The Case for Domain-Specific Networks

Dennis Abts
(NVIDIA)

John Kim
(KAIST)

What does domain specialization mean?

General Purpose

Domain-specific (capacity, reliability, and efficiency)

Domain-specific (performance, maneuverability, offensive and defensive weapons)

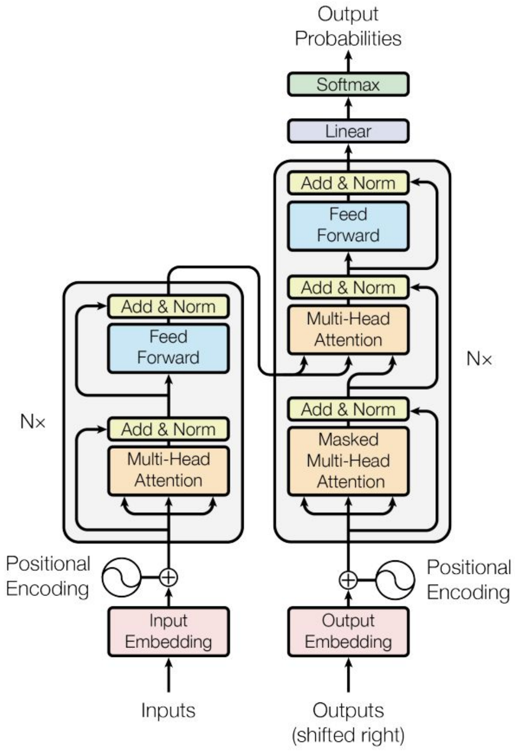
- Purpose-built for specific task
- Unique capabilities
- Uncompromising performance on limited task set

Large Language Models (LLMs) are a turning point

- Transformer[1] neural network that underpins GPT models is the dominant network architecture and adapts well to a broad array of tasks

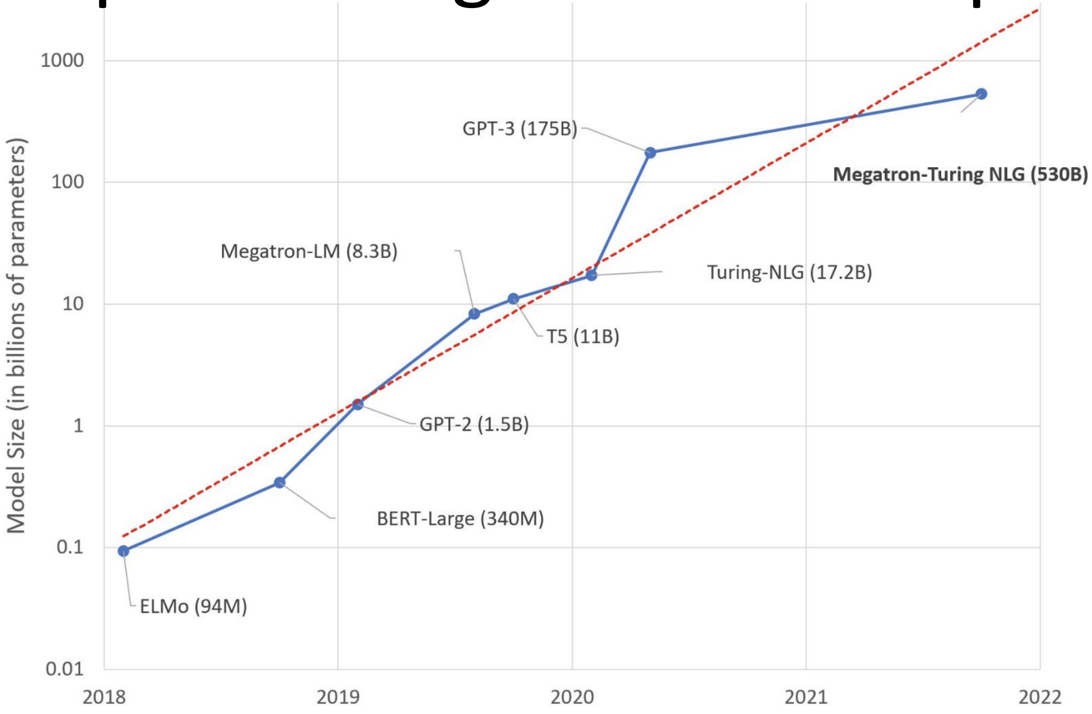
- Provides an efficient model for differentiable computing

- LLMs have experienced wide applications and are the modern day Oracle at Delphi



[1] Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

Demand for ever-increasing scale is driving exponential growth in computing resources



