Semiconductor Research Corporation

CENTER FOR EVOLVABLE COMPUTING

Energy Inefficiencies in Datacenter Computing

Josep Torrellas
University of Illinois Urbana-Champaign
Director, ACE Center for Evolvable Computing
(An SRC/DARPA JUMP2.0 Center)
https://acecenter.grainger.illinois.edu/

Compute-Energy Nexus Workshop, December 2025

General Directions

Leverage hardware accelerators and integration
Minimize data movement
Conserve costly memory
Do not leave the hardware unused

Move away from the current wasteful GPU path
Do not hope for breakthroughs in technology in the short term

The ACE Compute Paradigm

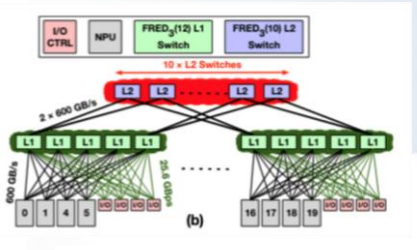
COMPUTE:

- Heterogenous hardware accelerators in datacenters
- New methodologies to generate, deploy and reconfigure accelerators
- Smart compilers/runtimes

MEMORY/STORAGE:

- Conserve memory: compress
- Heterogenous, distributed memory and storage
- Near- or in-memory/storage computing

ACE Compute Paradigm



COMMUNICATION:

- Reconfigurable network topologies
- Nimble runtimes that bundle computation in tiny buckets and ship them
- Accelerators in network switches and SmartNICs
- Application-specific designs

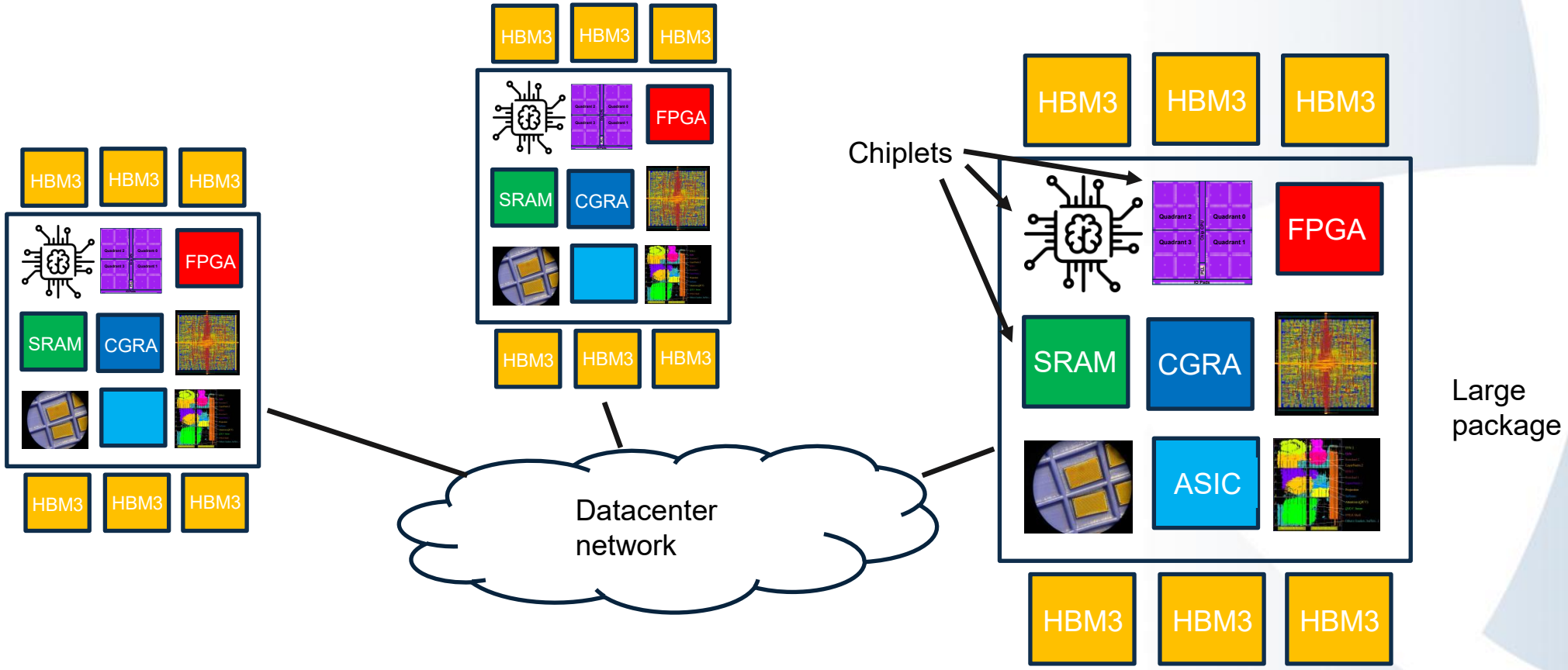
Semiconductor Research Corporation

ACE CENTER FOR EVOLVABLE COMPUTING

Compute

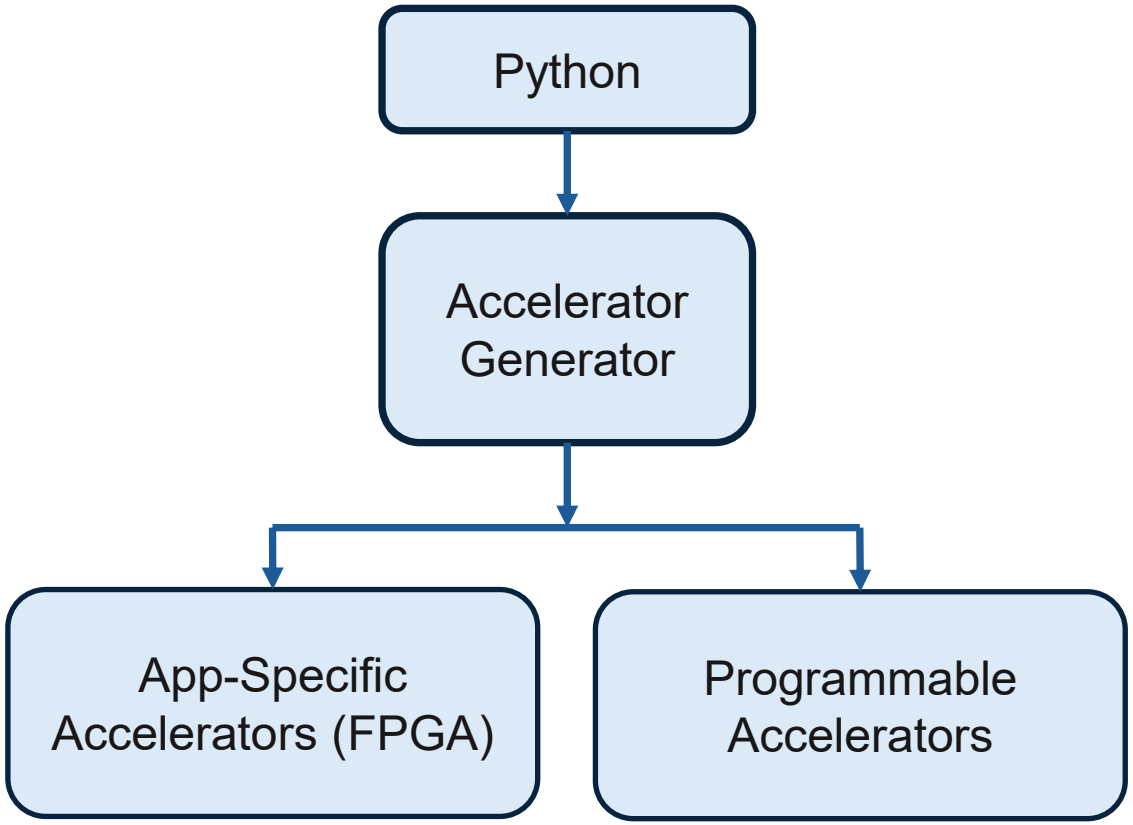
Move away from current GPUs; use accelerators

