COLOMBIAN HPC SUMMER SCHOOL

Argonne NATIONAL LABORATORY

STATE OF AI ACCELERATORS

SIDDHISANKET RASKAR
Assistant Computer Scientist
Argonne Leadership Computing Facility
Argonne National Laboratory
Chicago, Illinois, USA.

U.S. DEPARTMENT OF ENERGY Argonne National Laboratory is a U.S. Department of Energy laboratory managed by UChicago Argonne, LLC.

June 2025

Outline

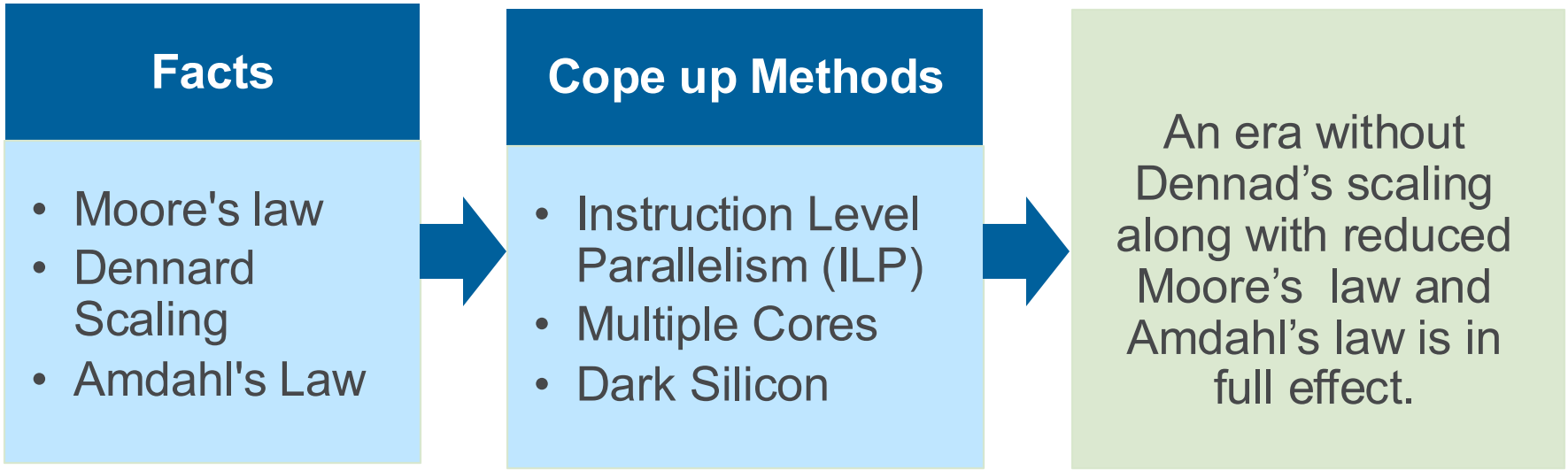
- Motivation
- Overview of AI Accelerators
- Benchmarking Results
- Accelerators for HPC
- Concluding Remarks

“If a problem has no solution,
it may not be a problem, but a fact—

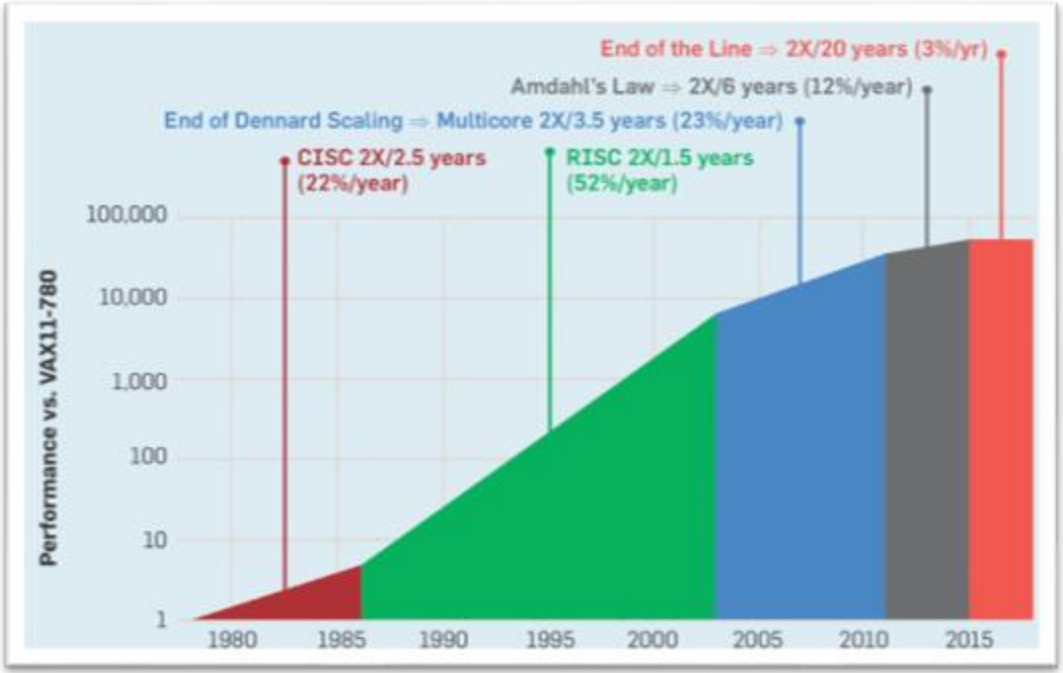
not to be solved,
but to be coped with over time.”

—Shimon Peres

MOTIVATION


Facts
Moore's law
Dennard Scaling
Amdahl's Law
Cope up Methods
Instruction Level Parallelism (ILP)
Multiple Cores
Dark Silicon
An era without Dennad's scaling along with reduced Moore's law and Amdahl's law is in full effect.

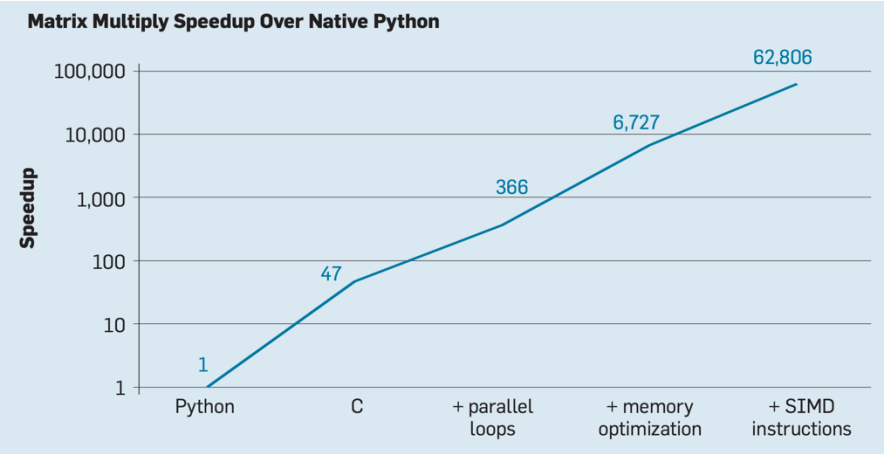
MOTIVATION



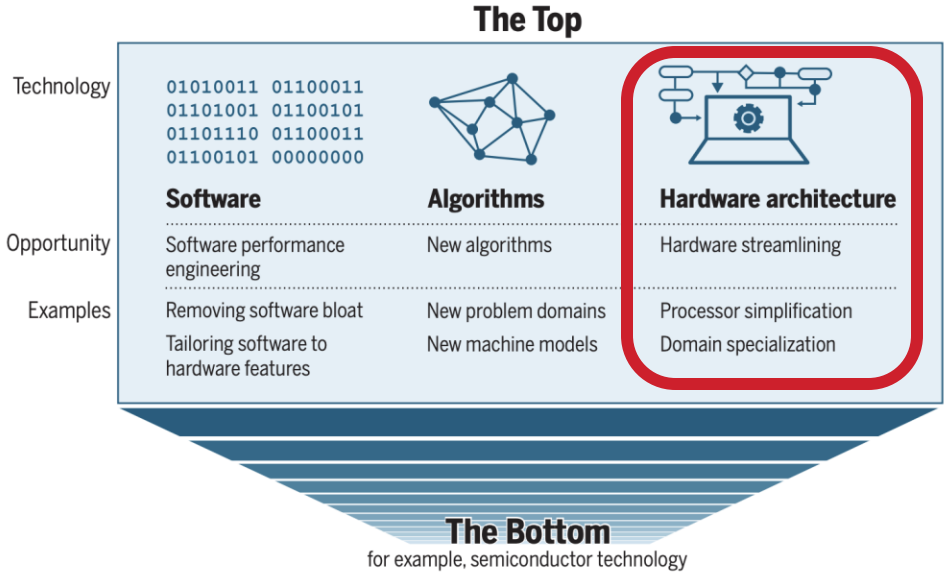
Growth of computer performance

An era without Dennad's scaling along with reduced Moore's law and Amdahl's law is in full effect.

NEXT COPE UP METHODS



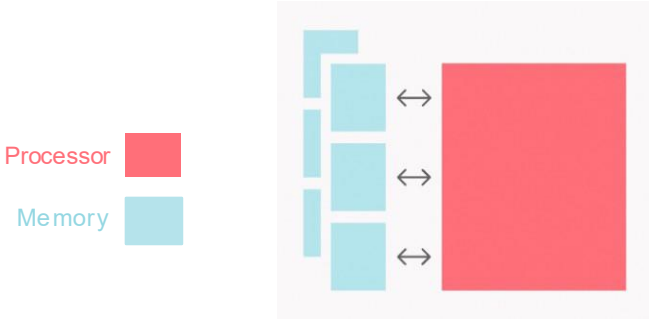
Better Software and algorithms



Domain Specific Architectures and Languages

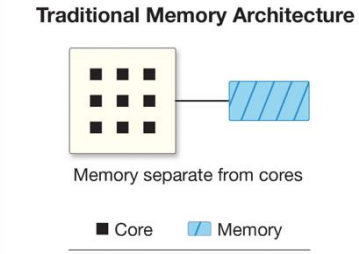
- Charles E. Leiserson et al. ,There's plenty of room at the Top: What will drive computer performance after Moore's law?. Science368, eaam9744(2020). DOI:10.1126/science.aam9744
- John L. Hennessy and David A. Patterson. 2019. A new golden age for computer architecture. Commun. ACM 62, 2 (February 2019), 48–60. https://doi.org/10.1145/3282307

CPU, GPU & AI ACCELERATORS

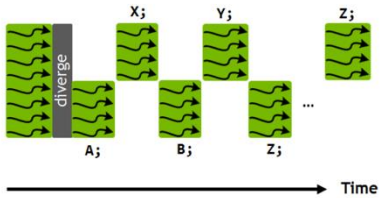
	CPU	GPU	AI Accelerators
Parallelism	Designed for scalar processing	SIMD/SIMT architecture. Designed for large blocks of dense contiguous data	Massively parallel MIMD architecture. High performance/efficiency for future ML trends
			
Memory Bandwidth	Off-chip memory	Model and Data spread across off-chip and small on-chip cache and shared memory (2TB/s for A100 HBM)	Main Model & Data in tightly coupled large locally distributed SRAM (Multiple TB/s on-chip bandwidth)

VON NEUMANN VS SPATIAL ARCHITECTURES

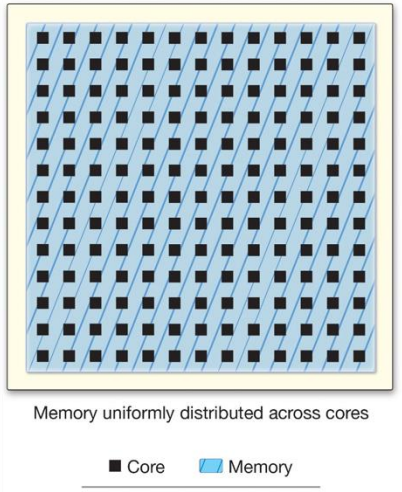
- Limitations of Traditional Architectures
- Heavy data movement leads to Increased Energy Cost in GPUs




```
if (threadIdx.x < 4) {
    A;
    B;
} else {
    X;
    Y;
}
Z;
```