Larger Models

More memory

More Accelerators

https://huggingface.co/blog/large-language-models

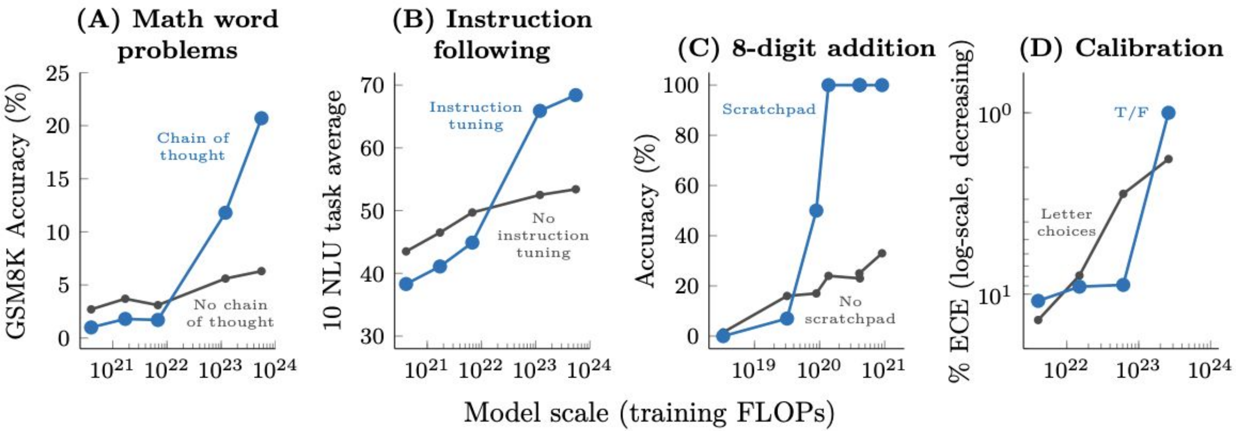
What trends are driving system design?

- New systems will need to provide **scale** among several dimensions
  - system size, number of parameters, training set
- LLMs are able to exploit scale to improve their learned behavior

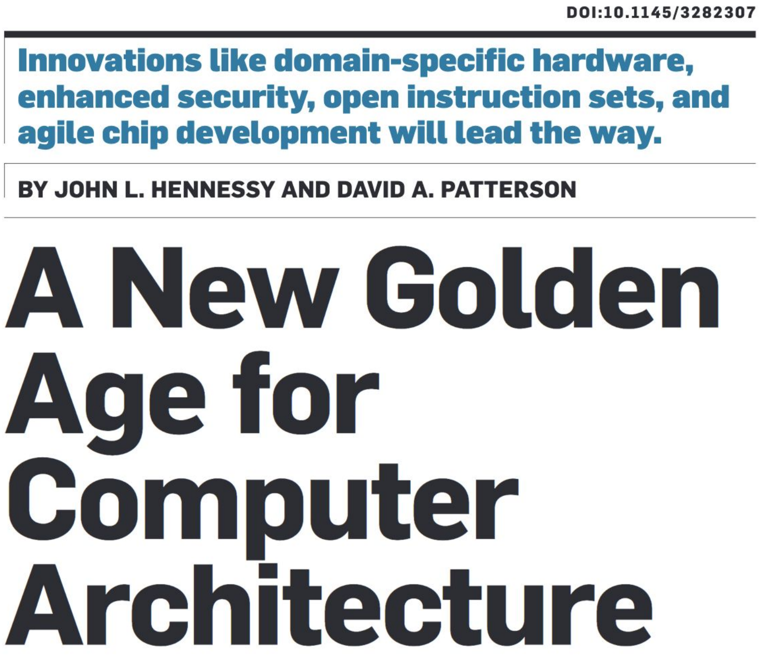
An ability is emergent if it is not present in smaller models but is present in larger models

- Accuracy improves and emergent capabilities occur with scaled compute

ChatGPT4 emergent capabilities



Domain-Specific Architectures

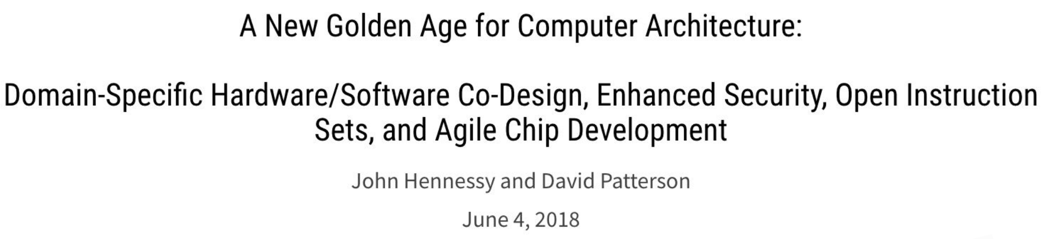


DOI:10.1145/3282307

Innovations like domain-specific hardware, enhanced security, open instruction sets, and agile chip development will lead the way.