An Ensemble of Accelerators

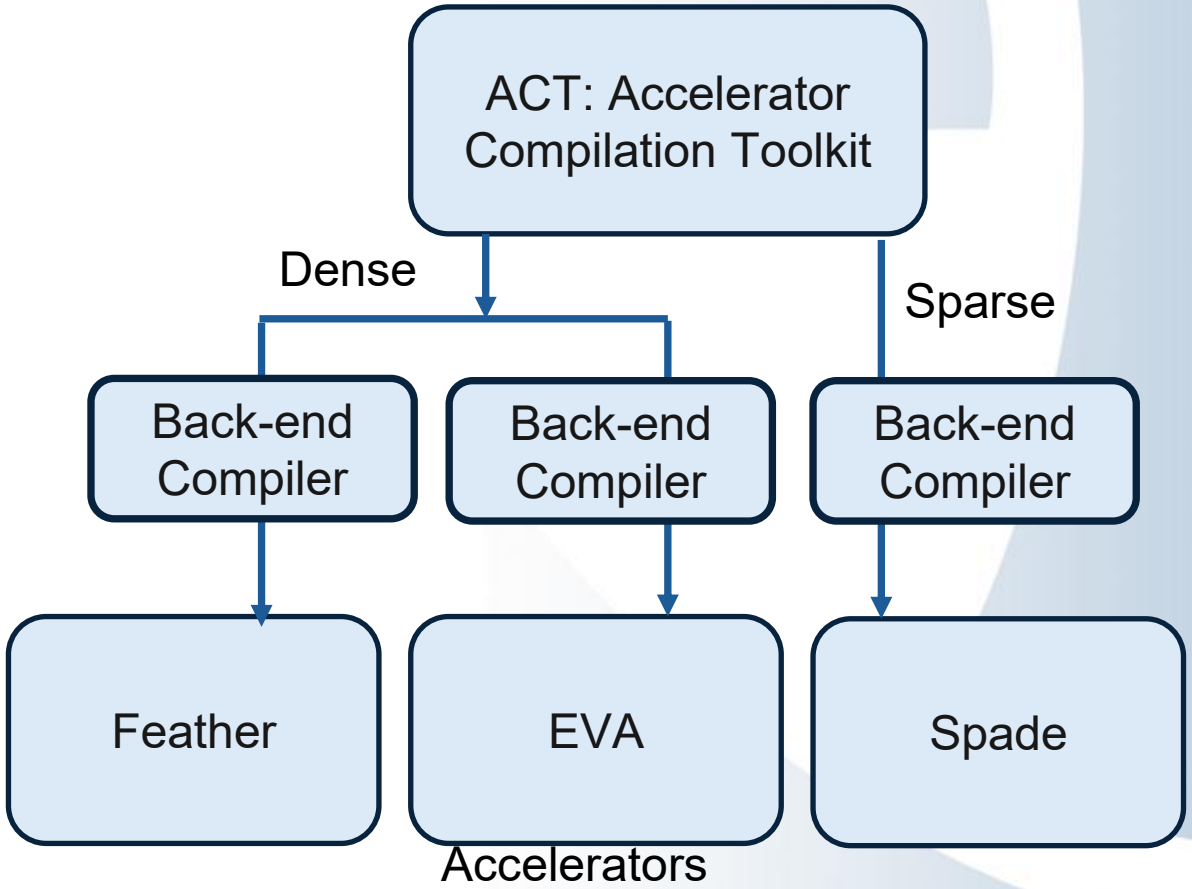


Distributed in the datacenter

Automatic Generation of Hardware Accelerators
And Automatic Generation of Back End Compilers

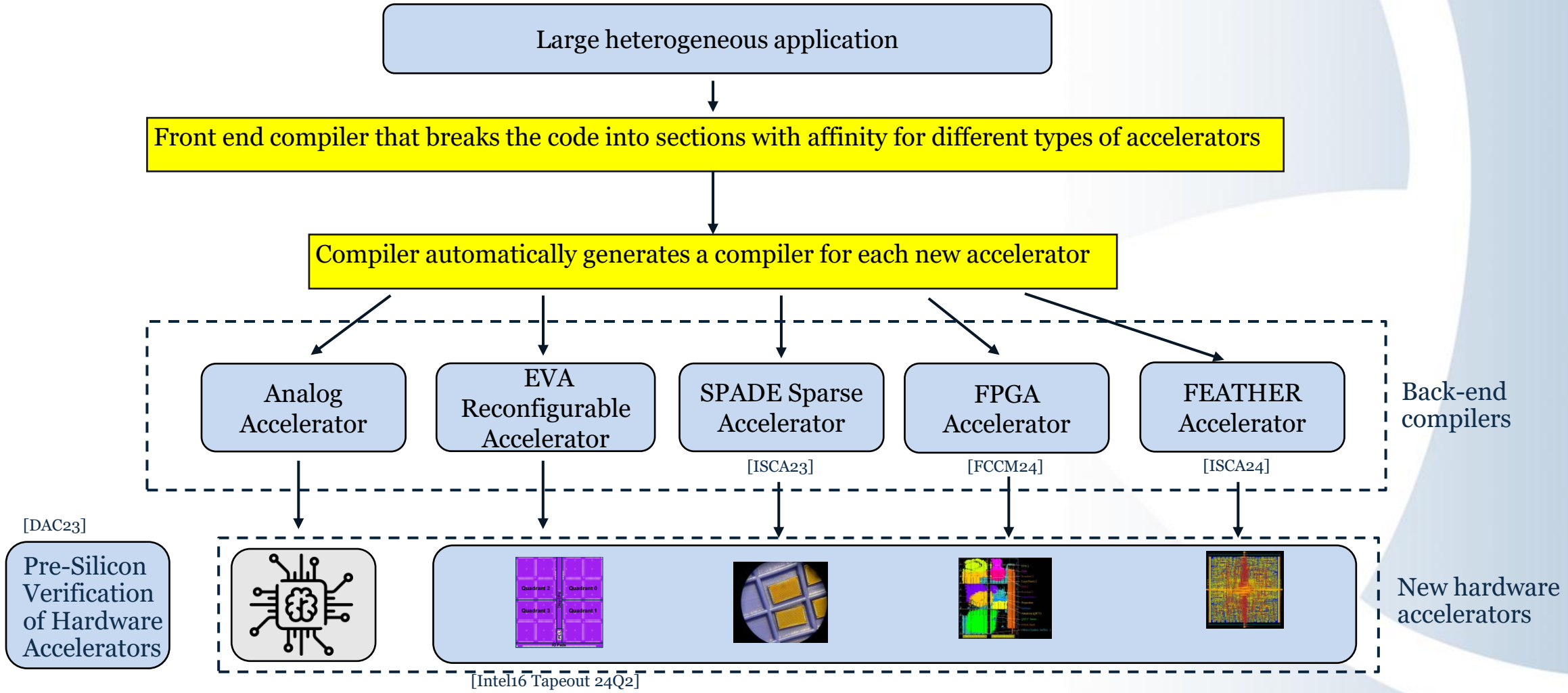


Evolvable Accelerator Generation



Evolvable Compiler Generation

A Software Framework to Program Multiple Accelerators

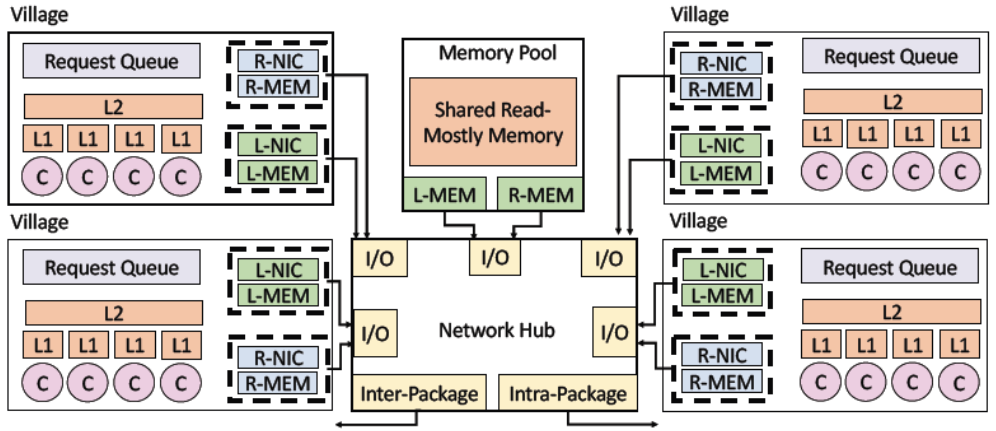


[ASPLOS24-1]: Z. Zhang et al., "SLAPO". [ASPLOS24-2]: C. Mendis et al., "TGLite". [ISCA23]: J. Torrellas et al., "SPADE". [DAC23]: S. Mitra et al., "G-QED". [FCCM24]: Z. Zhang et al., "LLM FPGA". [PLDI24]: Z. Zhang et al., "Allo". [ISCA24]: T. Krishna et al., "FEATHER". [MLBench23]: T. Krishna et al., "Chakra Traces".