Thread Divergence



Rise of domain-specific dataflow inspired architectures

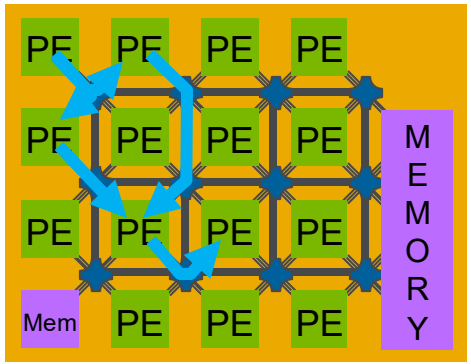
DL Workload Characteristics

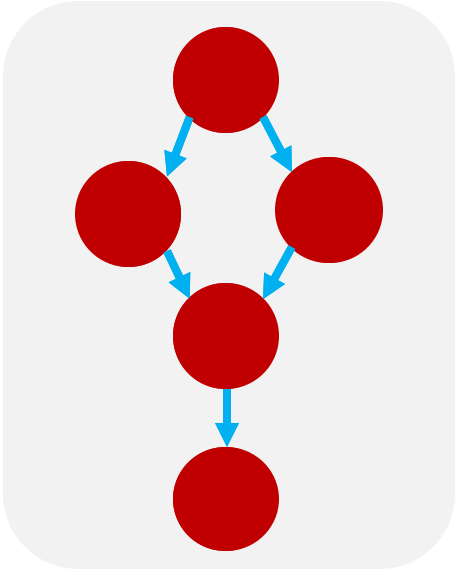
- Sparsity of data: makes computation and memory access patterns irregular
- Memory wall in LLM: SIMT approach makes data movement inefficient
- Conditional data paths result in thread divergence making execution inefficient
- GPU Thread management: SIMT requires threads to reconverge after divergence causing overheads

SPATIAL RECONFIGURABLE ARCHITECTURES

Workflow

- Program is represented as a graph
- This program graph is mapped on the architecture

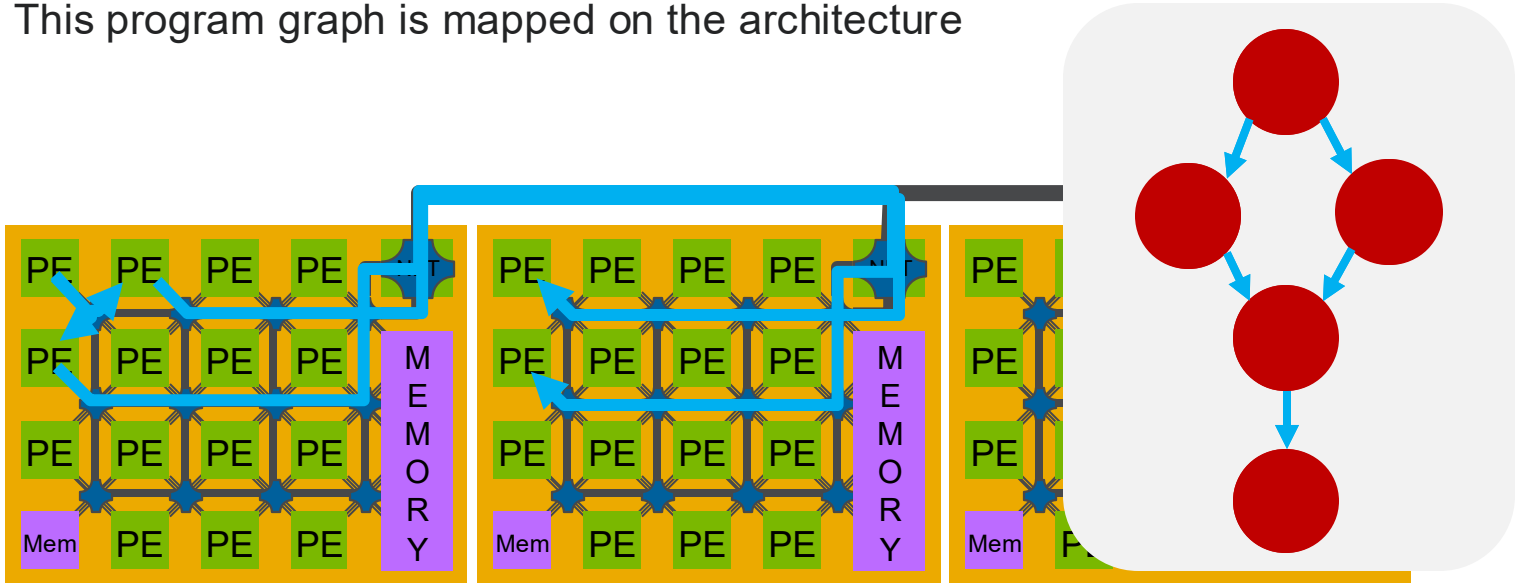




SPATIAL RECONFIGURABLE ARCHITECTURES

Workflow

- Program is represented as a graph
- This program graph is mapped on the architecture



ALCF AI Testbed

https://www.alcf.anl.gov/alcf-ai-testbed

Cerebras
CS-2

SambaNova
DataScale SN30

Graphcore
Bow Pod64

Habana Gaudi1

GroqRack

- **Cerebras:** 2 CS-2 nodes, each with 850,000 Cores, compute-intensive models
- **SambaNova:** DataScale SN30 8 nodes (8 SN30 RDUs per node) - 1TB mem per device, models with large memory footprint
- **Graphcore:** Bow Pod64 4 nodes (16 IPUs per node) - MIMD, irregular workloads such as graph neural networks
- **GroqRack:** 8 nodes, 8 GroqNodes per node - inference at batch 1
- **Habana Gaudi1:** 2 nodes, 8 cards per node - On-chip integration of RDMA over Converged Ethernet (RoCE2), scale-out efficiency

ALCF AI Testbed: Objectives

https://www.alcf.anl.gov/alcf-ai-testbed

Cerebras
CS-2

SambaNova
DataScale SN30

Graphcore
Bow Pod64

Habana Gaudi1

GroqRack

- Infrastructure of next-generation machines with AI hardware accelerators
- Provide a platform to evaluate usability and performance of AI4S applications
- Understand how to integrate AI systems with supercomputers to accelerate science

ALCF AI Testbed: Challenges

https://www.alcf.anl.gov/alcf-ai-testbed

Cerebras
CS-2

SambaNova
DataScale SN30

Graphcore
Bow Pod64

Habana Gaudi1

GroqRack

- Understand how these systems perform for different workloads given diverse hardware and software characteristics
- What are the unique capabilities of each evaluated system
- Opportunities and potential for integrating AI accelerators with HPC computing facilities

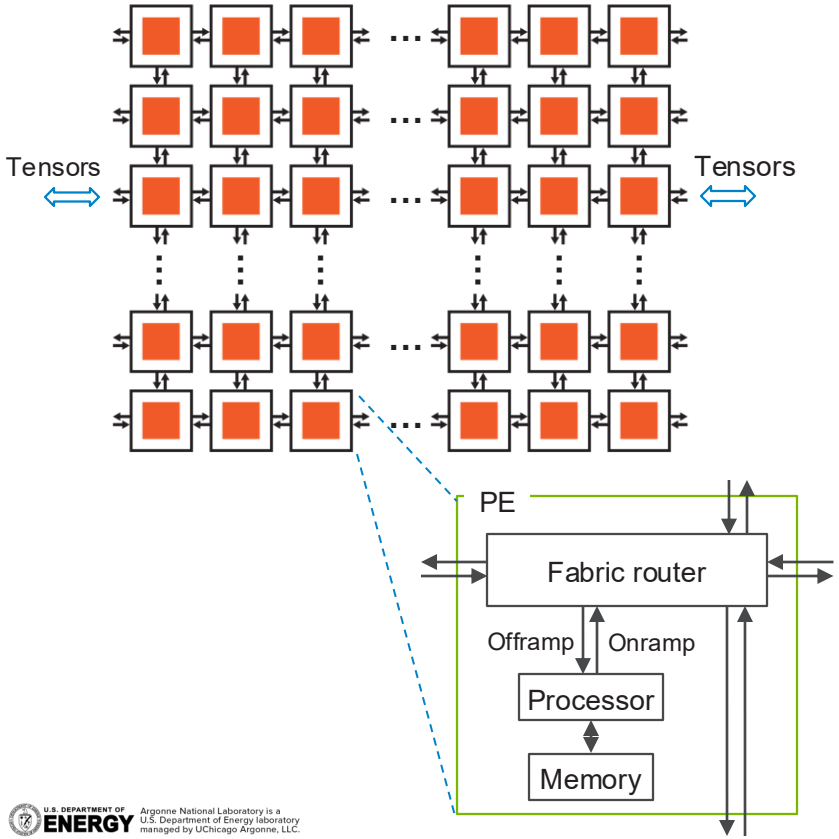
Outline

- Motivation
- Overview of AI Accelerators
- Benchmarking Results
- Accelerators for HPC
- Concluding Remarks

cerebras

- **850,000** cores optimized for sparse linear algebra
- **46,225 mm²** silicon
- **2.6 trillion** transistors, **7nm** process technology
- **40 gigabytes** of on-chip memory
- **20 PByte/s** memory bandwidth **220 Pbit/s** fabric bandwidth

WSE-2 ARCHITECTURE



The WSE appears as a logical 2D array of individually programmable Processing Elements

Flexible compute

- 850,000 general purpose CPUs
- 16- and 32-bit native FP and integer data types
- **Dataflow programming**: Tasks are activated or triggered by the arrival of data packets

Flexible communication

- Programmable router
- Static or dynamic routes (**colors**)
- Data packets (**wavelets**) passed between PEs
- 1 cycle for PE-to-PE communication

Fast memory

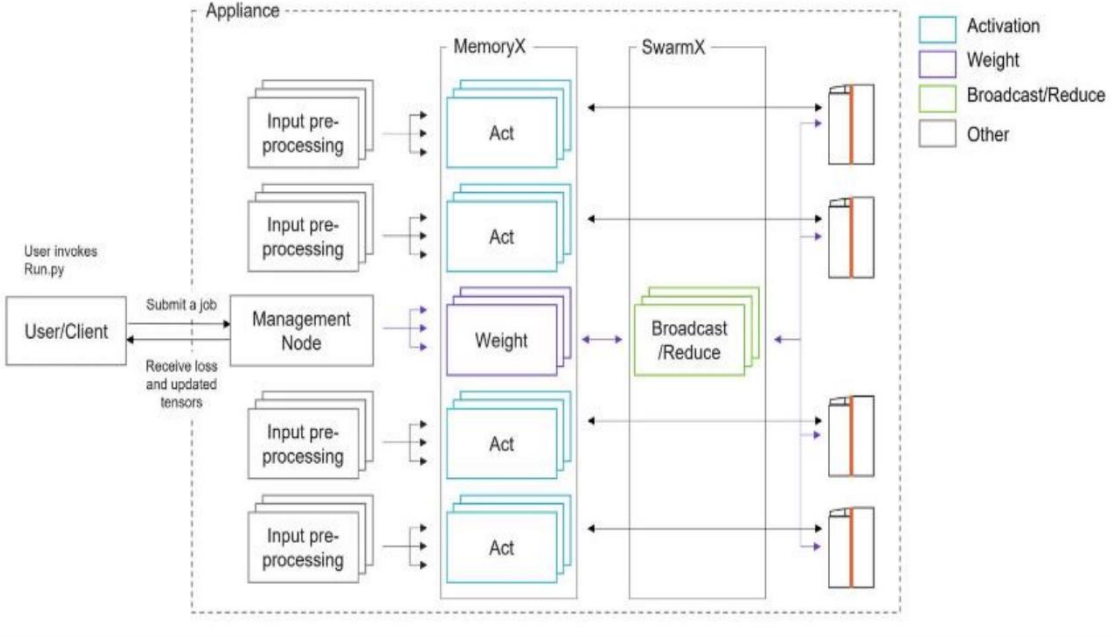
- 40GB on-chip SRAM
- Data and instructions
- 1 cycle read/write

Cerebras CS-2 Cluster

https://www.alcf.anl.gov/alcf-ai-testbed

ALCF's CS-2 Cluster

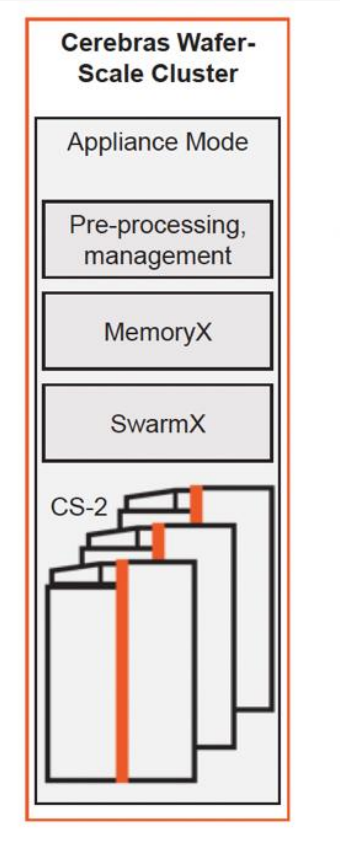
- 2 CS-2 Appliances (each chip 46225 mm^2)
- 1 Management node
- 16 Worker nodes
- 24 MemoryX nodes
- 6 SwarmX nodes
- 3 user login nodes



