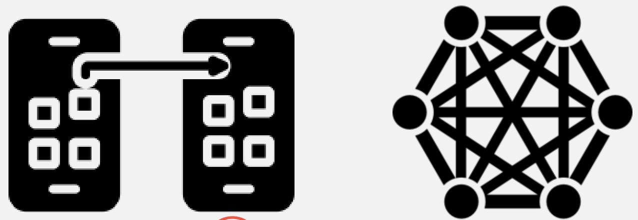
$$L\_lookup_j = \frac{\sum_{i=1}^{T} t_{ij} * B * A_i * D_i * P_i * size(float)}{BW_j}$$

$$BW_j = \begin{cases} BW_{GPU}, & j = 0 \\ Min(\sum_{i=1}^{M} m_{ij} * BW_{MEU}, & BW_{max} * 2^{j-1}), \\ & j \in \{1,2,\cdots,H\} \end{cases}$$

Future Research: New Hardware-System Optimization



Portability (Sustainable AI)



New **Hardware-System** Designs that adapt to diverse real-world application settings.

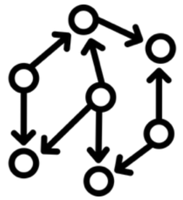
Efficient DLRM with **Disaggregated Memory (e.g., CXL)**

- Co-design/optimization with diverse accelerators.
- Co-scheduling Workloads with Near-data Processing.
- Multi-Tenant Support for co-locating Diverse DL applications.

Eco-friendly DL with **Energy/Carbon-aware HW-System** Co-Optimization

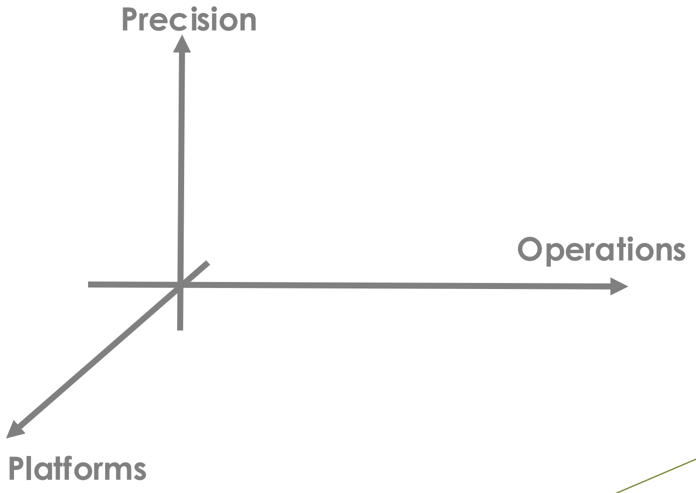
- Dynamic Energy Scaling to balance runtime performance and energy cost.
- Carbon-Aware Scheduling to balance energy availability and job requirements.

Accelerated DL with **Reconfigurable Dataflow Architecture** (e.g., SambaNova)



- Compiler optimization for dataflow design space search.
- Workload-aware Runtime Dataflow Reconfiguration.

Future Research: Exploring New DL Workloads



+

Portability (Sustainable AI)

N
t

Satisfaction (Human-Centric AI)

...

New **Algorithm-System** designs that can intelligently suit various demands.

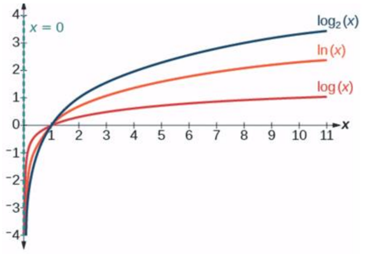
Patch-based Diffusion Model Serving

- SLO-aware dynamic re-patching for efficient parallelism.
- Patch-based early exit for efficient diffusion.

Locality-aware Text-to-3D Scene Construction

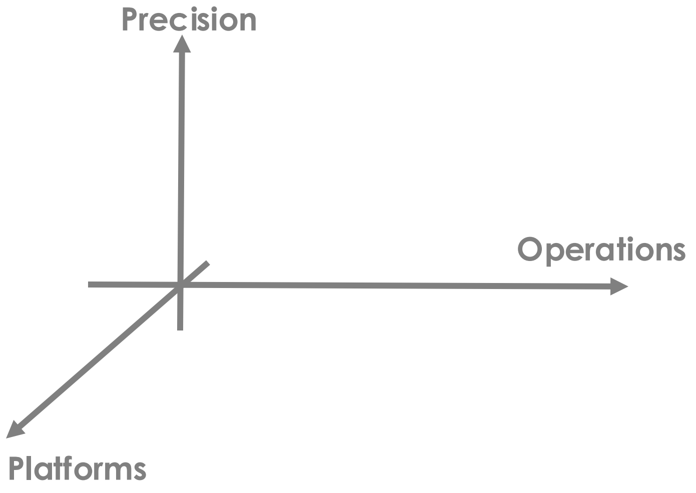
- Cross-iteration spatial and temporal in 360 Panorama rendering and 3D scene lifting.
- Cross device (e.g., multi-GPU platforms) objects artifacts locality.

Resource-Efficient DL with Logarithmic Number System (LNS)



- Dynamic DL precision with multi-base LNS.
- Memory-efficient model quantization with selective LNS.

Future Future: Secure and Resilient DL System



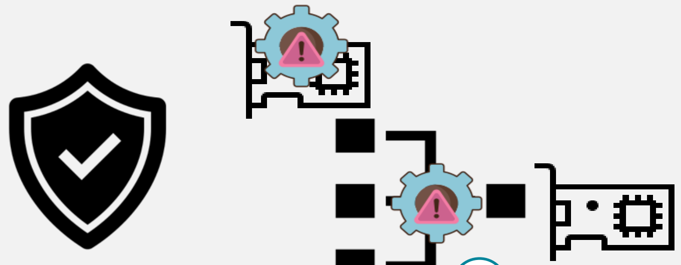
+

...

Portability (Sustainable AI)

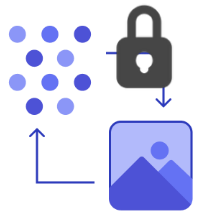
Satisfaction (Human-Centric AI)

Safety (Trustworthy AI)



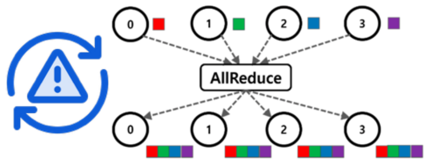
System that are **Robust** and **Resilient**.

Confidential/Encrypted DL Model Serving



- Holistic scheduling of encryption and regular DL ops for low-cost DL model training.

Failure-resilient Collective Communication



- Failure-resilient routine for heterogenous memory and compute devices.

Efficient **Watermarking** for IP-Protected DL.

...

- Robustness-aware Watermarking for efficient real-time processing.

Thank You

Q & A

yuke.wang@rice.edu

github.com/YukeWang96

wang-yuke.com