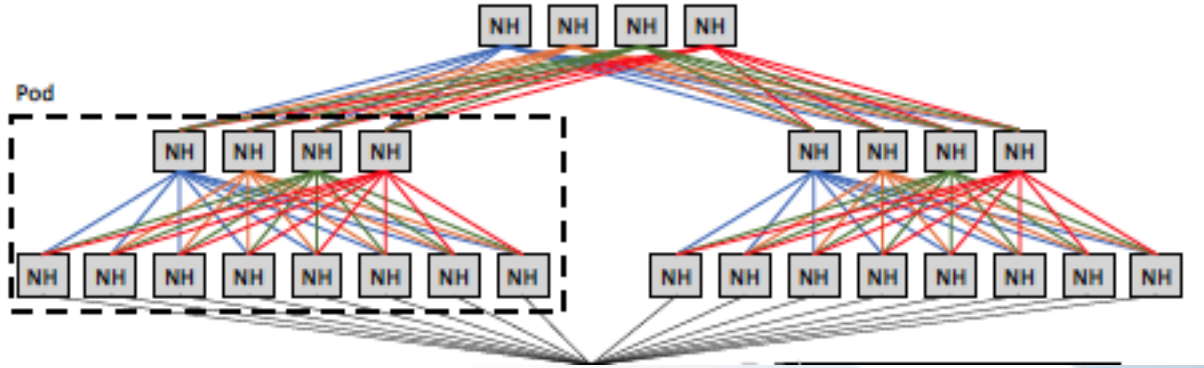
Specialize CPUs for Certain Workloads

µManycore: processor designed for cloud-native workloads (microservices and serverless)

- Main focus: limit **tail latency**
  - Hardware-supported request queue
  - Leaf-spine interconnect for many redundant, low-hop-count paths between clusters



Processor organized in chiplets of simple cores



High-connectivity on-chip interconnection network

Semiconductor Research Corporation

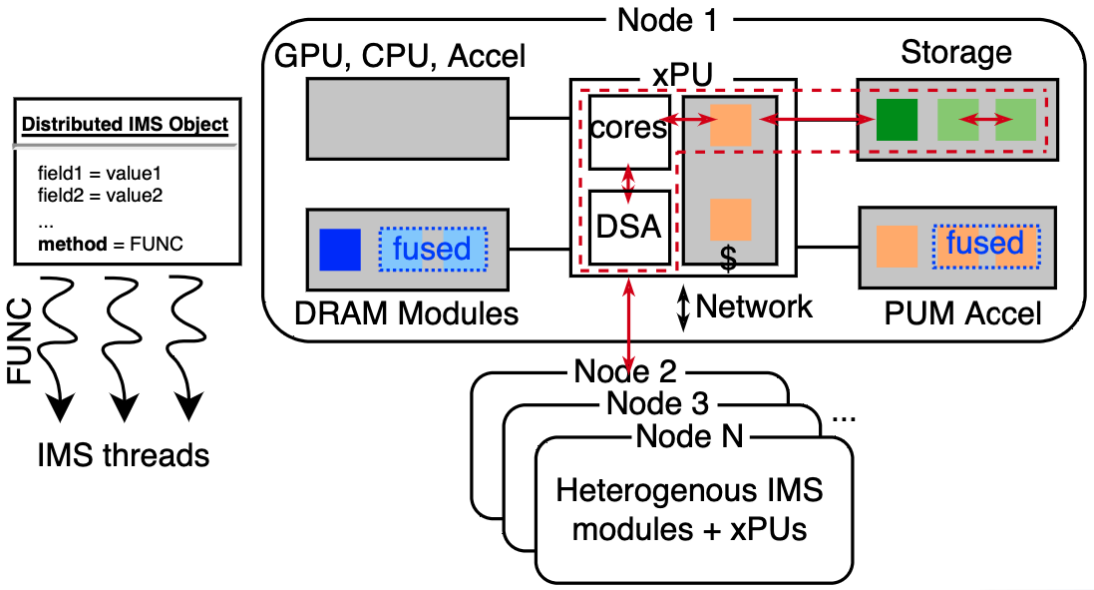
ACE CENTER FOR EVOLVABLE COMPUTING

Memory

Compute near memory

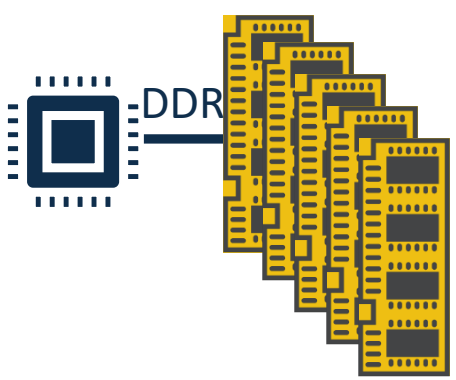
Near- and In-memory/storage (IMS) Acceleration

- Ubiquitous IMS blocks in memory hierarchy, network switches, SSDs
- Harness distributed IMS blocks to operate with coordination

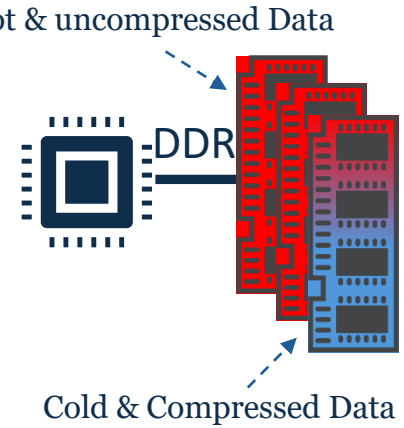


XFM: Near-Memory Acceleration of Software-Defined Far Memory

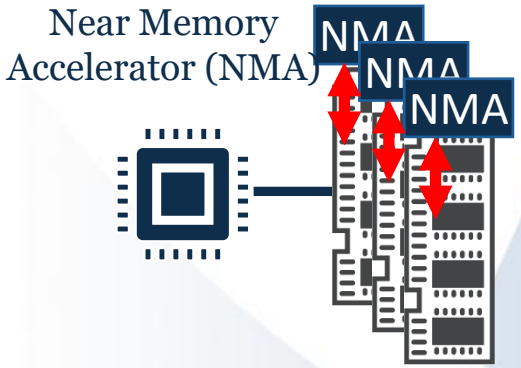
- Software-defined far memory (SFM): compressing cold memory pages
  - + Saves DRAM capacity at the expense of computation
  - - Hogs the memory channel
- *XFM:* offloads SFM operations on a Near-Memory Accelerator at the buffer device of the DIMM



Conventional Server



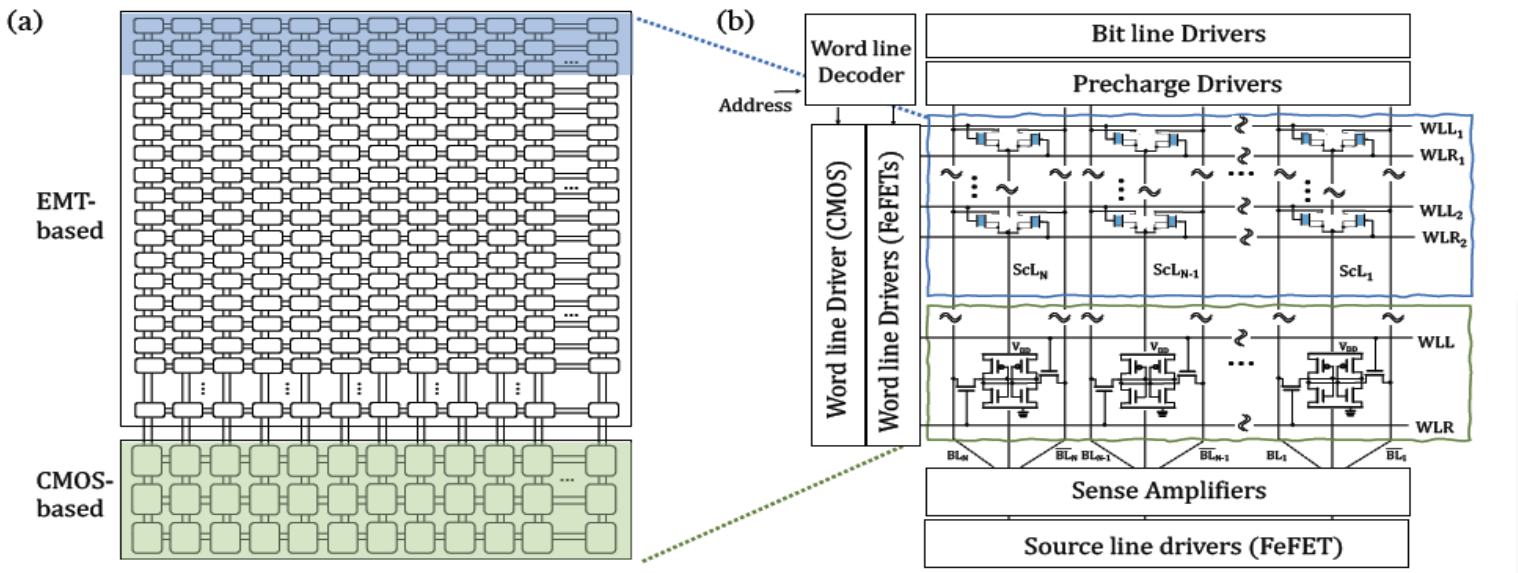
Software-Defined Far Memory (SFM)



XFM

Associative Processor in Memory that Combines CMOS and FeFET

- Integrated hybrid-technology SRAM
  - Majority FeFET (dense and energy efficient but slow)
  - CMOS buffers for select wr (fast but not dense or energy efficient)
- Hidden from programmer or compiler



Semiconductor Research Corporation

ACE CENTER FOR EVOLVABLE COMPUTING

Coordination/Communication

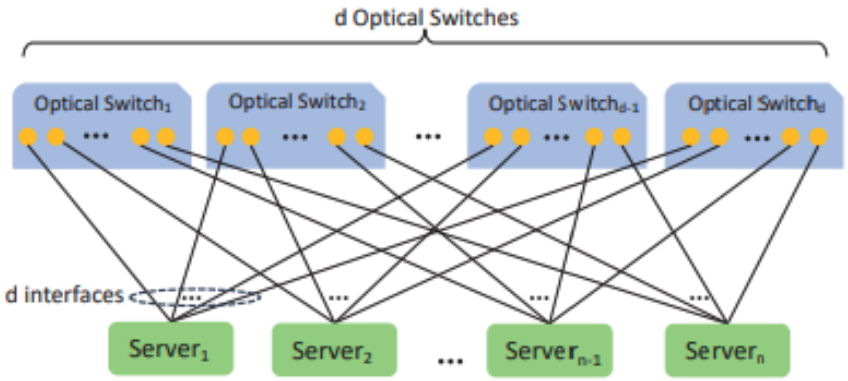
Do not leave the hardware unused

Reconfigurable Networking & Storage

Hardware in the cloud is massively underutilized. What can we do?