Topology of a Cerebras Wafer-Scale cluster

Training Giant Neural Networks Using Weight Streaming on Cerebras Wafer-Scale Clusters

WAFER-SCALE CLUSTER



Input preprocessing servers stream training data

MemoryX - Stores and streams model's weights

SwarmX – weight broadcasts and gradient across multiple CS2s

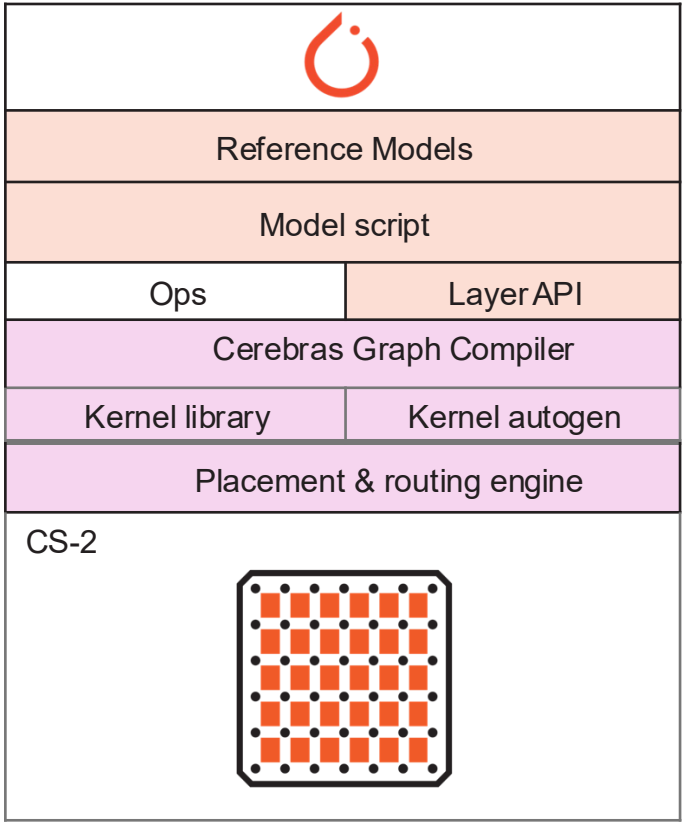
Compilation (maps graph to kernels) Execution (training)

Image Courtesy: Cerebras

LOWERING FROM MODEL TO WAFER

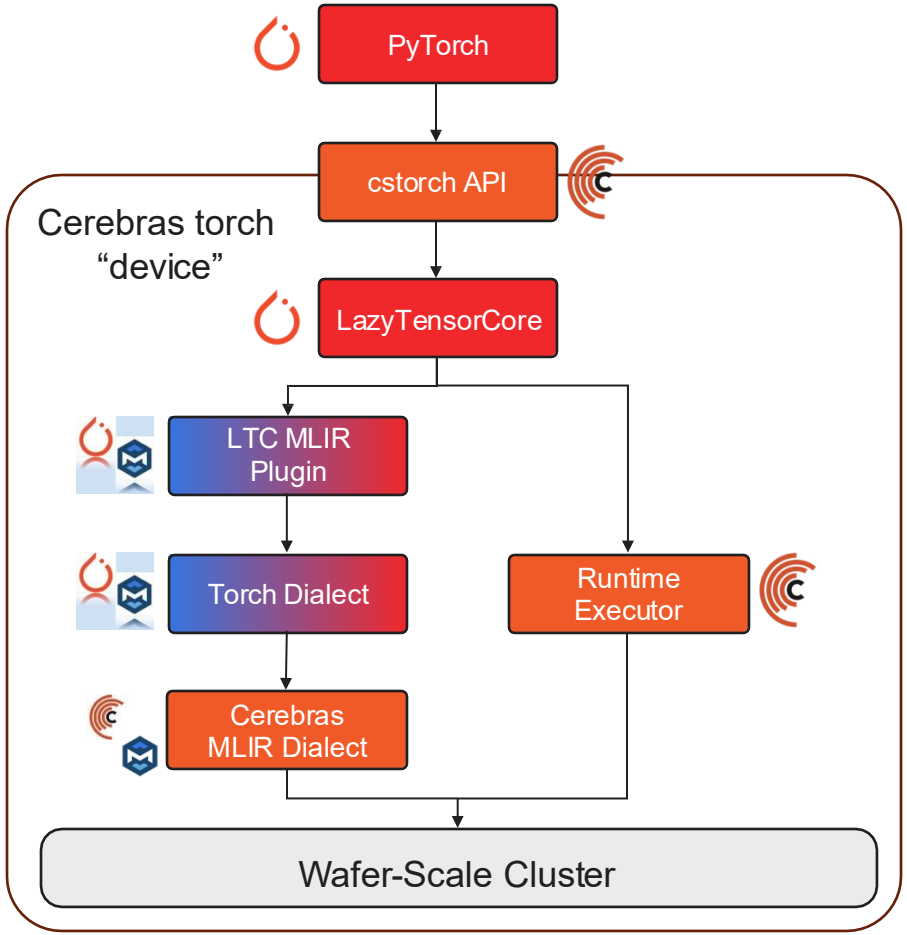
Integration with PyTorch

- Models defined in framework + Cerebras API
- Optimally maps from PyTorch to high performance kernels
  - Uses polyhedral code-generation or hand-written kernels
- Compiler using industry standard MLIR framework
  - Cerebras is an active contributor to the MLIR open- source community
- User does not worry about distributed compute or parallelism



CS TORCH

- cstorch API mirrors torch API
  - Helps with single device abstraction
- Tensor Ops traced through LazyTensorCore
  - Graph-by-execution with lazy evaluation
  - Also powers Google's xla/tpu device
- MLIR translation from LTC provided by torch-mlir
  - Hardware focused compiler ecosystem for torch
- Cerebras MLIR stack handles cluster optimizations
- Tensors get transferred to cluster as needed
  - Initial weights sent before first step
  - Inputs sent each step from custom data executor
- Execution driven asynchronously by cluster

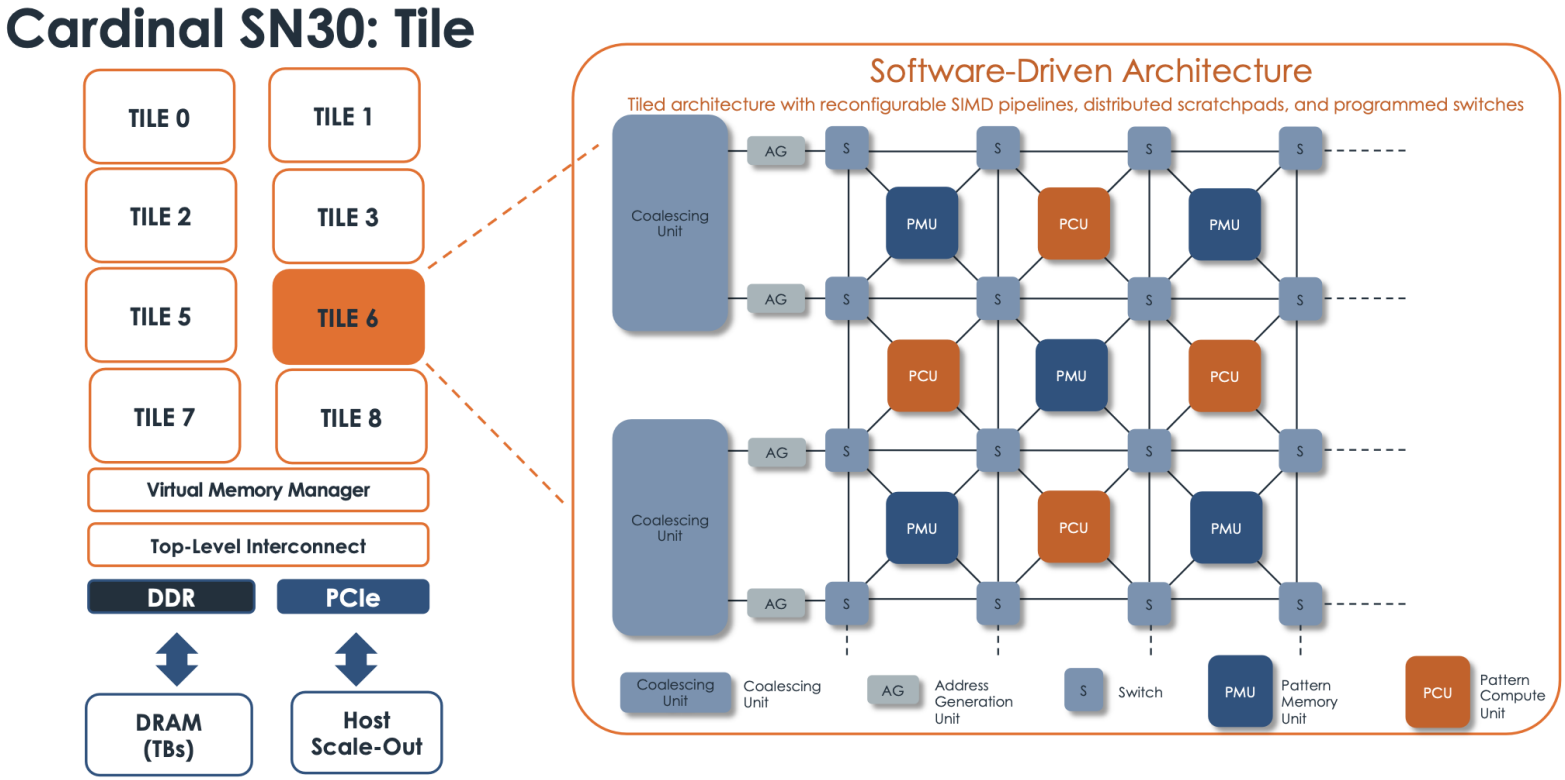


SambaNova™ SYSTEMS

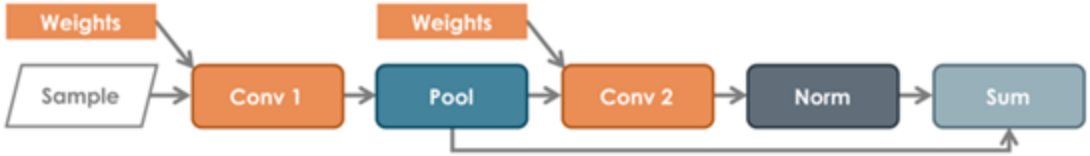
Cardinal SN30™
Reconfigurable Dataflow Unit™

- 7nm TSMC, 86B transistors
- 102 km of wire
- 640 MB on-chip, 1,024 GB external
- 688 TFLOPS (bf16)
- RDU-Connect™