BY JOHN L. HENNESSY AND DAVID A. PATTERSON

A New Golden Age for Computer Architecture

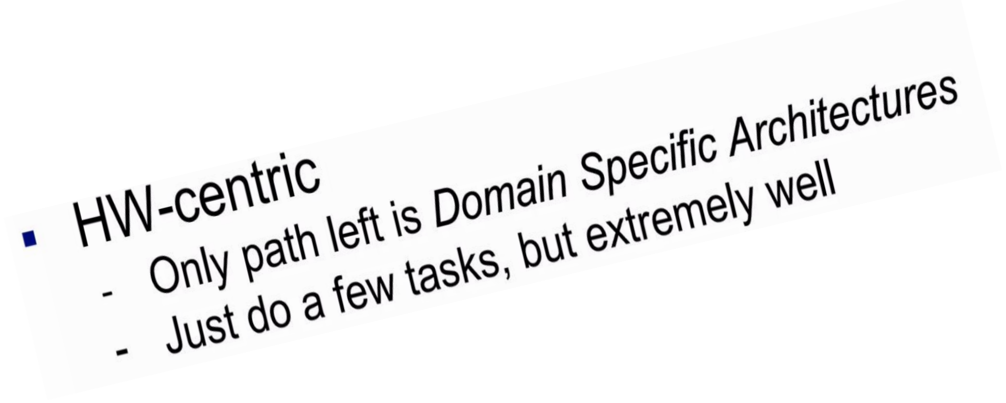


A New Golden Age for Computer Architecture:

Domain-Specific Hardware/Software Co-Design, Enhanced Security, Open Instruction Sets, and Agile Chip Development

John Hennessy and David Patterson

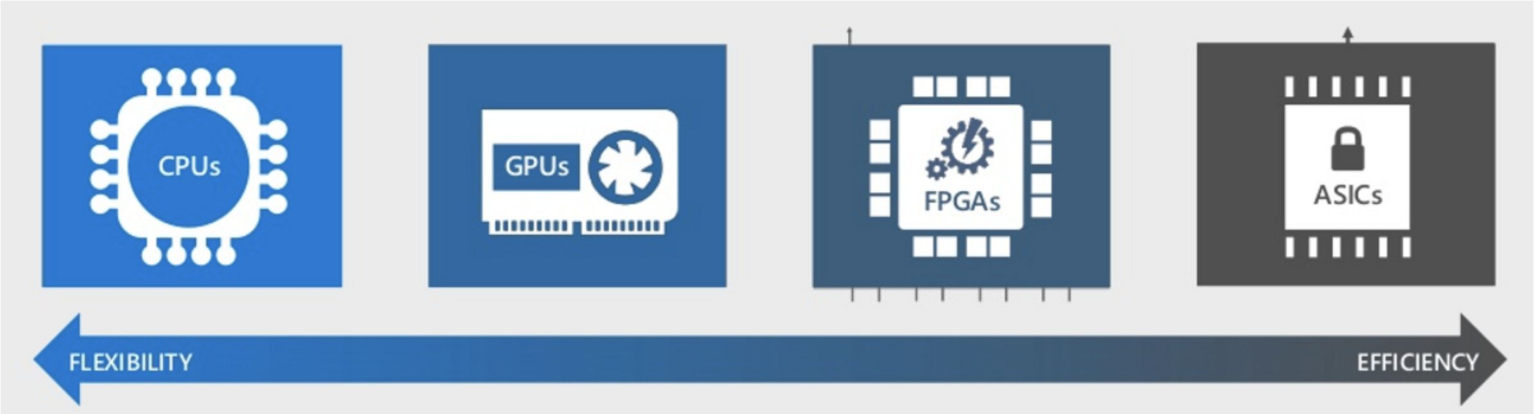
June 4, 2018



- HW-centric
  - Only path left is Domain Specific Architectures
  - Just do a few tasks, but extremely well