- Network: A reconfigurable optical interconnect that adjusts the network topology based on application requirements.
- Storage: A logically-centralized storage disaggregation system that steers I/O to specific SSDs based on knowledge of network topology and reconfiguration delays.

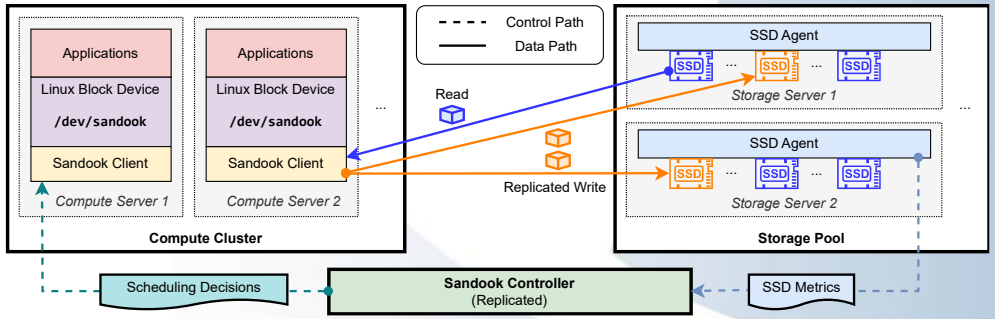
Network



Conceptual sketch of our reconfigurable infrastructure

Major gains in performance and in utilization

Storage



- SOTA: Shard across pool of flash SSDs → poor performance due to hotspots, heterogeneity, and read/write interference

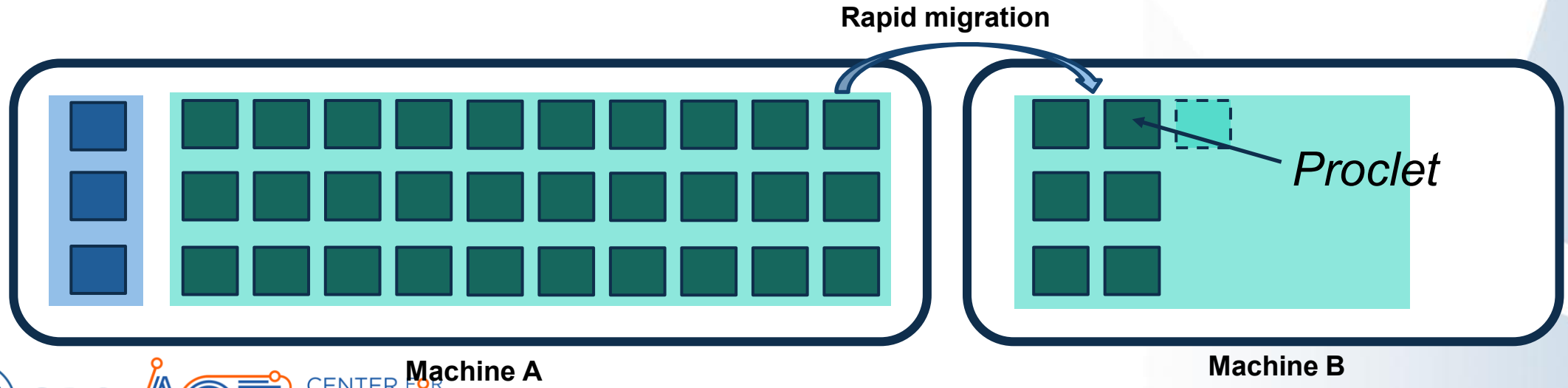
A Distributed Runtime for Self-balancing Systems

Accelerators should not be waiting for work

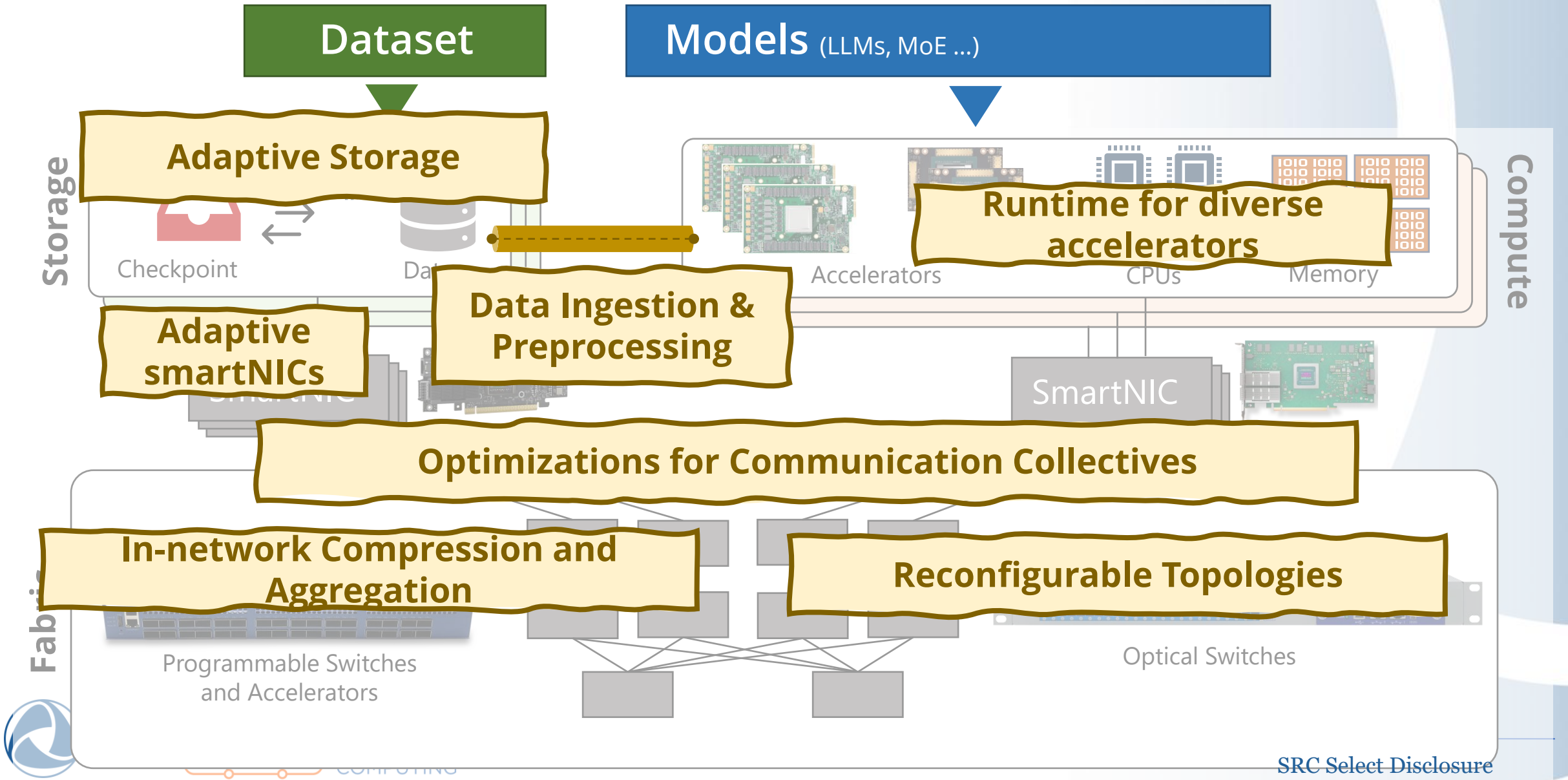
Data movement should be minimized

New programming model

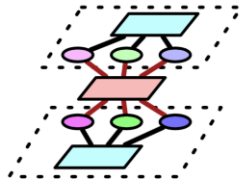
- Bundles computation in small tasks (*proclets)*
- Tasks rapidly migrate across machines
- Goals: Reduce load imbalance and minimize data movement



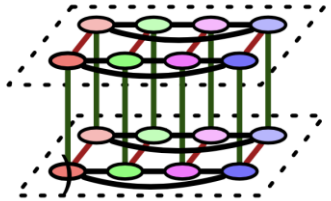
AI Training/Inference



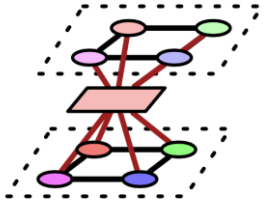
Automatically Synthesize Collective Communications



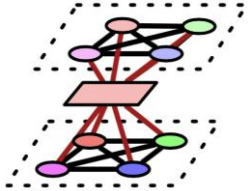
SW(3)_SW(2)
NVIDIA DGX-2
NVIDIA DGX-A100