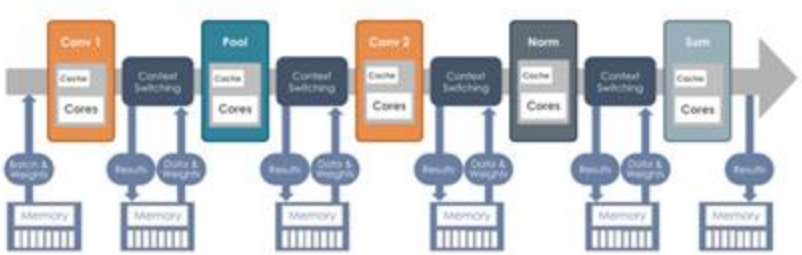
ABSTRACT VIEW



Dataflow Architectures

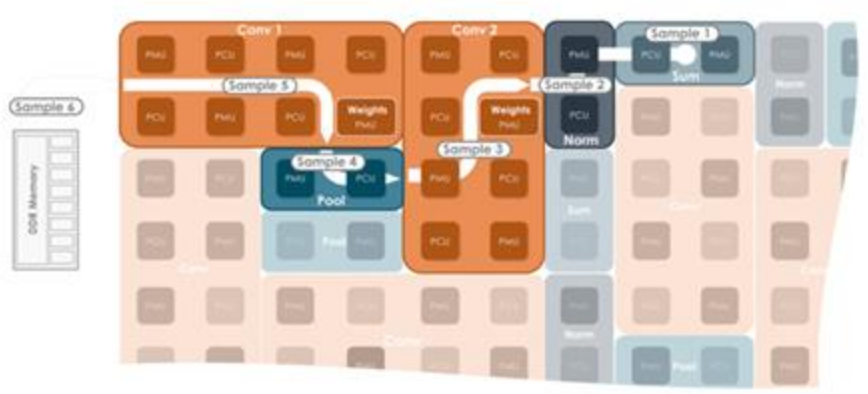


Simple Convolution Graph



The old way: kernel-by-kernel

Bottlenecked by memory bandwidth and host overhead



The Dataflow way: Spatial

Eliminates memory traffic and overhead

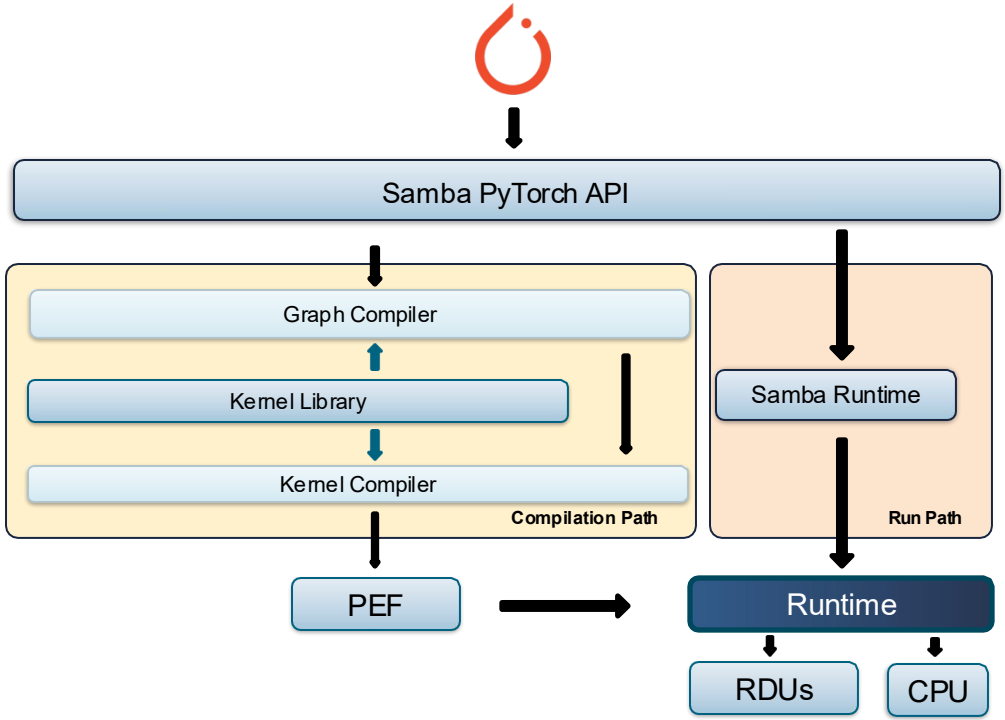
SAMBANOVA DATASCALE SN30-8 SYSTEM

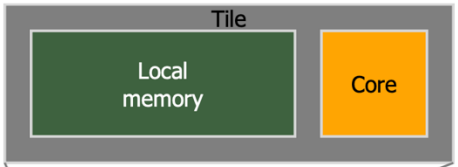
- 8 x Cardinal SN30 Reconfigurable Dataflow Unit
- 8 TB total memory (using 64 x 128 GB DDR4 DIMMs)
- 6 x 3.8 TB NVMe (22.8 TB total)
- PCIe Gen4 x16
- Host module

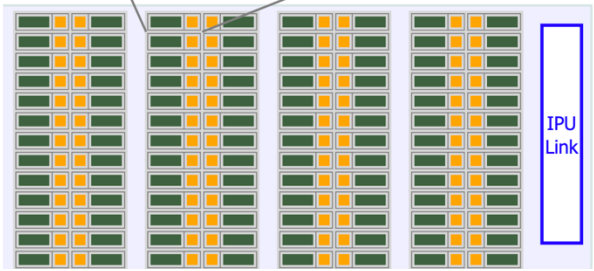
Image Courtesy: SambaNova

SAMBA COMPILATION FLOW

- **Samba**
  - SambaNova PyTorch compilation & run APIs
- **Graph compiler**
  - High-level ML graph transformation & optimizations
- **Kernel compiler**
  - Low-level RDU operator kernel transformation & optimizations
- **Kernel library**
  - RDU operator implementations







IPU-Tiles™

1472 independent IPU-Tiles™ each with an IPU-Core™ and In-Processor-Memory™

IPU-Core™

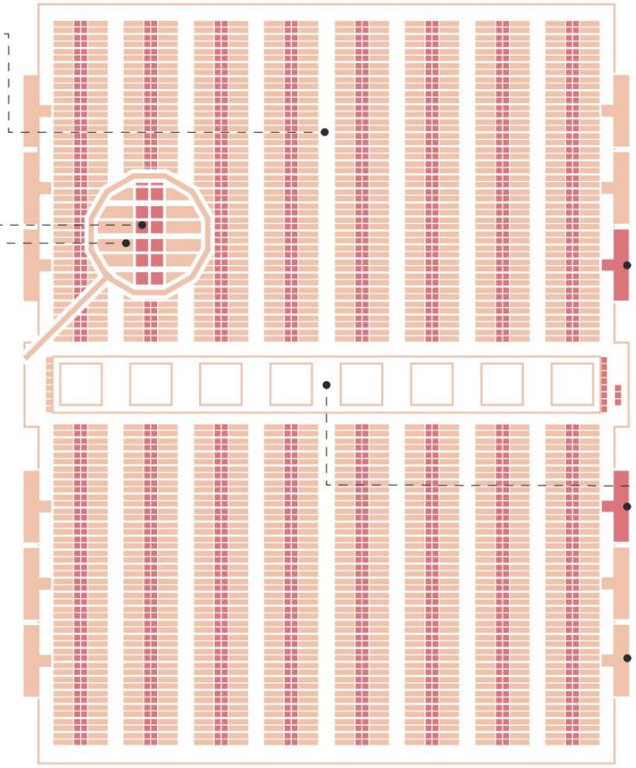
1472 independent IPU-Core™

8832 independent program threads executing in parallel

In-Processor-Memory™

900MB In-Processor-Memory™ per IPU

65TB/s memory bandwidth per IPU



IPU-Exchange™

11 TB/s all to all IPU-Exchange™
Non-blocking, any communication pattern

PCIe

PCI Gen4 x16
64 GB/s bidirectional bandwidth to host

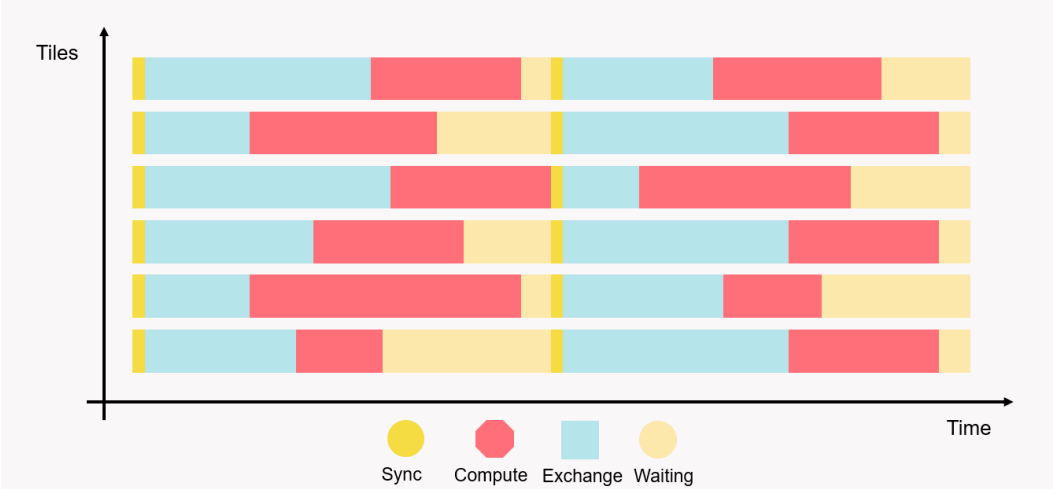
IPU-Links™

10 x IPU-Links,
320GB/s chip to chip bandwidth

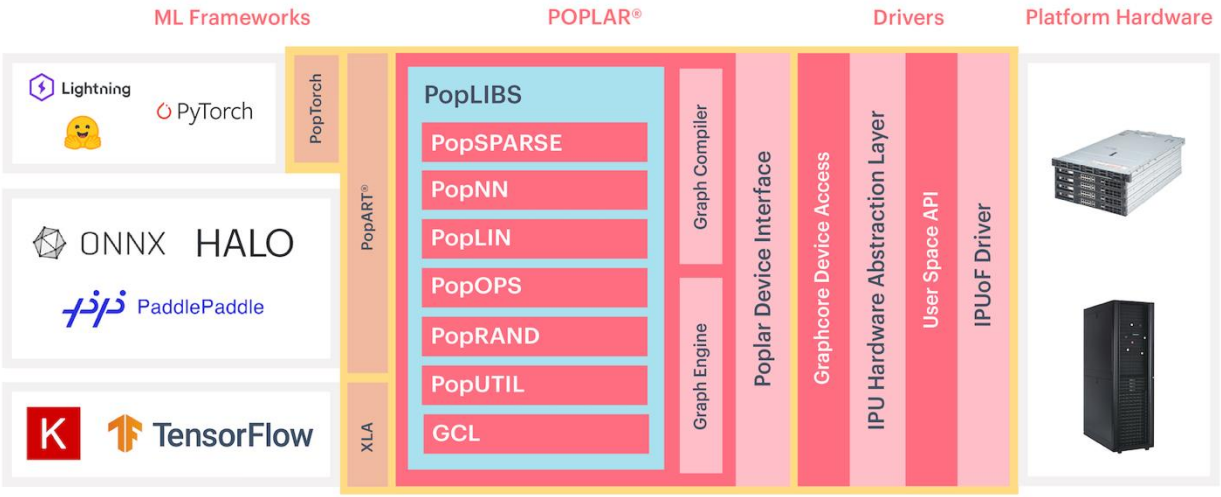
GRAPHCORE

BULK SYNCHRONOUS PARALLEL (BSP)

- The IPU uses the bulk-synchronous parallel (BSP) model of execution where the execution of a task is split into steps.
- Each step consists of the following phases:
  - local tile compute,
  - global cross-tile synchronization,
  - data exchange

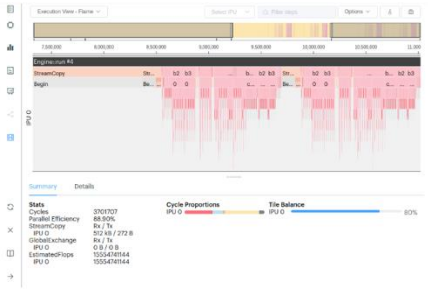


POPLAR SOFTWARE STACK



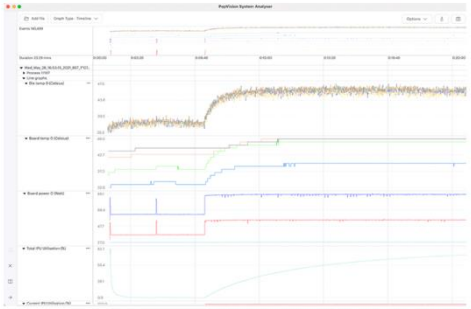
General purpose, extensible Parallel programming framework which is close to metal and targets the IPU

PROFILING: POPVISION TOOLS



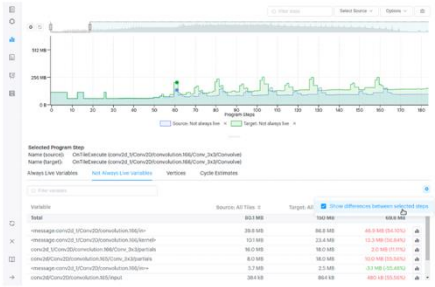
EXECUTION TRACE REPORT

View the output of instrumenting a Poplar program, capturing cycle counts for each step. See execution statistics, tile balance, cycle proportions and compute-set details.



GRAPH DATA

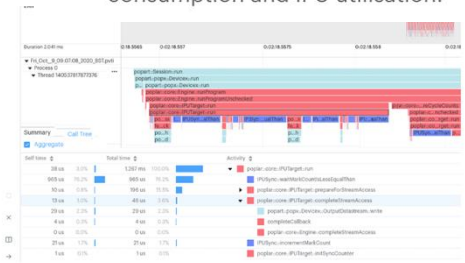
Plot graph data of any numerical data points from the host or IPU processor systems, such as board temperature, power consumption and IPU utilisation.