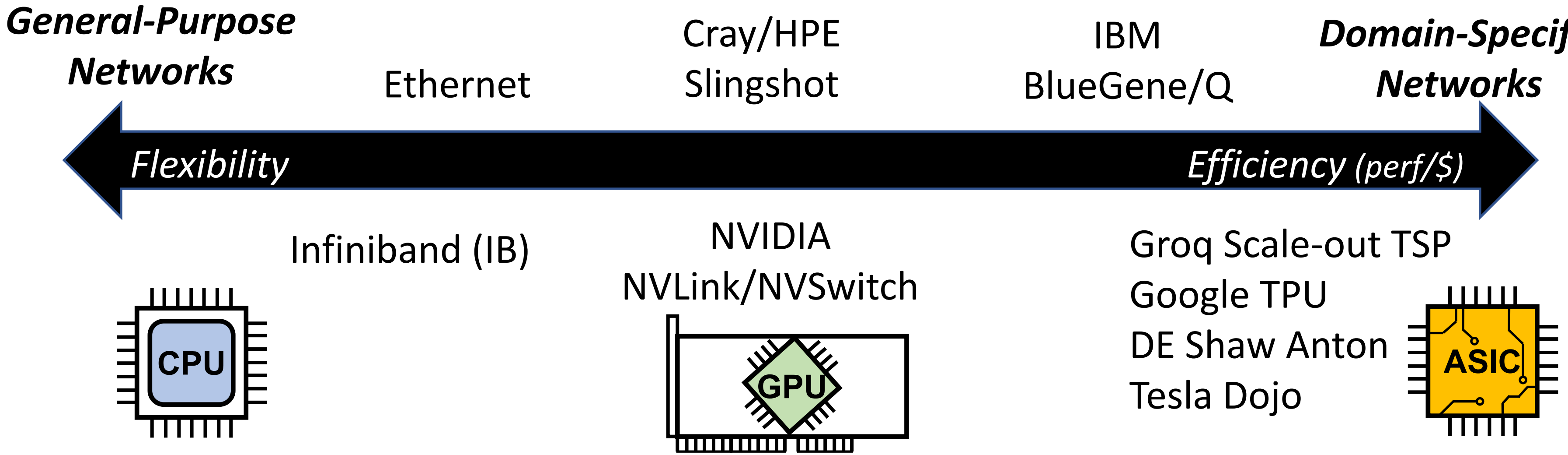
Domain-specific Processors

- Trending toward domain-specific **systems**
- CPUs - GPUs - FPGAs - DSAs are the processing elements
- Large-scale workloads require both computation *and* communication



Domain-specific ***Networks***

- Trading off generality and flexibility for purpose-built networks that accelerate the communication phases of the application
- A wide variety of designs across this space



General-Purpose Networks

Ethernet

Cray/HPE Slingshot

IBM BlueGene/Q

Domain-Specific Networks

Flexibility

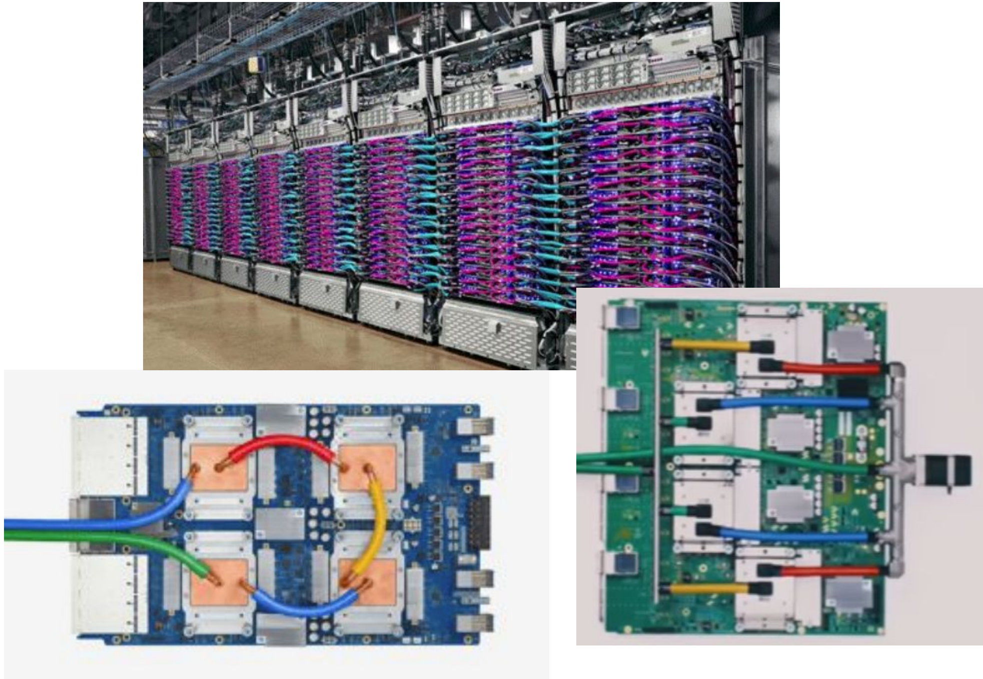
Efficiency (perf/$)