R(4)_R(2)_R(2)
Google TPUv4



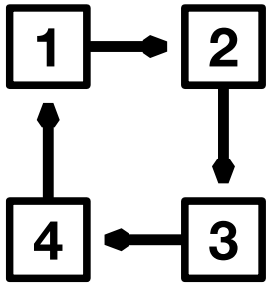
R(4)_SW(2)
Meta Zion
NVIDIA DGX-1



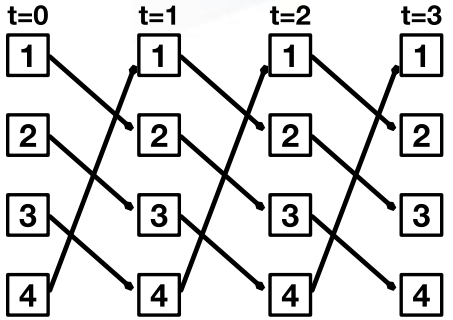
FC(4)_SW(2)
Intel
Habana

Multi-dim Networks

Input: collective communication pattern + network topology
Output: topology-aware collective algorithm

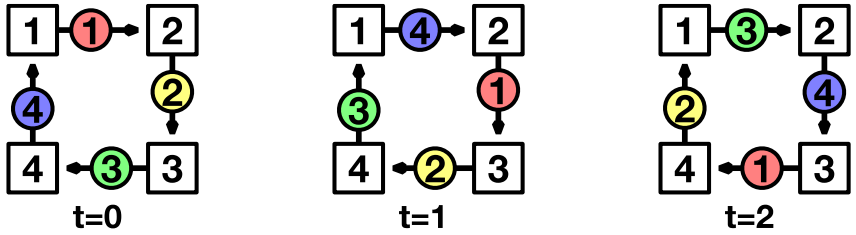


Topology

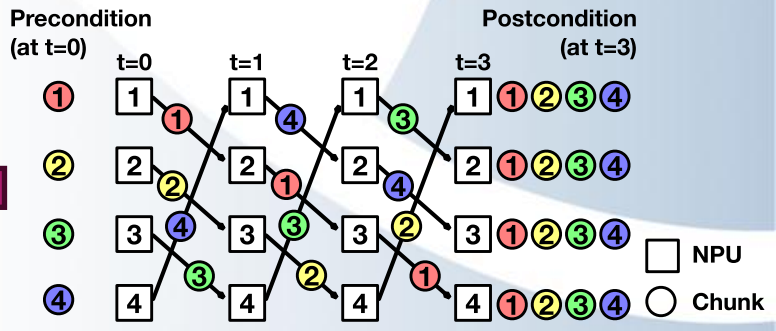


Time Expanded Network

Algo Synthesis



Collective Algorithm



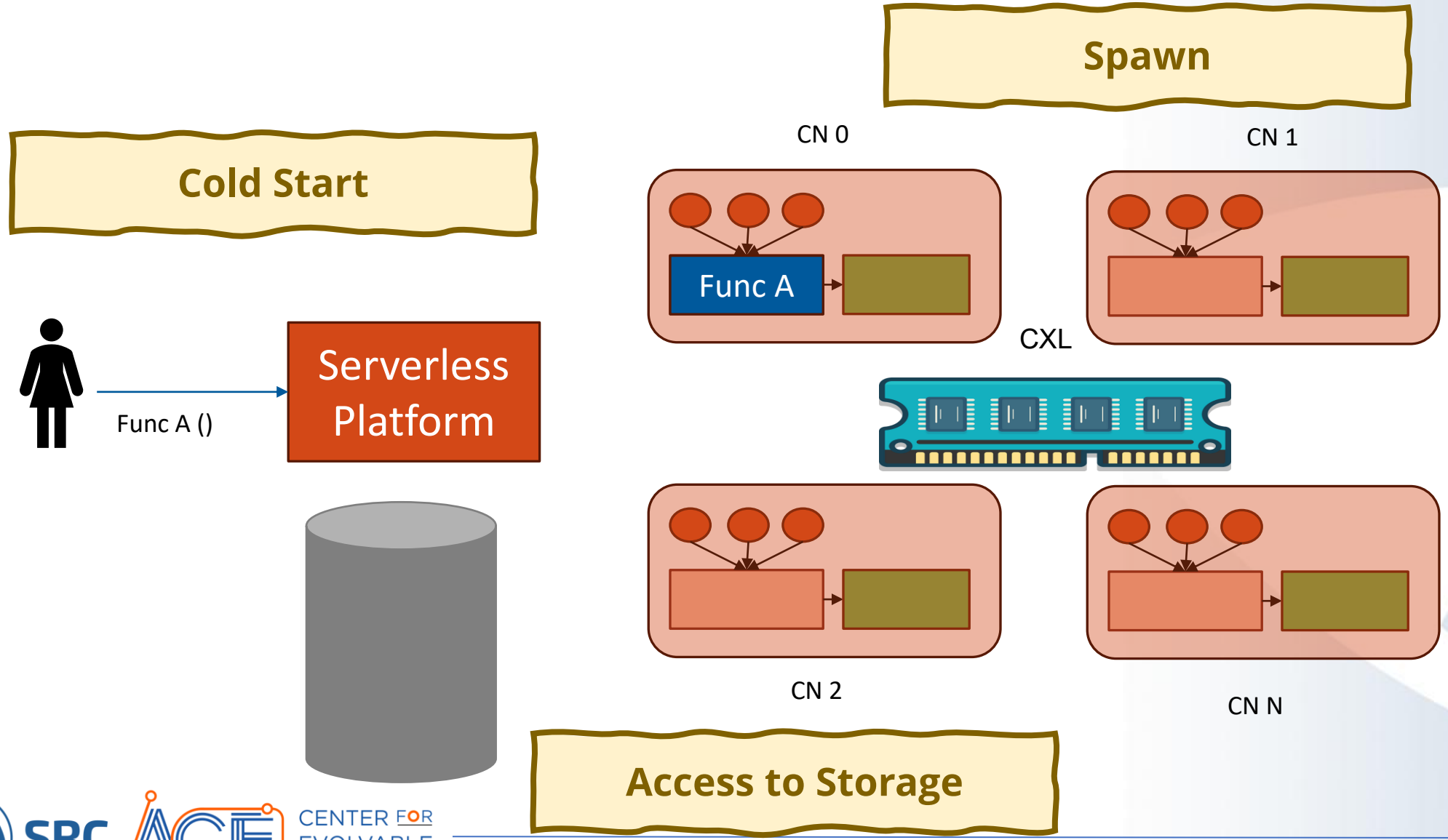
Collective-TEN Representation

SRC ACE CENTER FOR EVOLVABLE COMPUTING

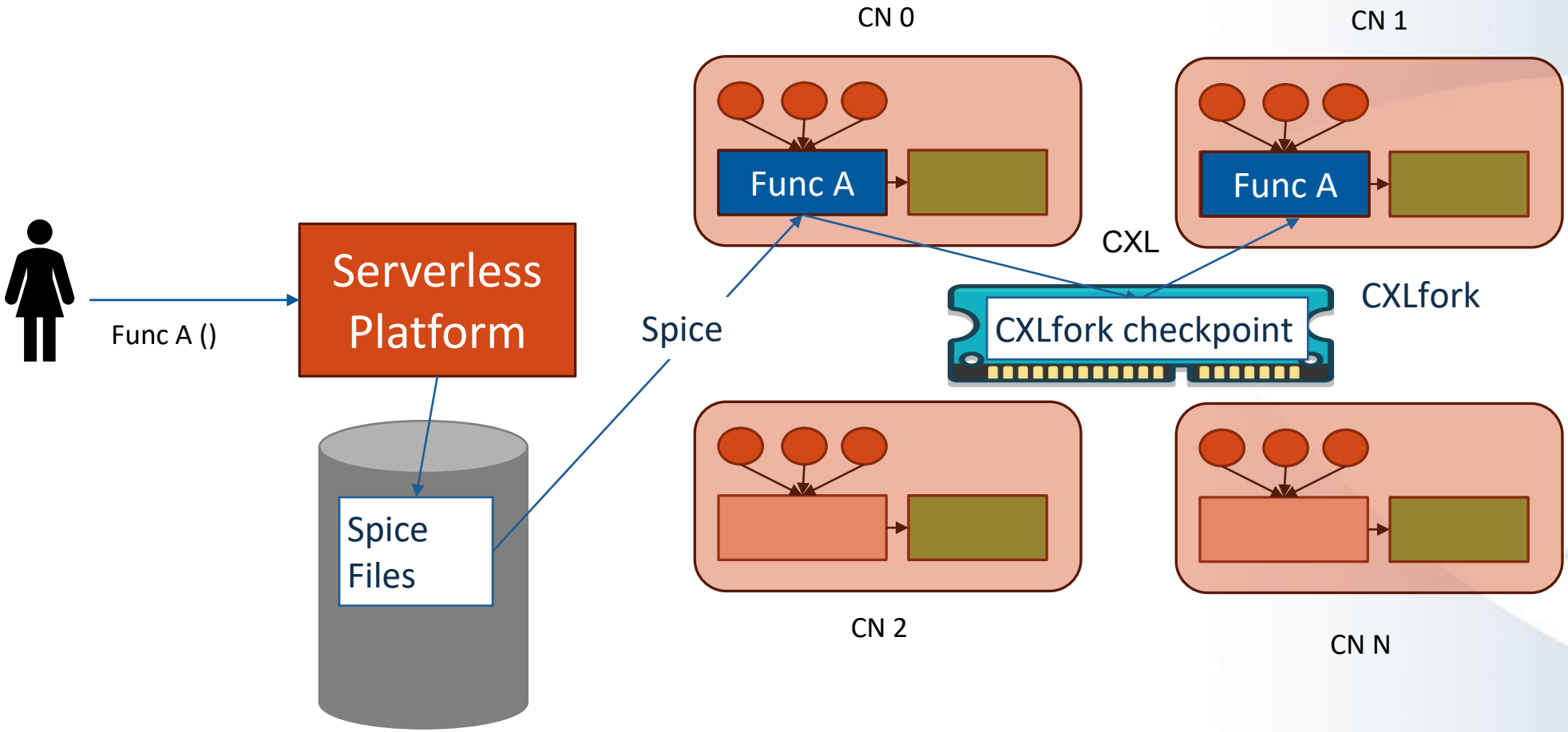
Other Optimizations for Collective Communications