REPORT COMPARISONS

Open two reports at once to compare their memory, execution, liveness and operations. Visualise where efficiencies can be made with different model parameters.

PopVision Graph Analyzer



HOST EXECUTION ANALYSIS

Understand the execution of IPU-targeted software on your host system processors. Identify any bottlenecks between CPUs and IPUs across a visual interactive timeline.

IPU MEMORY ANALYSIS

Capture memory information from your ML models when executed on IPUs. Inspect variable placement, size and liveness throughout the execution.

PopVision System Analyzer

GROQ LPU OVERVIEW

SRAM Memory

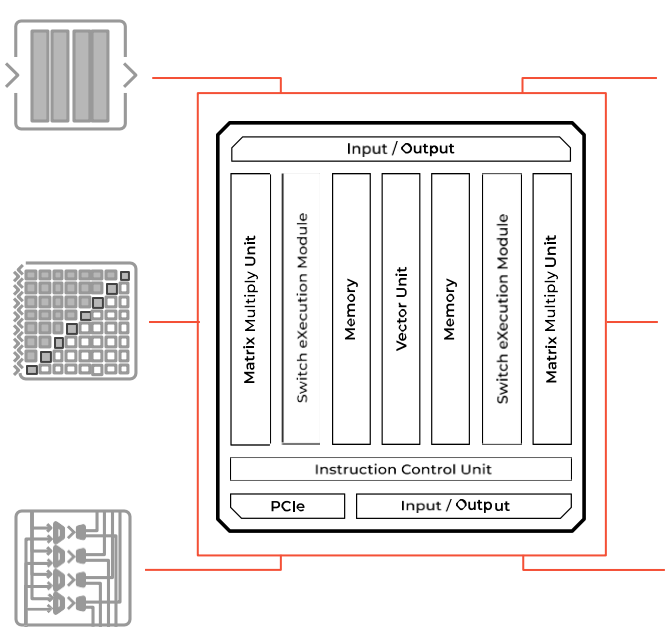
Massive concurrency
80 TB/s of BW
230MB capacity
Stride insensitive

Groq TruePoint™ Matrix

4x Engines
750 TOP/s int8
188 TFLOP/s fp16
320x320 fused dot product

Programmable Vector Units

5,120 Vector ALUs for high performance



Networking

480 GB/s bandwidth
Extensible network scalability
Multiple topologies

Data Switch

Shift, Transpose, Permuter for improved data movement and data reshapes

Instruction Control

Multiple instruction queues for instruction parallelism

GROQ LPU BUILDING BLOCKS

Build different types of specialized SIMD units

MXM
Matrix-Vector /
Matrix-Matrix Multiply

VXM
Vector-Vector
Operations

SXM
Data Reshapes

MEM
On-chip SRAM

ARCHITECTURE EMPOWERING SOFTWARE

Software-controlled memory

No dynamic hardware caching

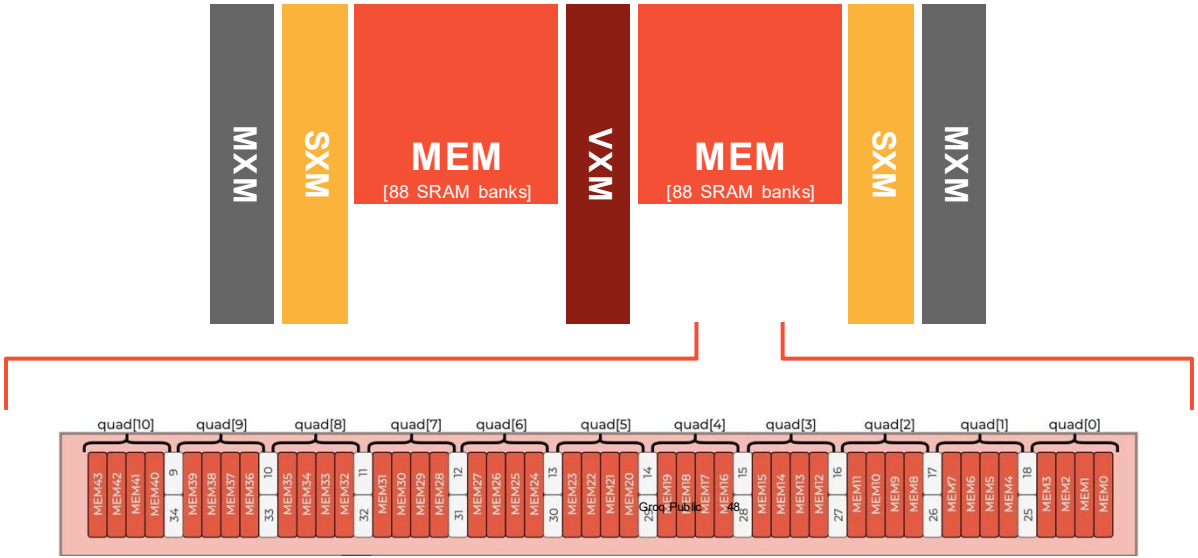
- Compiler aware of all data locations at any given point in time

Flat memory hierarchy
(no L1, L2, L3, etc)

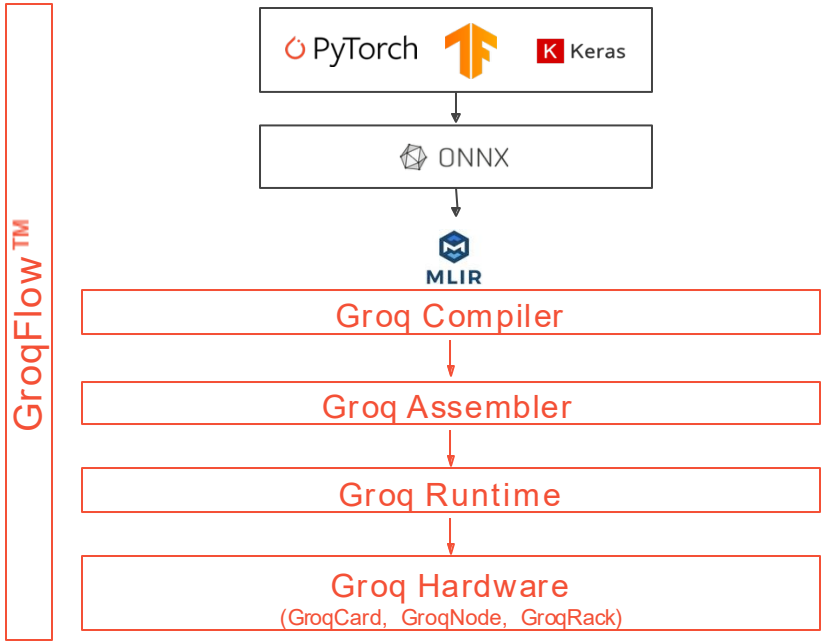
- Memory exposed to software as a set of physical banks that are directly addressed

Large on-chip memory capacity (220 MiB) at very high-bandwidth (80 TBps)

- Achieves high compute efficiency even at low operational intensity



GROQWARE™ SUITE



DIVERSE SUITE OF DEVELOPMENT TOOLS

Out-of-Box

Groq Compiler provides out-of-box support for standard Deep Learning models

+

Productivity Tools

GroqView Profiler provides visualization of the chip's compute and memory usage at compile time

GroqFlow Tool Chain enables a single line of Pytorch or TensorFlow code to import and transform models through a fully automated tool chain to run on Groq hardware

GENERAL GROQ LLM DEVELOPMENT FLOW

1. Modify PyTorch Model
2. Export ONNX Model
3. Convert ONNX Model from fp32 to fp8/fp16
4. Decoder Partition
5. Groq Compile!
6. Multi-node/Multi-rack Host-Code Invocation

	Cerebras CS2	SambaNova Cardinal SN30	Groq GroqRack	GraphCore GC200 IPU	Habana Gaudi1	NVIDIA A100
Compute Units	850,000 Cores	640 PCUs	5120 vector ALUs	1472 IPUs	8 TPC + GEMM engine	6912 Cuda Cores
On-Chip Memory	40 GB L1, 1TB+ MemoryX	>300MB L1 1TB	230MB L1	900MB L1	24 MB L1 32GB	192KB L1 40MB L2 40-80GB
Process	7nm	7nm	7 nm	7nm	7nm	7nm
System Size	2 Nodes including Memory-X and Swarm-X	8 nodes (8 cards per node)	9 nodes (8 cards per node)	4 nodes (16 cards per node)	2 nodes (8 cards per node)	Several systems
Estimated Performance of a card (TFlops)	>5780 (FP16)	>660 (BF16)	>250 (FP16) >1000 (INT8)	>250 (FP16)	>150 (FP16)	312 (FP16), 156 (FP32)
Software Stack Support	Tensorflow, Pytorch	SambaFlow, Pytorch	GroqAPI, ONNX	Tensorflow, Pytorch, PopArt	Synapse AI, TensorFlow and PyTorch	Tensorflow, Pytorch, etc
Interconnect	Ethernet-based	Ethernet-based	RealScale ™	IPU Link	Ethernet-based	NVLink

Outline

- Motivation
- Overview of AI Accelerators
- Benchmarking Results
- Accelerators for HPC
- Concluding Remarks

APPROACH

Perform a comprehensive evaluation of Generative AI models

LLM Workloads

- ***Micro-Benchmarks*:** Transformers forward and backward
- ***Benchmarks*:** GPT2-XL – Throughput, Scaling, GAS, Sequence Length
- ***AI-driven Science applications*:** GenSLM

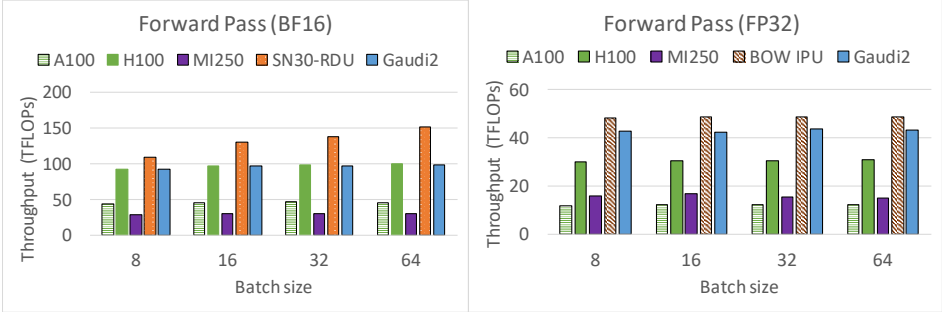
Architectures:

- **GPUs:** Nvidia A100, AMD MI250
- **AI Accelerators:** Cerebras CS-2, Samabanova Datascale SN30, Intel Habana Gaudi 2, Grapchore Pod64, Groq TSP

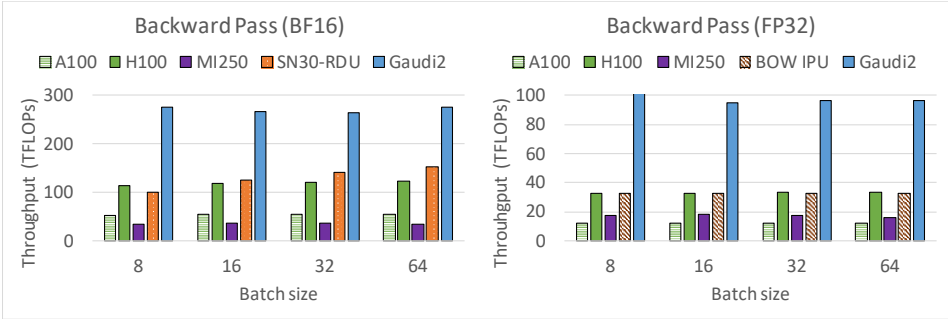
Details: Toward a holistic performance evaluation of large language models across diverse ai accelerators

MICRO-BENCHMARKS: TRANSFORMER

- The throughput with FP16 and BF16 precision is ~4x higher than FP32
- Overall, the H100 demonstrated 2x better throughput than the A100 GPU
- SambaNova exhibits impressive performance with BF16 precision
- Intel Gaudi 2 exhibits highest performance with all three formats Spelling for backward pass