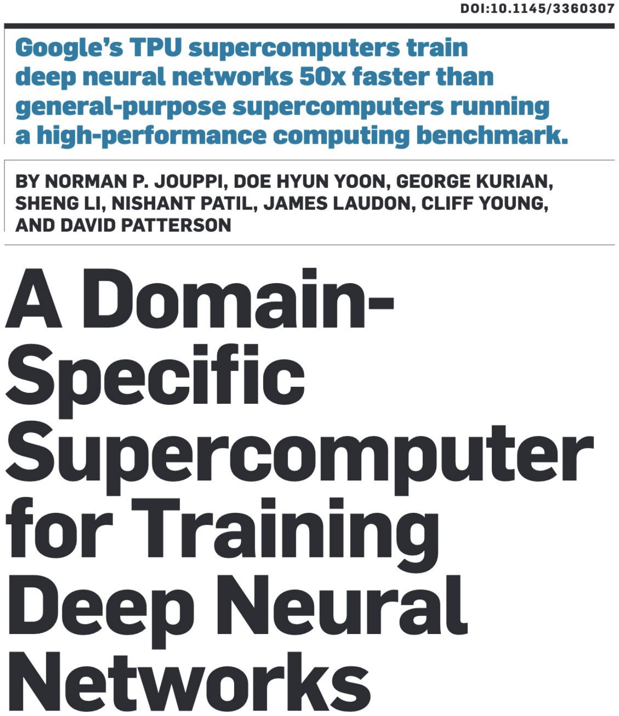
Infiniband (IB)

NVIDIA NVLink/NVSwitch

Groq Scale-out TSP
Google TPU
DE Shaw Anton
Tesla Dojo

Domain-Specific (AI) Supercomputers





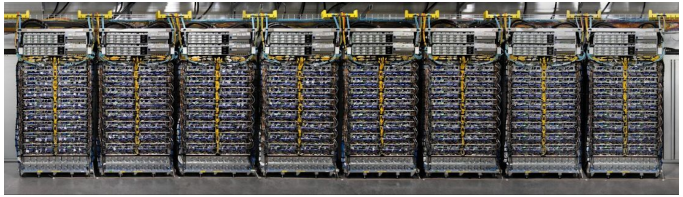
DOI:10.1145/3360307

Google's TPU supercomputers train deep neural networks 50x faster than general-purpose supercomputers running a high-performance computing benchmark.

BY NORMAN P. JOUPPI, DOE HYUN YOON, GEORGE KURIAN, SHENG LI, NISHANT PATIL, JAMES LAUDON, CLIFF YOUNG, AND DAVID PATTERSON

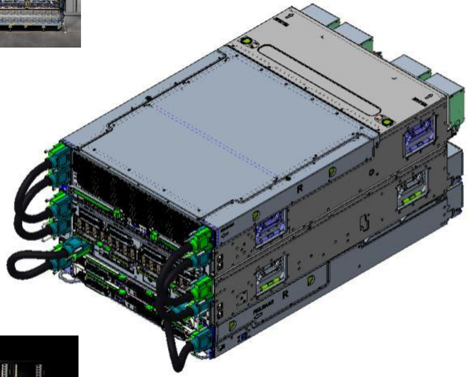
A Domain-Specific Supercomputer for Training Deep Neural Networks

Domain-Specific (AI) Supercomputers



TPU v4 (Google)

Dojo (Tesla)



ZionEX (Meta)

DGX H100 & Superpod (NVidia)

WSE 2 & CS-2 (Cerebras)

System Architecture Trends for ML Training

- Scale matters
  - More parameters in neural networks (increased computational intensity)
  - More endpoints used for training (thousands -> tens-of-thousands)
  - More training data (larger corpus of curated training data)

- Motivated specialized systems
  - Google’s TPU AI Supercomputer
  - Tesla DoJo training supercomputer
  - NVIDIA Salene supercomputer

Range of innovations in hardware accelerators

- Cambrian explosion of domain-specific hardware accelerators
  - Graphcore, Cerebras, TensTorrent, Groq, SambaNova…
  - All these systems were designed prior to 2017
    - "[Attention Is All You Need](#)" introduced the Transformer model architecture
  - Limited on-chip memory for weights - emphasis on CNN performance

- Rapid adoption of Transformers changed everything
  - Need more memory capacity and faster on-chip memory -> HBM
  - Exploit regular sparsity
  - Memory bandwidth for embedding lookups
  - Workload dominated by Vector * Matrix multiplication

Why Domain-Specific Networks?