- **Tensor Homomorphic Compression (THC) [NSDI'24]**
  - Compress gradients to reduce collective traffic
  - Direct aggregation of compressed tensors
- **In-network aggregation**
  - Aggregating (compressed) gradients in programmable switches or smartNICs to further speed up collectives
- **OptiReduce: A tail-optimal reduce [NSDI'25]**
  - Time-bounded AllReduce using Transpose AllReduce, unreliable bounded transport, and adaptive timeouts

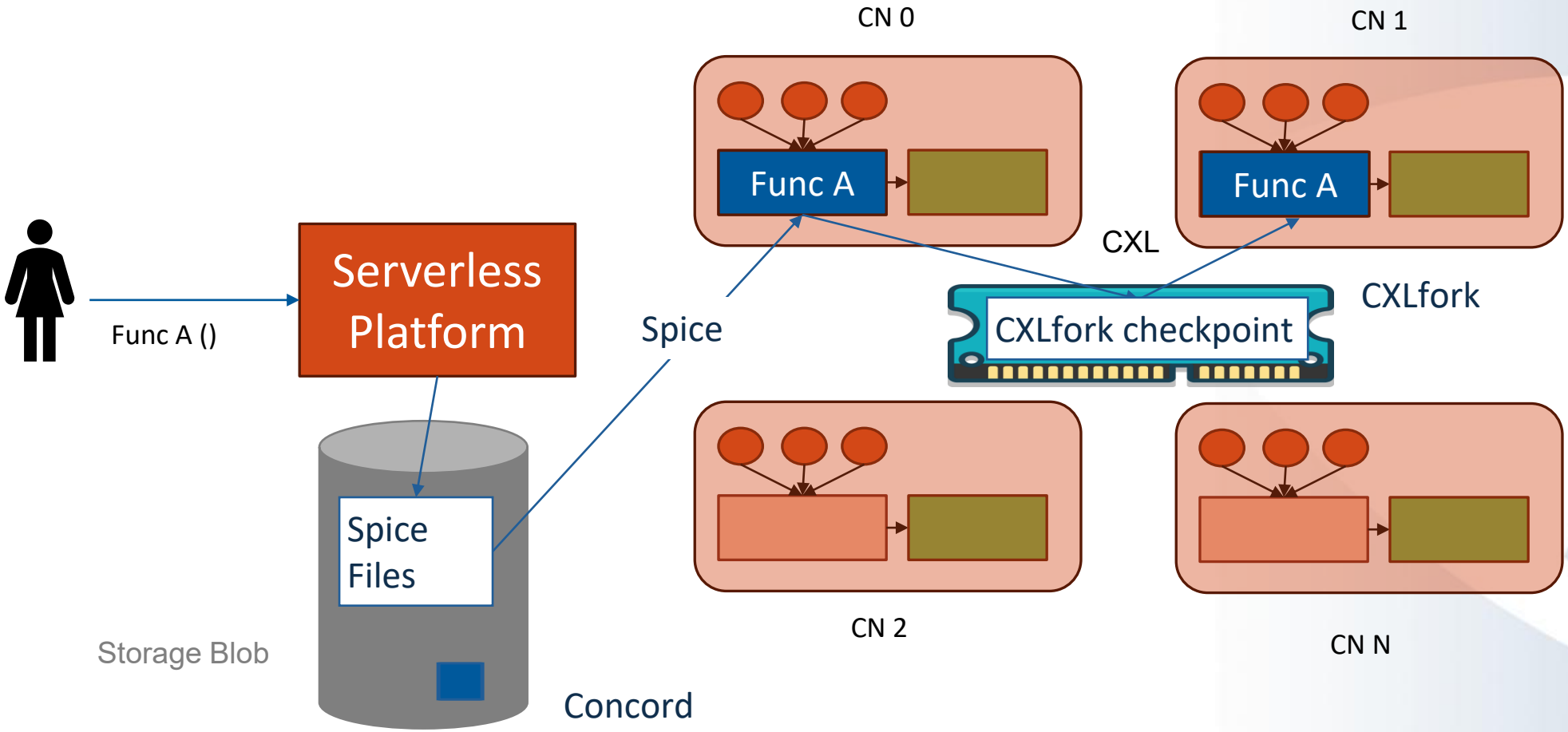
Microservice/Serverless Cloud Environments



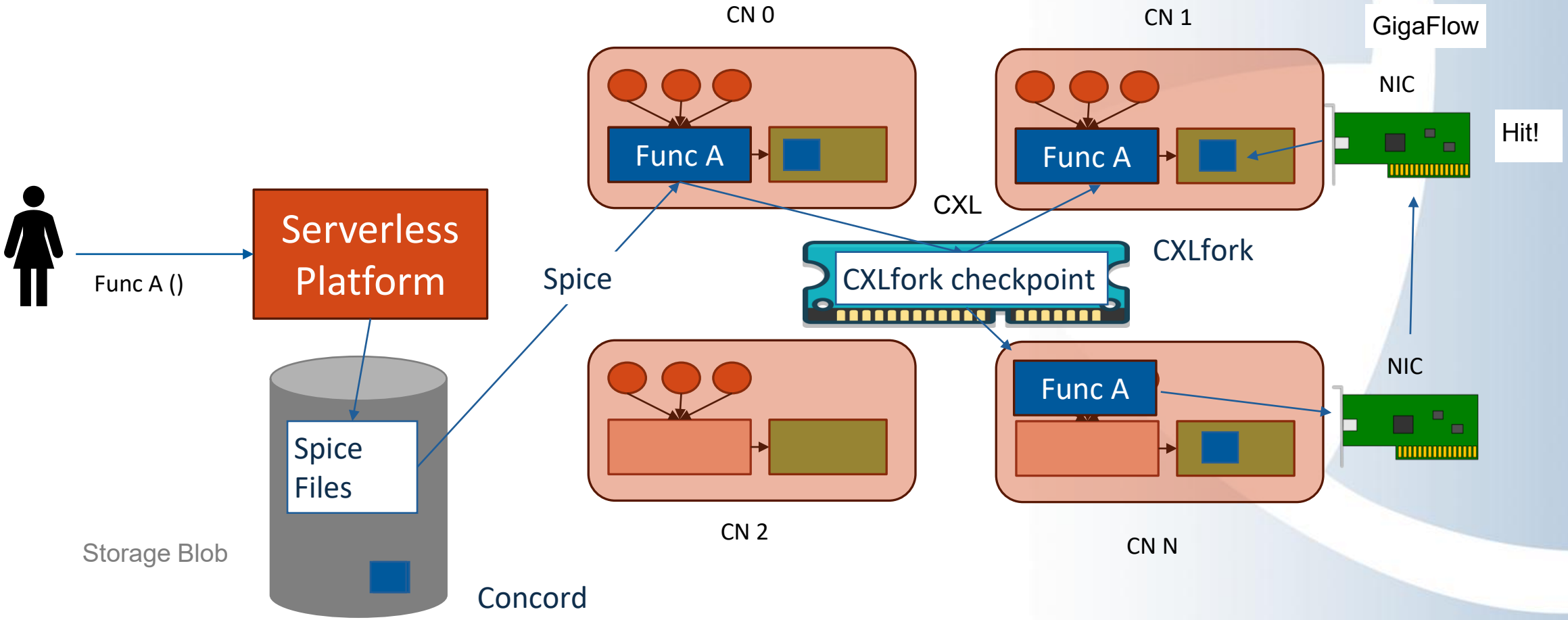
Spice + CXLfork: Cold Start and Remote Fork