Transformer Layer: Forward Pass

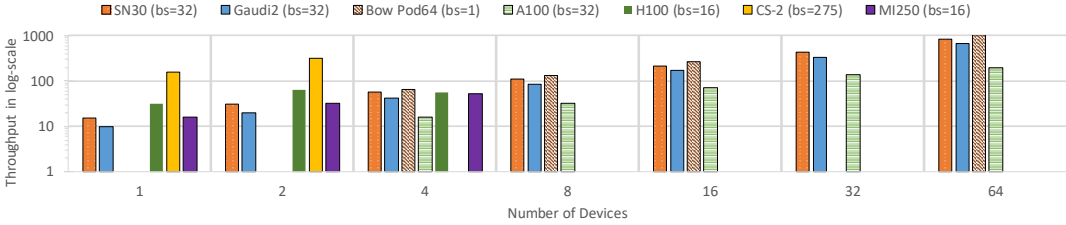


Transformer Layer: Backward Pass

BENCHMARK: GPT2-XL

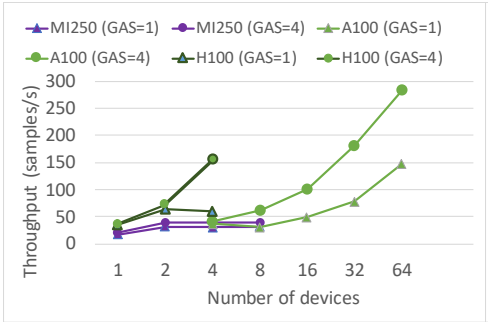
Pre-trained GPT-2 XL 1.5B parameter model on OWT dataset

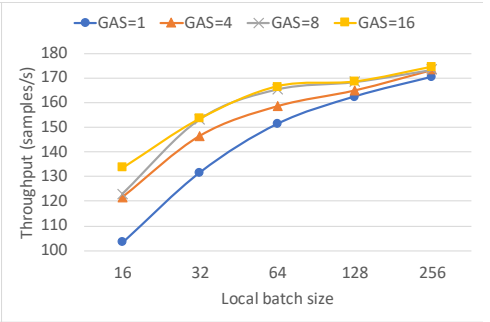
- same sequence length, tuned batch sizes
- 2 CS-2s outperformed the runs on 64 A100s
- Near-linear scaling behavior



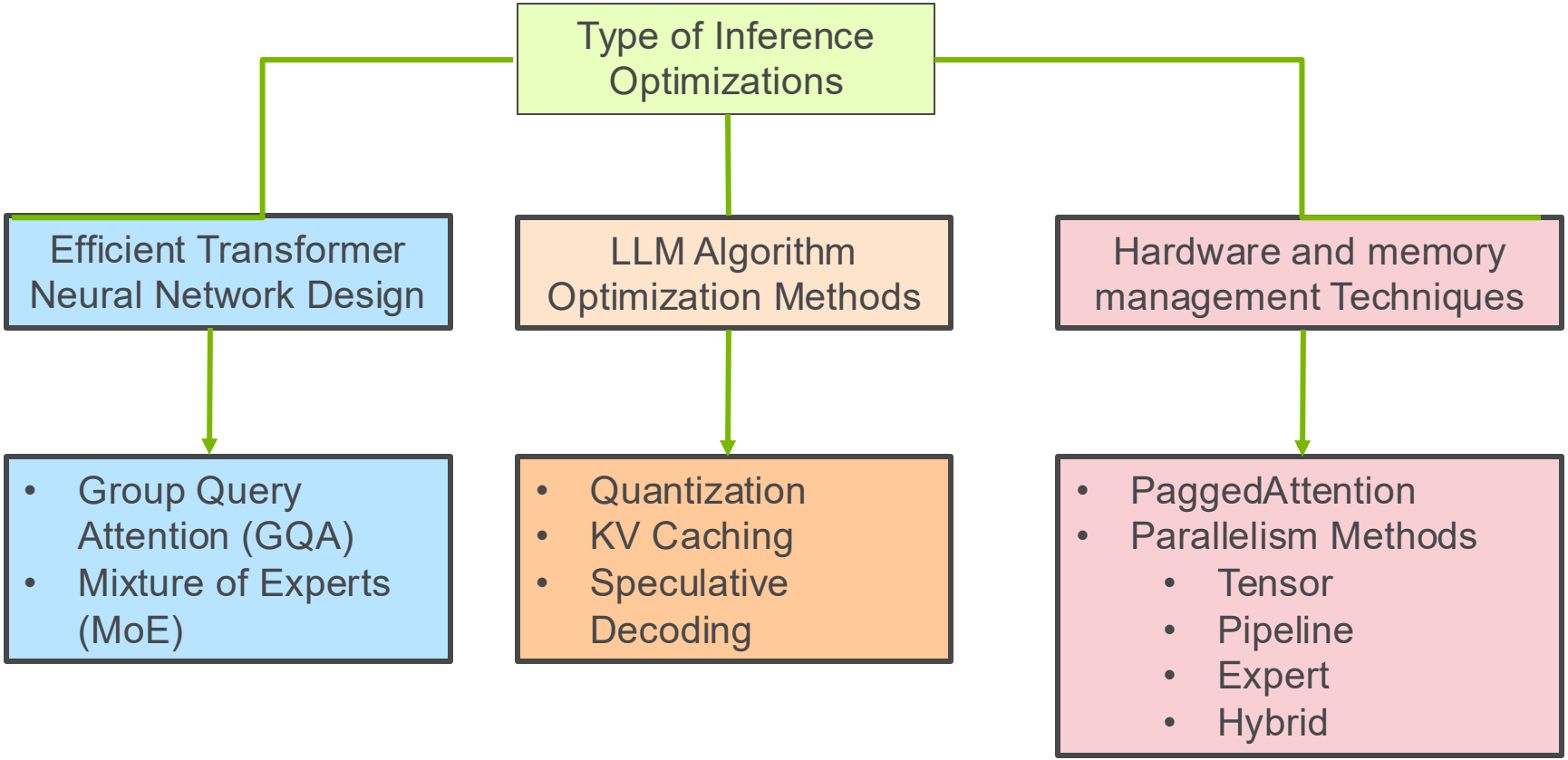
System	min #devices	max #devices	scale #devices	scaling efficiency	Speedup
Gaudi2	1	64	64	104%	66.4x
Bow Pod64	4	64	16	100.1%	16x
CS-2	1	2	2	99.87%	1.99x
SN30	1	64	64	97.5%	62.4x
MI250	1	4	4	80%	3.2x
A100	4	64	16	75.8%	12.1x
H100	1	4	4	43%	1.73x

- Throughput improvement for larger GAS values before it saturates as the number of devices increase
- Increasing the GAS values has less or no effect on the throughput, which saturates at larger local batch sizes
- Graphcore can support very large GAS values ranging from 16 to 2048





LLM INFERENCE OPTIMIZATION TECHNIQUES

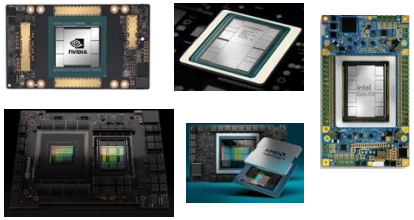


LLM-INFERENCE-BENCH
BRIDGING LLMS, ACCELERATORS AND FRAMEWORKS

Open source LLMs
LLaMA, Mistral, Qwen

LLM-Inference-Bench

AI Accelerators
Nvidia, AMD GPUs, SambaNova SN40L, Habana Gaudi



vLLM

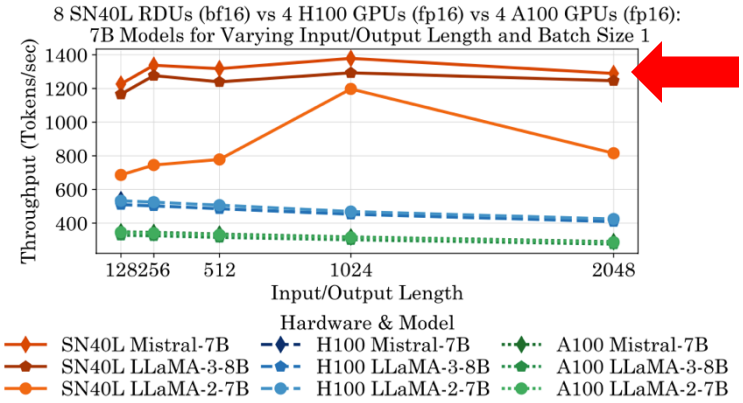
NVIDIA TensorRT-LLM

DeepSpeed MII

LLaMA C++

Details: LLM-Inference-Bench: Inference Benchmarking of Large Language Models on AI Accelerators
Interactive dashboard: https://github.com/argonne-lcf/LLM-Inference-Bench

MODEL COMPARISON

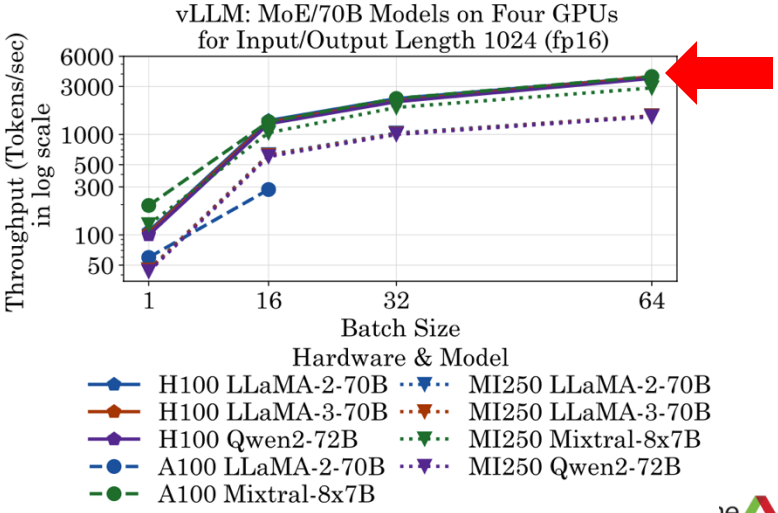


7B Models

- Mistral-7B performs better than LLaMA-3-8B due to one billion less parameters
- LLaMA-3-8B performs better than LLaMA-2-7B due to Group Query Attention (GQA) despite one billion more parameters

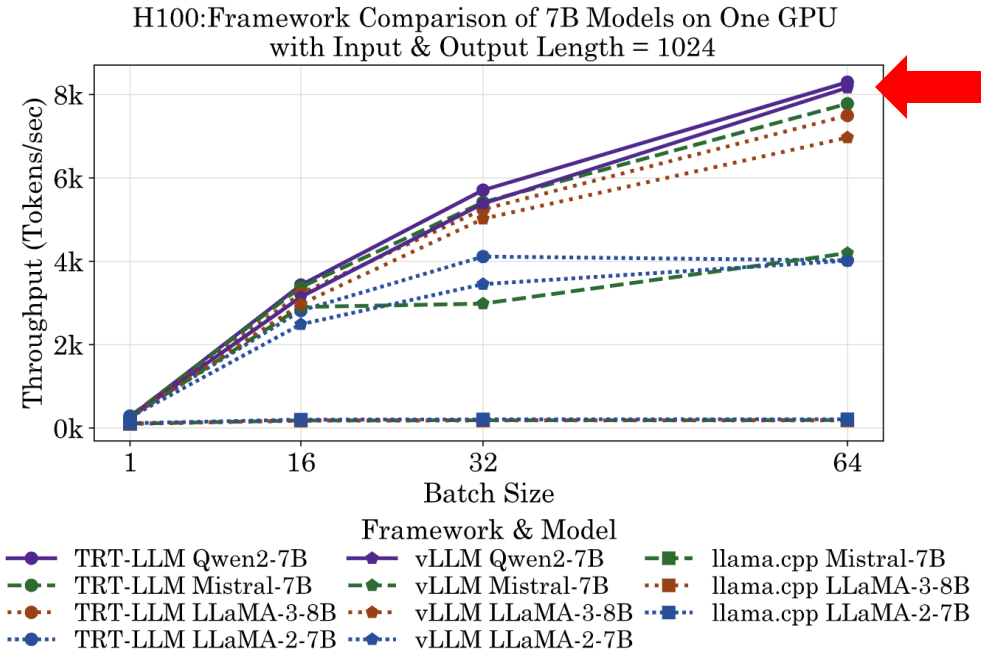
Large Models

- Mixtral-8x7B performs better than LLaMA-3-70B and LLaMA-2-70B due to mixture of experts layer utilizing less active parameters
- LLaMA-2-70B performs better than LLaMA-3-70B due to smaller vocabulary size