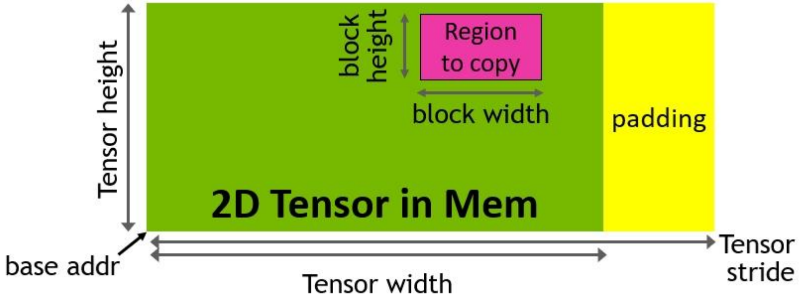
- Enable Domain-Specific Supercomputers
- Exploit multiple dimensions of domain-specific parallelism
- Low latency synchronization
- Efficient communication protocol (minimize overhead)
- Support Domain-Specific Communication (e.g., AllReduce, Gather/Scatter, better collective communication)

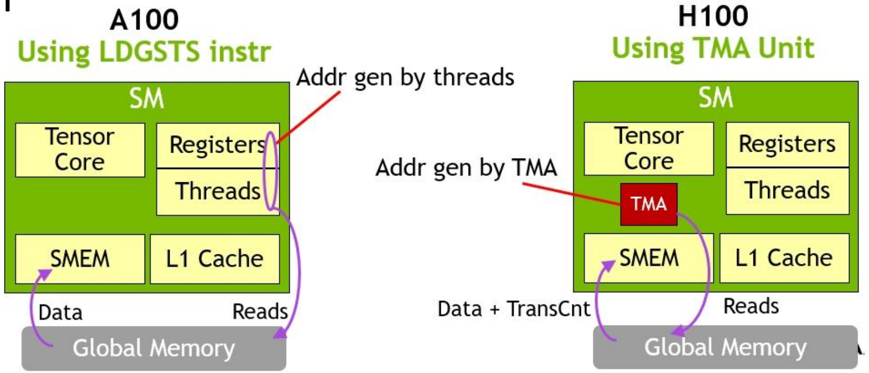
Support for transformers

- Efficient Gather/Scatter support in memory system for embeddings

- Asynchronous (offload) engine for Tensor Memory Acceleration (TMA)
  - Descriptor indicates the tensor's shape (width, height, padding, stride)
  - Avoids using ALUs in SM/threads for effective address generation

- Reduces end-to-end latency to remote copy a tensor



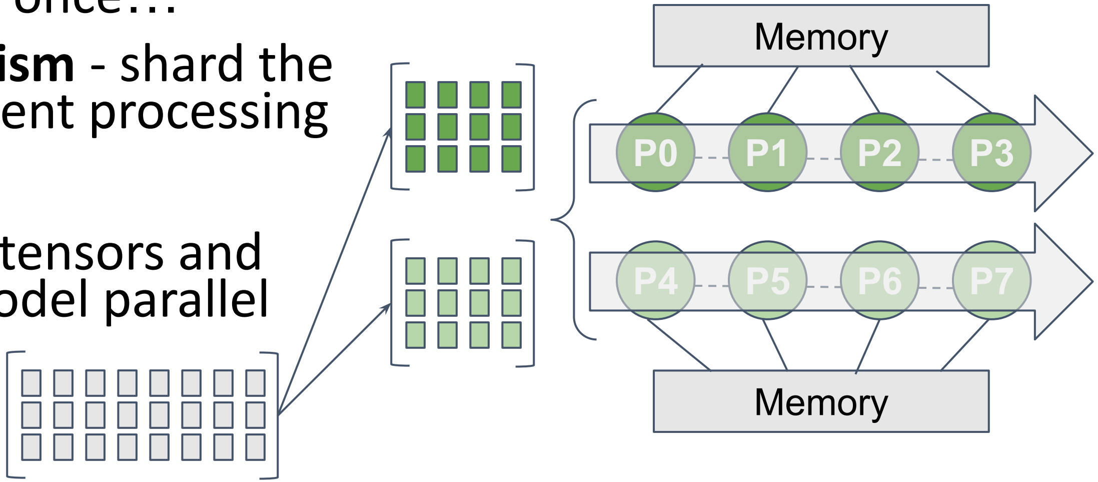


Parallelism in training deep learning networks

Everything, everywhere, all at once…

- **Model (pipeline) parallelism** - shard the model layers across different processing elements