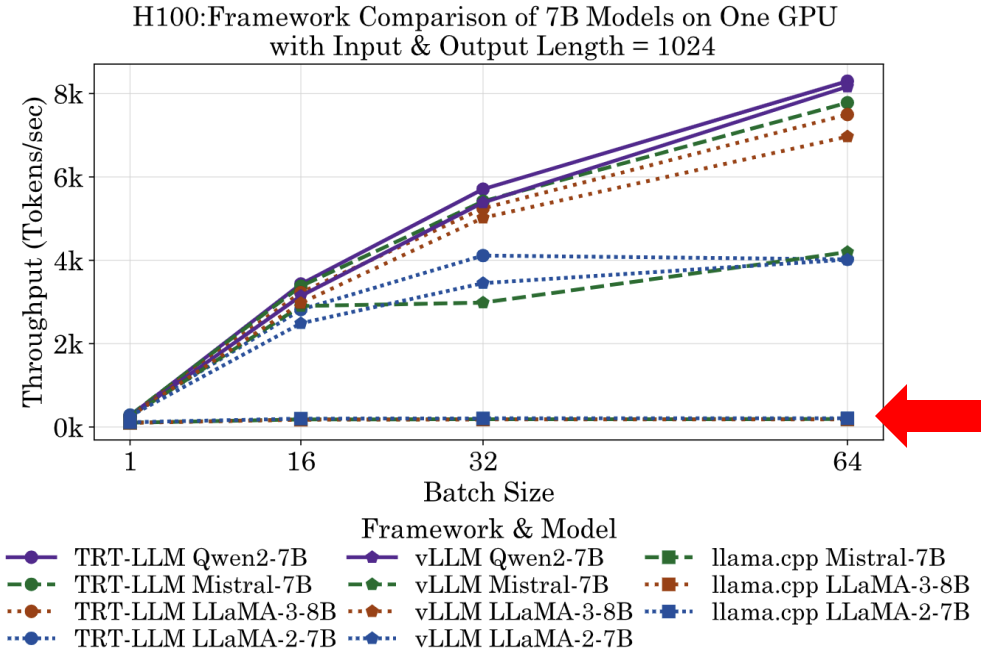
FRAMEWORK COMPARISON

- TensorRT-LLM attains the highest throughput on Nvidia GPUs across different Large Language Models
- vLLM is the second best performer
- llama.cpp shows least performance due to lack of efficient transformer algorithm methods such as GQA and PaggedAttention



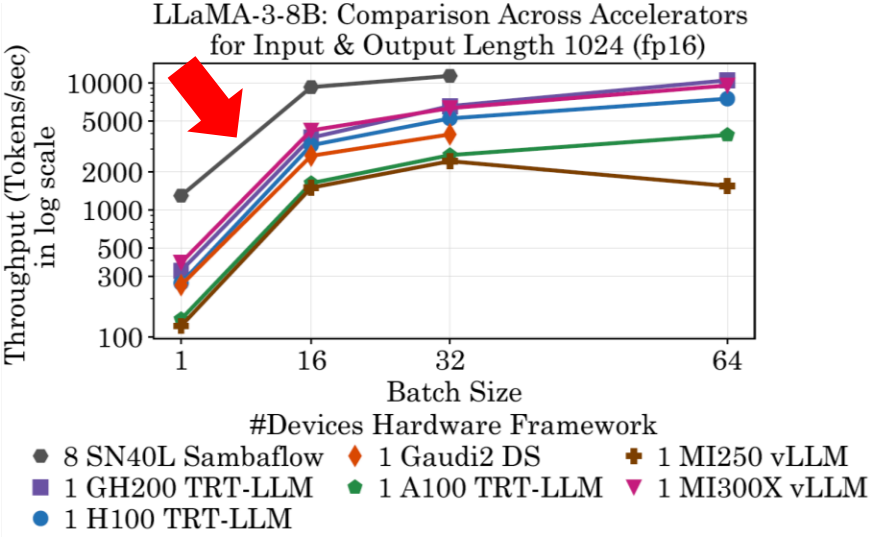
FRAMEWORK COMPARISON

- TensorRT-LLM attains the highest throughput on Nvidia GPUs across different Large Language Models
- vLLM is the second best performer
- llama.cpp shows least performance due to lack of efficient transformer algorithm methods such as GQA and PaggedAttention



ACCELERATOR COMPARISON

- SambaNova SN40L achieves has the best performance among all the accelerators we benchmarked
  - However, as of July 2024, the maximum batch size SN40L supports is 32
- Nvidia GH200 > H100 > A100 (in terms of throughput)
- MI300X and GH200 are comparable
- Habana Gaudi's performance is between A100 and H100
- The performance of AMD MI250 saturates for large batch sizes



Outline

- Motivation
- Overview of AI Accelerators
- Benchmarking Results
- Accelerators for HPC
- Concluding Remarks

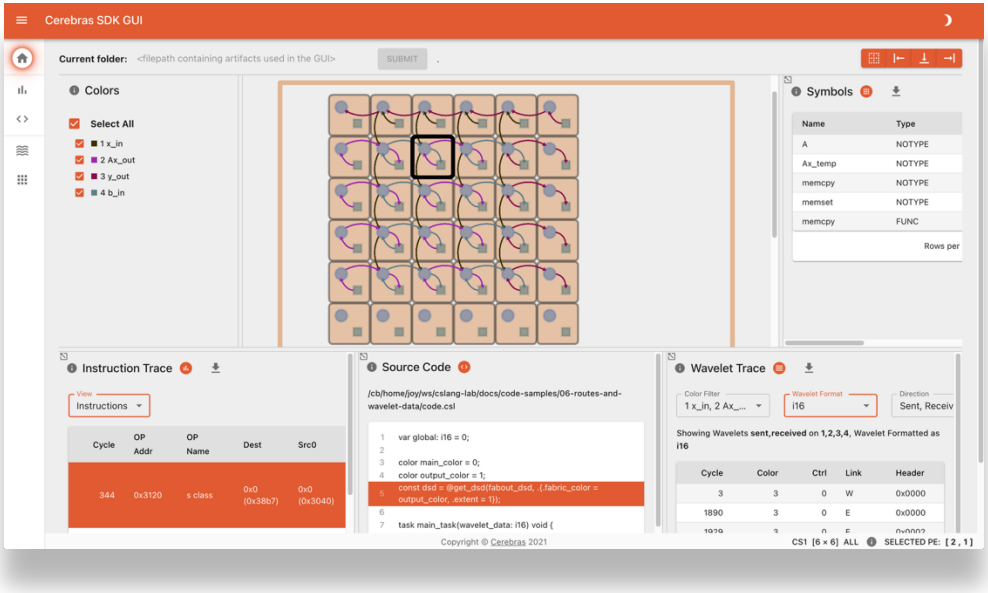
cerebras

- **850,000** cores optimized for sparse linear algebra
- **46,225 mm²** silicon
- **2.6 trillion** transistors, **7nm** process technology
- **40 gigabytes** of on-chip memory
- **20 PByte/s** memory bandwidth **220 Pbit/s** fabric bandwidth

CEREBRAS SDK

A general-purpose parallel-computing platform and API allowing software developers to write custom programs (“kernels”) for Cerebras systems.

Language	CSL: Cerebras Software Language	
	Host APIs with Python	
Libraries	Optimized primitives	
Tools	Simulator	Debugger
		Visualization



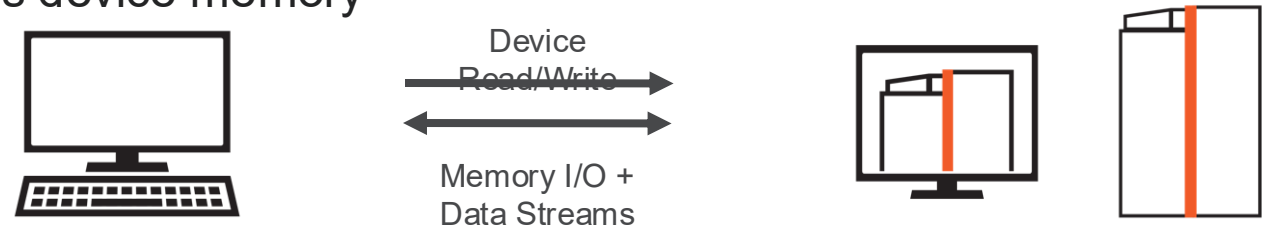
FROM A PROGRAMMER'S PERSPECTIVE

Host CPU(s): Python

- Loads program onto simulator or CS-2 system
- Streams in/out data from one or more workers
- Reads/writes device memory

Device: CSL

- Target software simulator or CS-2
- CSL programs run on groups of cores on the WSE, specified by programmer
- Executes dataflow programs



CSL: LANGUAGE BASICS

- Types
- Functions
- Control structures
- Structs/Unions/Enums
- Comptime

Straight from C
(via Zig)

- Builtins
- Module system
- Params
- Tasks
- Data Structure Descriptors
- Layout specification

CSL specific

Used for writing device kernel code

Familiar to C/C++/HPC programmers

SDK USAGE AND IMPACT

Over the past year, SDK has evolved from a closed tool requiring NDA access to a public platform for Wafer-Scale Computing. We're supporting more research and publications than ever.

Scaling the "Memory Wall" for Multi-Dimensional Seismic Processing with Algebraic Compression on Cerebras CS-2 Systems

Hatem Ltaief, Yuxi Hong — Extreme Computing Research Center
Leighton Wilson, Mathias Jacquelin — Cerebras Systems Inc.
Matteo Ravasi, David Keyes — Extreme Computing Research Center

Using Wafer-Scale AI Hardware for Traditional HPC Simulation Workloads: A Case Study in Developing a Monte Carlo Particle Transport Application for the Cerebras WSE2 AI Accelerator

Kazutomo Yoshii* Andrew Siegel* Leighton Wilson‡

portance to both fission and fusion reactor simulation fields, and because the MC algorithm has historically failed to achieve more than a few percent of theoretical peak FLOP performance due to its inherently stochastic

Near-Optimal Wafer-Scale Reduce

Piotr Luczynski — Department of Computer Science, ETH Zurich
Lukas Gianinazzi — Department of Computer Science, ETH Zurich
Patrick Iff — Department of Computer Science, ETH Zurich
Leighton Wilson
Daniele De Sensi — Sapienza University of Rome
Torsten Hoefler — Department of Computer Science, ETH Zurich

and various other HPC applications [35, 38, 51, 58]. However, maximizing performance on this architecture necessitates tailoring communication patterns to its unique characteristics. This need motivates our investigation of Reduce and AllReduce on the WSE.

1.2 Limitations of state-of-the-art

Current wafer-scale Reduce and AllReduce implementations are

ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich — SPCL

Communication Collectives for the Cerebras Wafer-Scale Engine

Bachelor Thesis

Piotr Luczynski
pluczynski@ethz.ch

DEPARTMENT OF INFORMATICS
TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Thesis in Informatics

Implementation and Evaluation of Matrix Profile Algorithms on the Cerebras Wafer-Scale Engine

Vyas Giridharan

Massively Distributed Finite-Volume Flux Computation

Ryuichi Sai* — TotalEnergies EP Research & Technology US, LLC. Houston, Texas, USA — ryuichi@rice.edu
Mathias Jacquelin — Cerebras Systems, Sunnyvale, California, USA
François P. Hamon — TotalEnergies EP Research & Technology US, LLC. Houston, Texas, USA
Mauricio Araya-Polo — TotalEnergies EP Research & Technology US, LLC. Houston, Texas, USA
Randolph R. Settgast — Lawrence Livermore National Laboratory, Livermore, California, USA

Monte Carlo with Single-Cycle Latency: Optimization of a Continuous Energy Cross Section Lookup Kernel for AI Accelerator Hardware

John Tramm [1,*], Bryce Allen[1,2], Kazutomo Yoshii[1], Andrew Siegel[1]

[1] Argonne National Laboratory [2] University of Chicago, Chicago, IL

CereSZ: Enabling and Scaling Error-bounded Lossy Com

Matrix-Free Finite Volume Kernels on a Dataflow Architecture

Ryuichi Sai*, François P. Hamon†, John Mellor-Crummey*, Mauricio Araya-Polo†
*Rice University, Houston, TX, USA
†TotalEnergies EP Research & Technology US, LLC., Houston, TX, USA

***Abstract*—Fast and accurate numerical simulations are crucial for designing large-scale geological carbon storage projects ensuring safe long-term CO_2 containment as a climate change mitigation strategy. These simulations involve solving numerous large and complex linear systems arising from the implicit Finite Volume (FV) discretization of PDEs governing subsurface fluid flow. Compounded with highly detailed geomodels, solving linear systems is computationally and memory expensive, and accounts for the majority of the simulation time. Modern memory hierarchies are insufficient to meet the latency and bandwidth needs of large-scale numerical simulations. Therefore, exploring algorithms that can leverage alternative and balanced paradigms, such as dataflow and in-memory computing is crucial. This**

Advancements in HPC system design have enabled optimizations and algorithmic changes in scientific and business fields. Previous investigations into non-hierarchical architectures have improved computational efficiency [3], [4]. The emergence of highly parallel systems with distributed memory architecture is now considered as alternative to traditional accelerated systems. New chips from Cerebras [5], Groq [6], and SambaNova [7], systems with dataflow-like architecture design and on-chip memory possess higher memory bandwidth, lower memory latency, and lower energy cost for memory access.