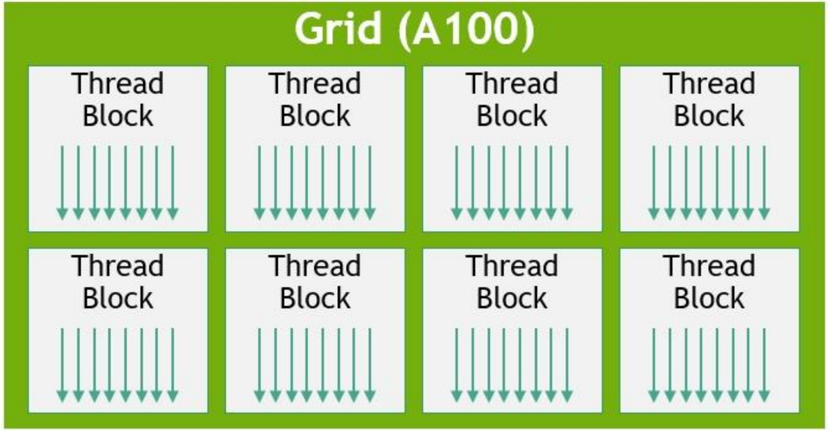
- **Tensor parallelism** - slice tensors and spread across multiple model parallel instances

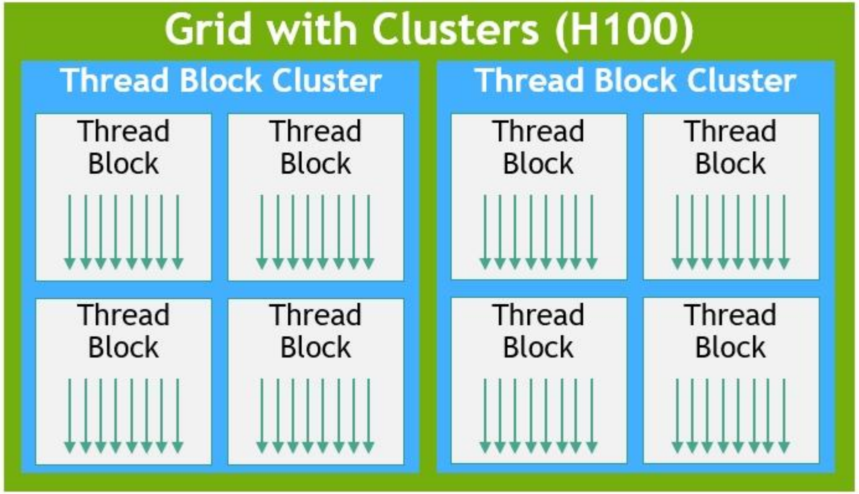


- **Data parallelism** - multiple independent mini-batches are trained in parallel, increasing batch size to crank up the computational intensity of each instance

Exploiting locality where possible

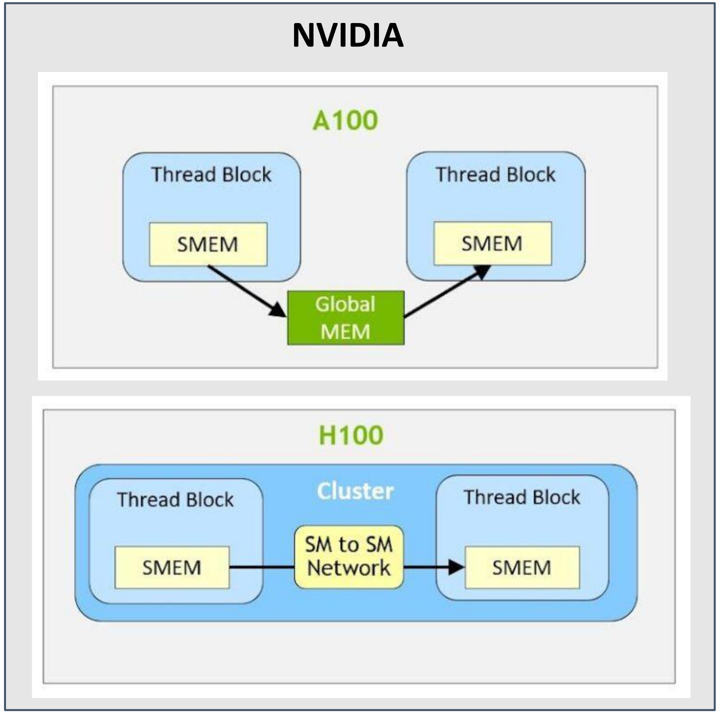
- Programmers will uncover parallelism and exploit locality
- NVIDIA Hopper introduces a new layer of hierarchy – thread block clusters – with efficient synchronization between SMs of a GPU
- New synchronization primitives allow low-latency communication between SMs in same cluster