In this article, we explore the capabilities of a dataflow

6 Aug 2024

Trackable Agent-based Evolution Models at Wafer Sc

Matthew Andres Moreno[1,2,3,*], Connor Yang[4], Emily Dolson[5,6], and Lui
[1]Department of Ecology and Evolutionary Biology, University of Michigan, Ann Arbor, U
[2]Center for the Study of Complex Systems, University of Michigan, Ann Arbor, Unite
[3]Michigan Institute for Data Science, University of Michigan, Ann Arbor, United S
[4]Undergraduate Research Opportunities Program, University of Michigan, Ann Arbor, U
[5]Department of Computer Science and Engineering, Michigan State University, East Lansing
[6]Program in Ecology, Evolution, and Behavior, Michigan State University, East Lansing,
*corresponding author: morenoma@umich.edu

Abstract

Continuing improvements in computing hardware are poised to transform capabilities for *in silico* modeling of cross-scale phenomena underlying major open questions in evolutionary biology and artificial life, such as transitions in individuality, eco-evolutionary dynamics, and rare evolutionary events. Emerging ML/AI-oriented

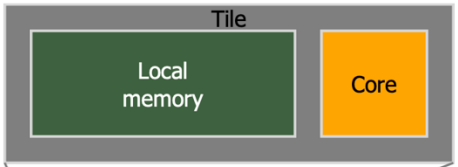
Simulation Runtime
#strat markers

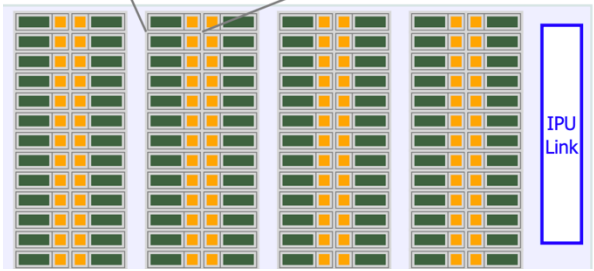
Near-optimal Reduce on the Cerebras Wafer Scale Engine — SPCL

ultiplication on Cerebras WSE-2: Evaluating Algorithms in Spatial Computing

Filip Dobrosavljević
dofilip@student.ethz.ch
ETH Zurich
Switzerland

Torsten Hoefler
torsten.hoefler@inf.ethz.ch
ETH Zurich
Switzerland





IPU-Tiles™

1472 independent IPU-Tiles™ each with an IPU-Core™ and In-Processor-Memory™

IPU-Core™

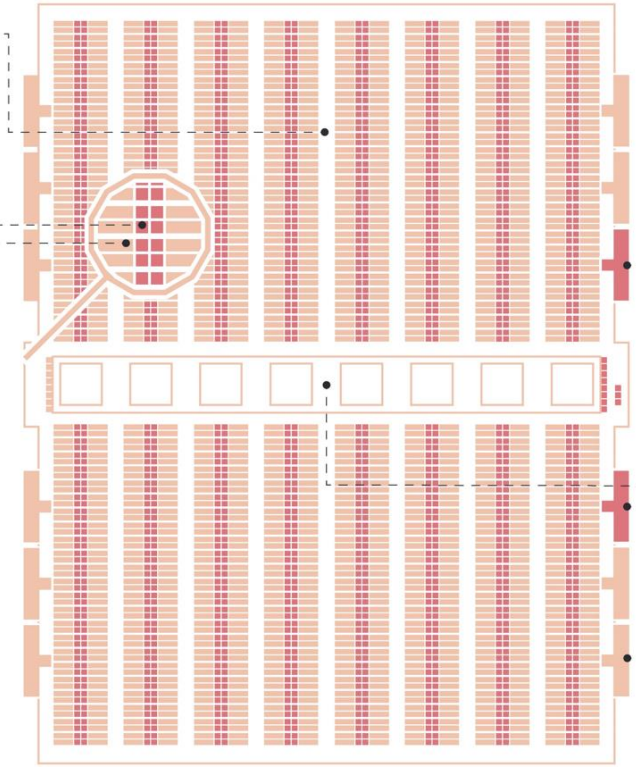
1472 independent IPU-Core™

8832 independent program threads executing in parallel

In-Processor-Memory™

900MB In-Processor-Memory™ per IPU

65TB/s memory bandwidth per IPU



IPU-Exchange™

11 TB/s all to all IPU-Exchange™
Non-blocking, any communication pattern

PCIe

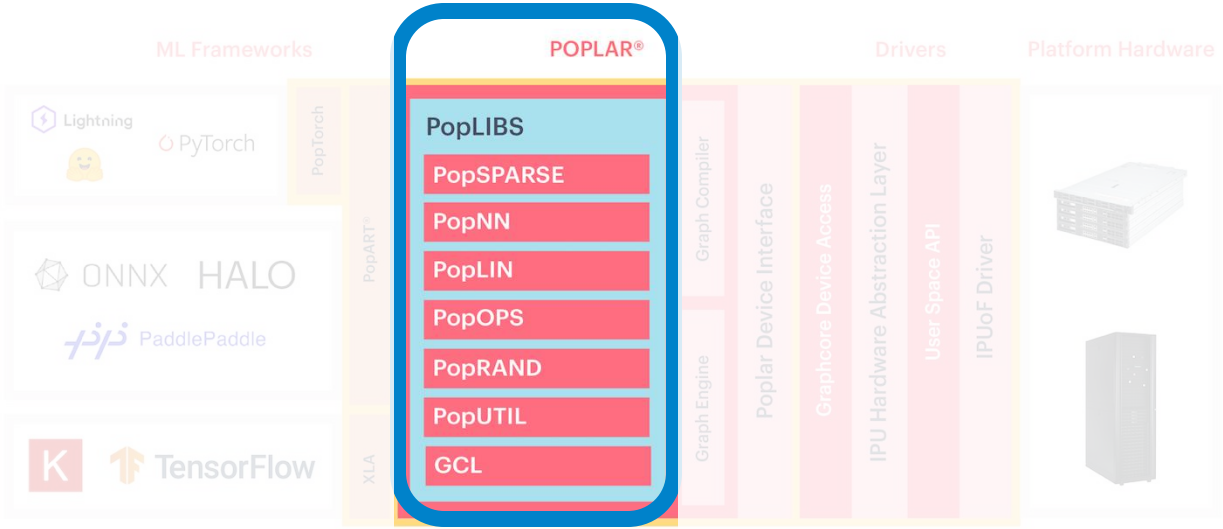
PCI Gen4 x16
64 GB/s bidirectional bandwidth to host

IPU-Links™

10 x IPU-Links,
320GB/s chip to chip bandwidth

GRAPHCORE

GRAPHCORE POPLAR SOFTWARE STACK



General purpose, extensible Parallel programming framework which is close to metal and targets the IPU

GRAPHCORE HOST PROGRAM

Host programs use the poplar library.

The Graph class is used to build up the computation graph.

The Engine class represents a fully compiled program ready to run on hardware.

```
#include <poplar/Engine.hpp>

using namespace poplar;
using namespace poplar::program;

...

Graph graph(target);
graph.addCodelets("my-codelets.cpp");

Program prog1, prog2;

constructMyGraph(graph, &prog1, &prog2);

Engine eng(device, graph, {prog1, prog2});
...

eng.run(0);
```

Codelets are loaded into the graph.

Control programs are built up out of instances of the Program class.

CODELET DEFINITIONS

Each codelet is defined as a C++ class that inherits from the Vertex class.

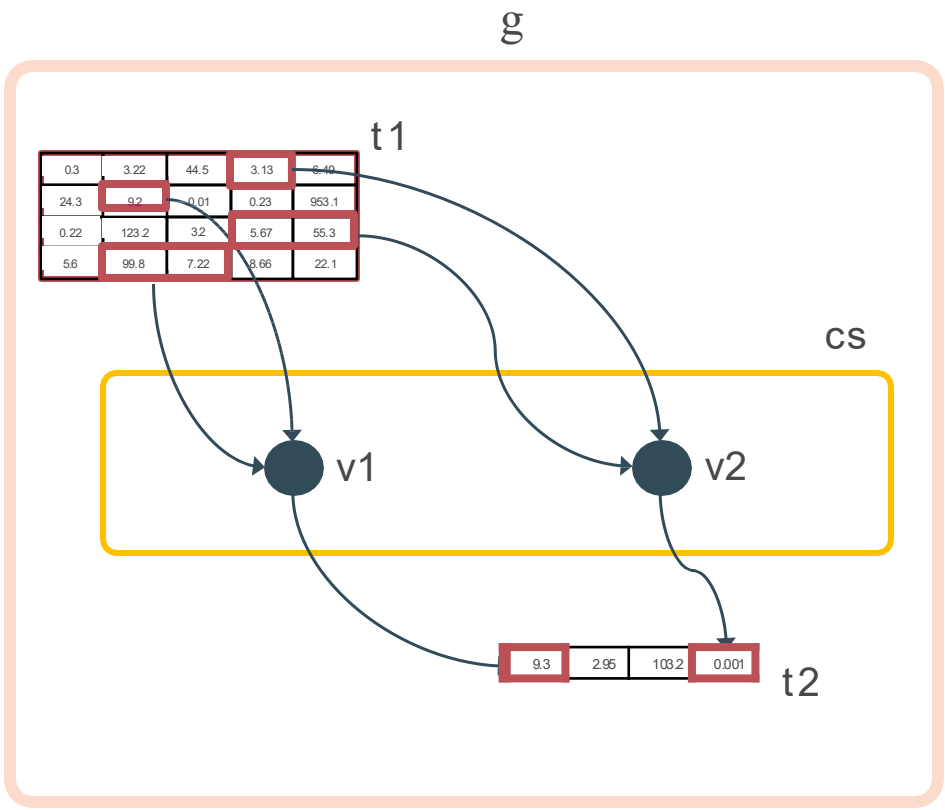
The fields of the vertex specify its inputs, outputs and internal data.

```
class AdderVertex : public Vertex {
public:
 Input<float> x;
 Input<float> y;
 Output<float> z;
 float bias;

 bool compute() {
   *z = x + y + bias;
   return true;
 }
}
```

The compute method specifies the vertex execution behaviour.

BUILDING THE COMPUTE GRAPH

```
Graph g(device);
g.addCodelets("codelets.cpp");

Tensor t1 = g.addVariable(FLOAT, {4, 5});
Tensor t2 = g.addVariable(FLOAT, {4});

ComputeSet cs = g.addComputeSet("myComputeSet")

VertexRef v1 = g.addVertex(cs, "AdderVertex");
VertexRef v2 = g.addVertex(cs, "AdderVertex");

g.connect(t1[1][1], v1["x"]);
g.connect(t1.slice({3, 1}, {4, 3}), v1["y"]);

g.connect(t2[0], v1["z"]);

g.connect(t1[0][3], v2["x"]);
g.connect(t1.slice({2, 2}, {3, 4}), v2["y"]);
g.connect(t2[3], v2["z"]);

g.setTileMapping(t1.slice({0, 0}, {4, 2}), 0);
g.setTileMapping(t1.slice({0, 2}, {4, 5}), 1);
g.setTileMapping(t2, 2);

g.setTileMapping(v1, 0);
g.setTileMapping(v2, 1);
```



AI ACCELERATORS FOR HPC

Benefits

- Significant Performance benefits over CPUs and GPUs.
- High memory bandwidth yields to high compute performance
- Programming models from ground up allow description of programs in truly parallel and scalable manner

Challenges

- Under development software stack and constantly evolving software stack
- Low level programming gives more flexibility at cost of higher learning curve
- Significant compilation and projection times.

Outline

- Motivation
- Overview of AI Accelerators
- Benchmarking Results
- Accelerators for HPC
- Concluding Remarks

Getting Started on ALCF AI Testbed

Available for Allocations

- Cerebras CS-2,
- SambaNova Datascale SN30,
- GroqRack
- Graphcore Bow Pod64

AI Testbed User Guide

Director's Discretionary (DD) awards

- Scaling code
- Preparing for future computing competition
- Scientific computing in support of strategic partnerships.

Allocation Request Form

https://www.alcf.anl.gov/science/directors-discretionary-allocation-program

NAIRR Pilot

aims to connect U.S. researchers and educators to computational, data, and training resources needed to advance AI research and research that employs AI.

https://nairrpilot.org/

THANK YOU

Contact

Linkedin: Sid Raskar
Email: sraskar@udel.edu