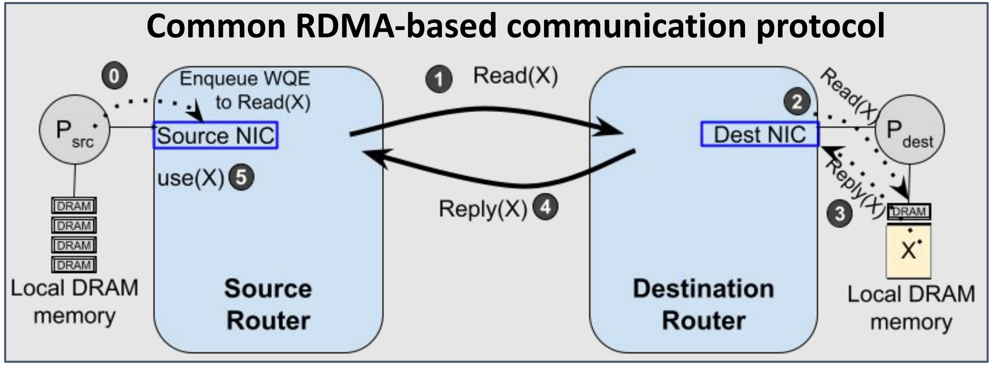


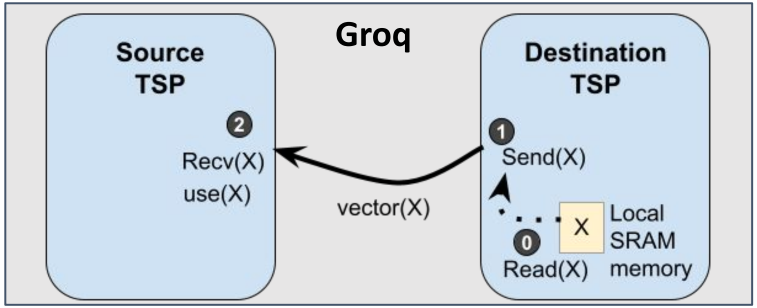


Fast distributed shared memory and synch.

- Direct access to shared memory for load/store/atomic operations







Efficient communication protocols

- The protocol efficiency (reducing overhead) and granularity (size of messages) of communication is paramount toward hiding communication latency in the shadow of computations

- Communication is carefully orchestrated to overlap with computation when possible
    - Model parallelism can spread the computation thinly across processing elements making communication overlap more difficult or impossible without fine-grained (small) messages

- Compute density to reduce communication cost
    - Keeping as much communication “local” is key
    - Eg: Tesla DoJo has 108 PFlops (BF16) per cabinet, minimizing the cost of global communication across long (expensive) optical links

Efficient synchronization across system

- As system size (scale) goes up, global synchronization cost increases

- AllReduce is the primitive used for exchanging gradients and ensuring each parallel worker has an up-to-date set of weights
  - Nearest-neighbor point-to-point communications for pipelined parallelism
  - AllReduce stresses the network's bisection bandwidth
    - In-network collectives (eg. NVSwitch/NVLink) provide hardware support directly in network switches

- Hierarchical AllReduce exploits the network topology and packaging boundaries to reduce the communication volume crossing the network's bisection.

Super Powers for Deep Learning…

- Multiple dimensions of parallelism are carefully orchestrated to align with the system’s architecture (memory hierarchy, network topology, and system packaging)

- Fine-grain communication is needed reduce synchronization overhead and exploit fine-grained parallelism (eg. model parallel and tensor parallelism)
  - Achieve good overlap among computation and communication

- Customized hardware for mapping tensor shape to memory (eg. TMA) and gather/scatter memory operations for embedding lookups

Conclusions

- Network design across domain-specific systems tradeoff specialization and generalization.
- Goal: Maximizing throughput and minimizing end-to-end latency
  - Efficient communication protocols and synchronization operations
  - Synchronous vs. Asynchronous communication modes
  - Lightweight (small) network packets to extend the load/store/atomics to globally shared memory
  - Unordered messages allows adaptive routing and hash-based bandwidth spreading to globally load-balance the network links
  - Single-sided messaging to avoid network round-trip
  - Exploit network topology for hierarchical collectives (AllReduce) at scale