Concord: Caching State in Nodes

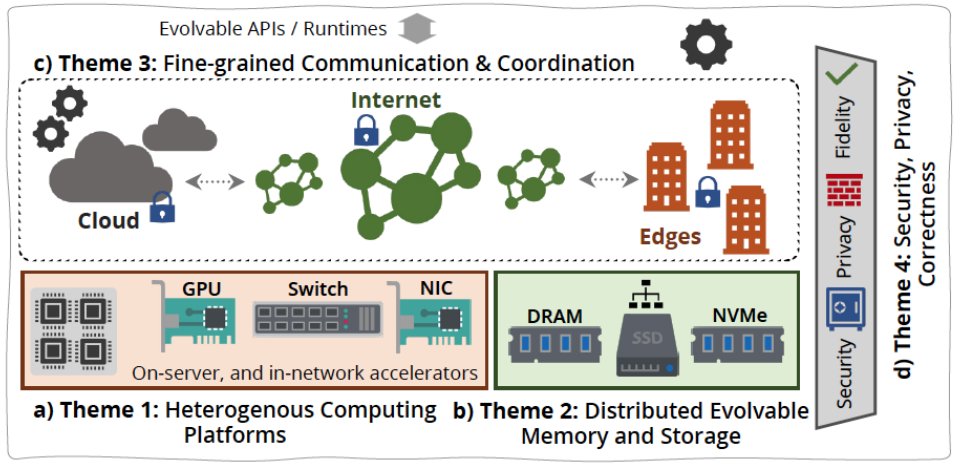


Gigaflow: Cache Packet-Processing Routines in NICs



Call to Action

- Current GPU computing is very inefficient → move to accelerators (and specialized CPUs)
- Data movement and hardware idle time → many software techniques, often application-specific
- Memory → processing near memory



Semiconductor Research Corporation

CENTER FOR EVOLVABLE COMPUTING

Energy Inefficiencies in Datacenter Computing

Josep Torrellas
University of Illinois Urbana-Champaign
Director, ACE Center for Evolvable Computing
(An SRC/DARPA JUMP2.0 Center)

Compute-Energy Nexus Workshop, December 2025

https://acecenter.grainger.illinois.edu/

CC-NIC: A Cache-Coherent Network Interface Card (NIC)

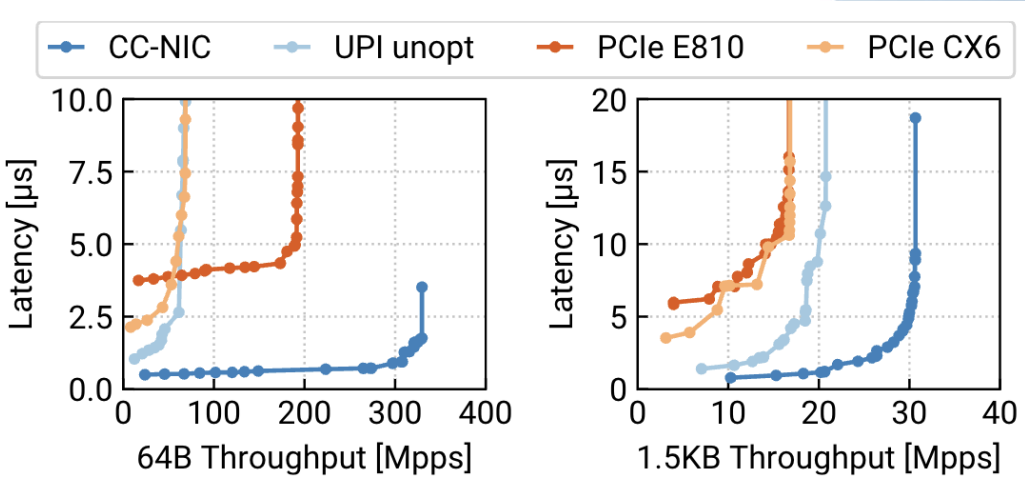
ASPLOS 2024

Today's NIC-Host interfaces are inefficient: PCIe is >80% of intra-rack roundtrip latency

Proposal: Use emerging interconnects (CXL, UPI, UCIe) to allow NIC to participate in CPU cache coherence

CC-NIC is a cache-coherent NIC design

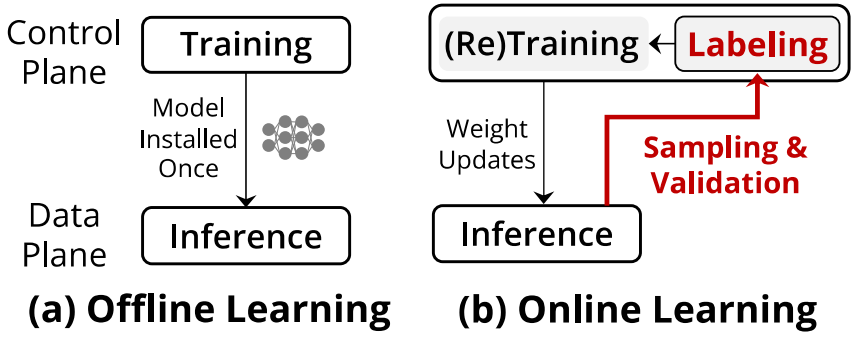
- New techniques in:
  - Host-device signaling
  - Cache-optimized memory layouts
  - Symmetric, shared buffer management
- CC-NIC emulated on Intel UPI:
  - Terabit throughput. Significant latency reduction
  - Up to 50% core savings for app workloads
- Opens the door to more off loading of work to SmartNICs



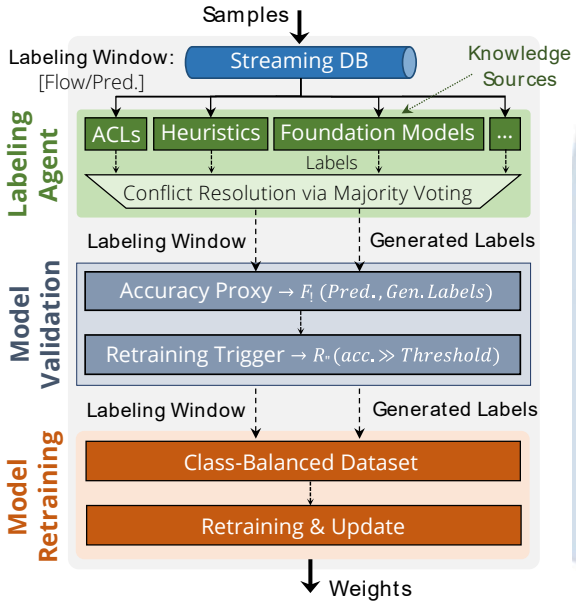
Computing in Network Switches

ASPLOS 2023
OSDI 2024

- *Homunculus*: automatically generates efficient ML models to run on switches (e.g., check for malware)
- *Caravan*: ML models learn online
- Other possible applications: Straggler mitigation

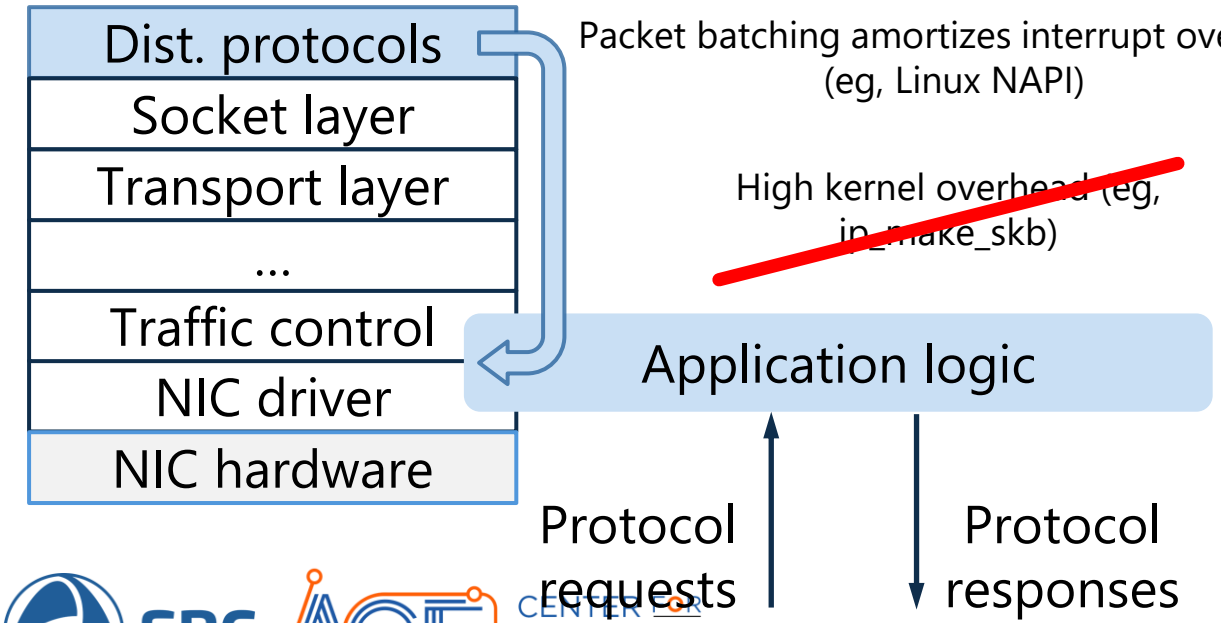


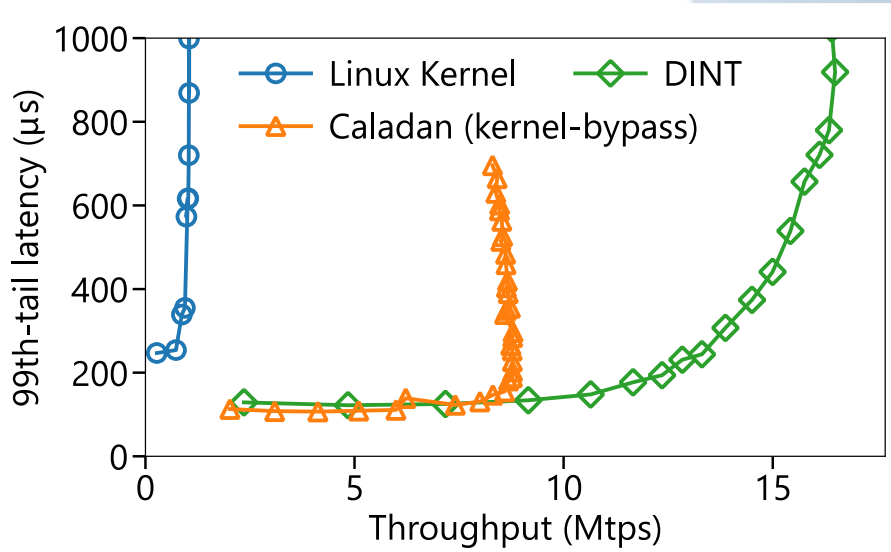
CARAVAN:



Application-customized Networking Stack with eBPF

- Accelerate distributed system network software while maintaining safety
  - *Electrode*: Accelerate consensus protocols with new communication primitives in eBPF
  - *Dint*: Accelerate distributed transactions (lock manager, key-value store, log manager)

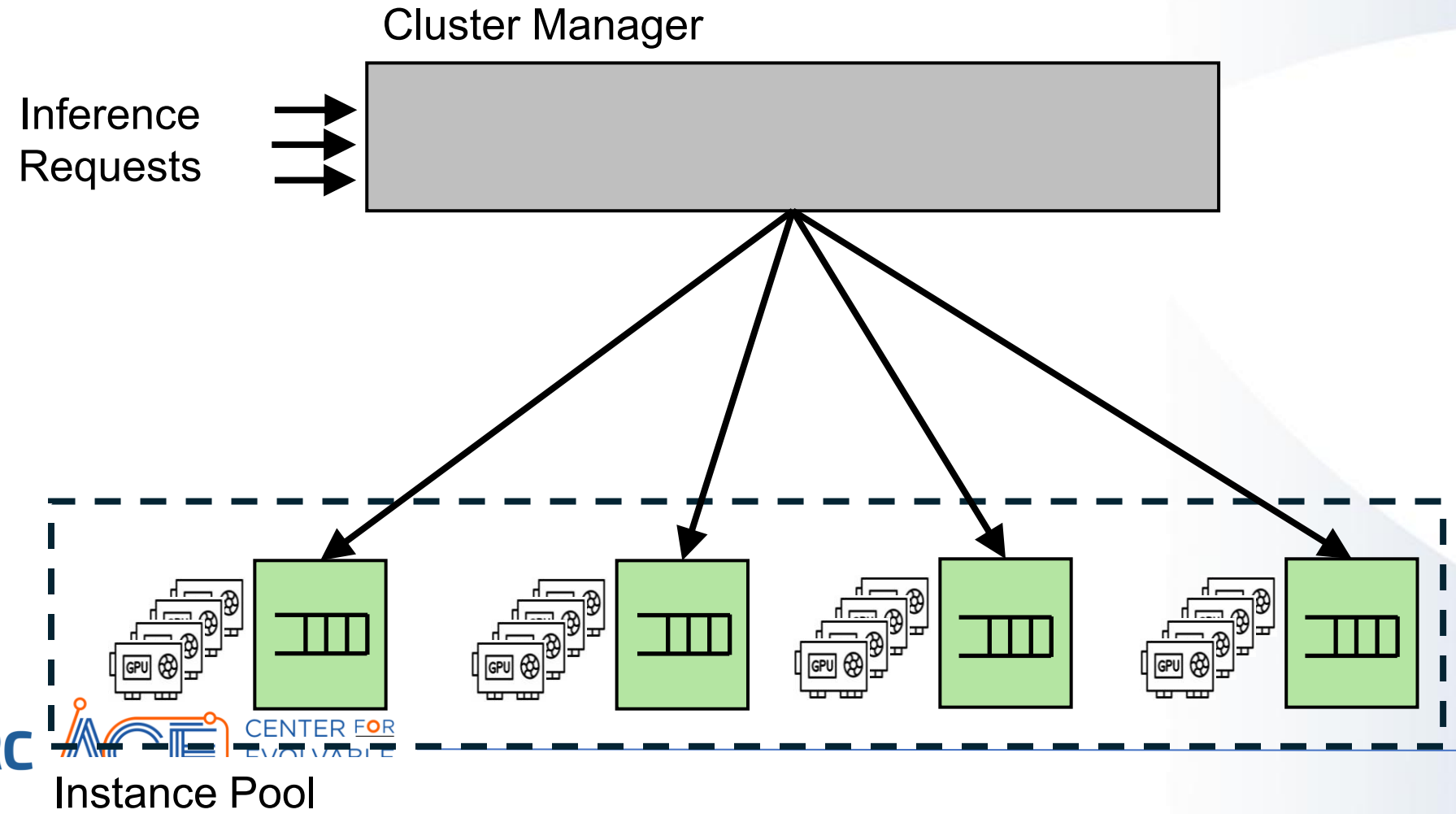




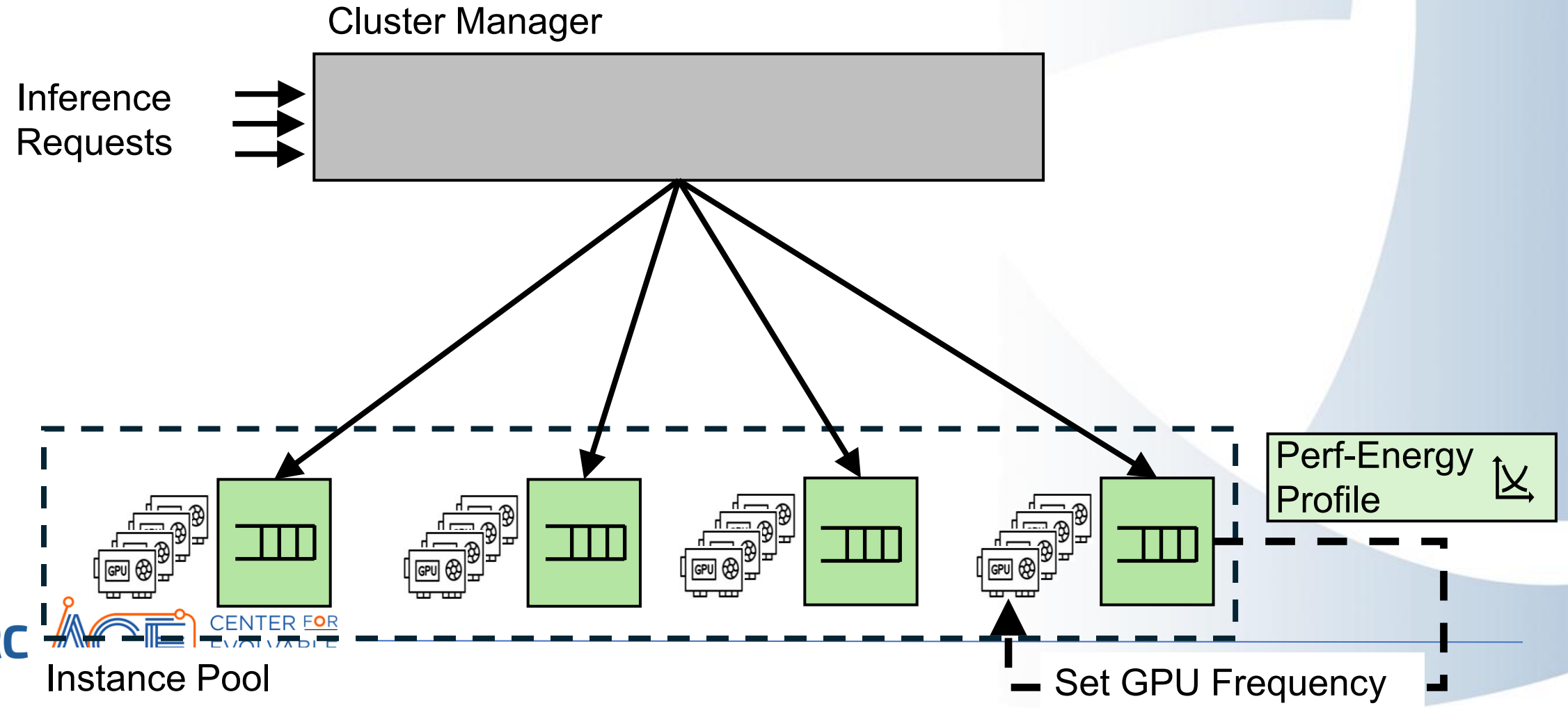
DynamoLLM: Energy-Management Framework for LLM Inference Clusters

- 1. Profile-driven energy configuration setting
- 2. Instance pools for diverse workloads
- 3. Hierarchical control for dynamic load

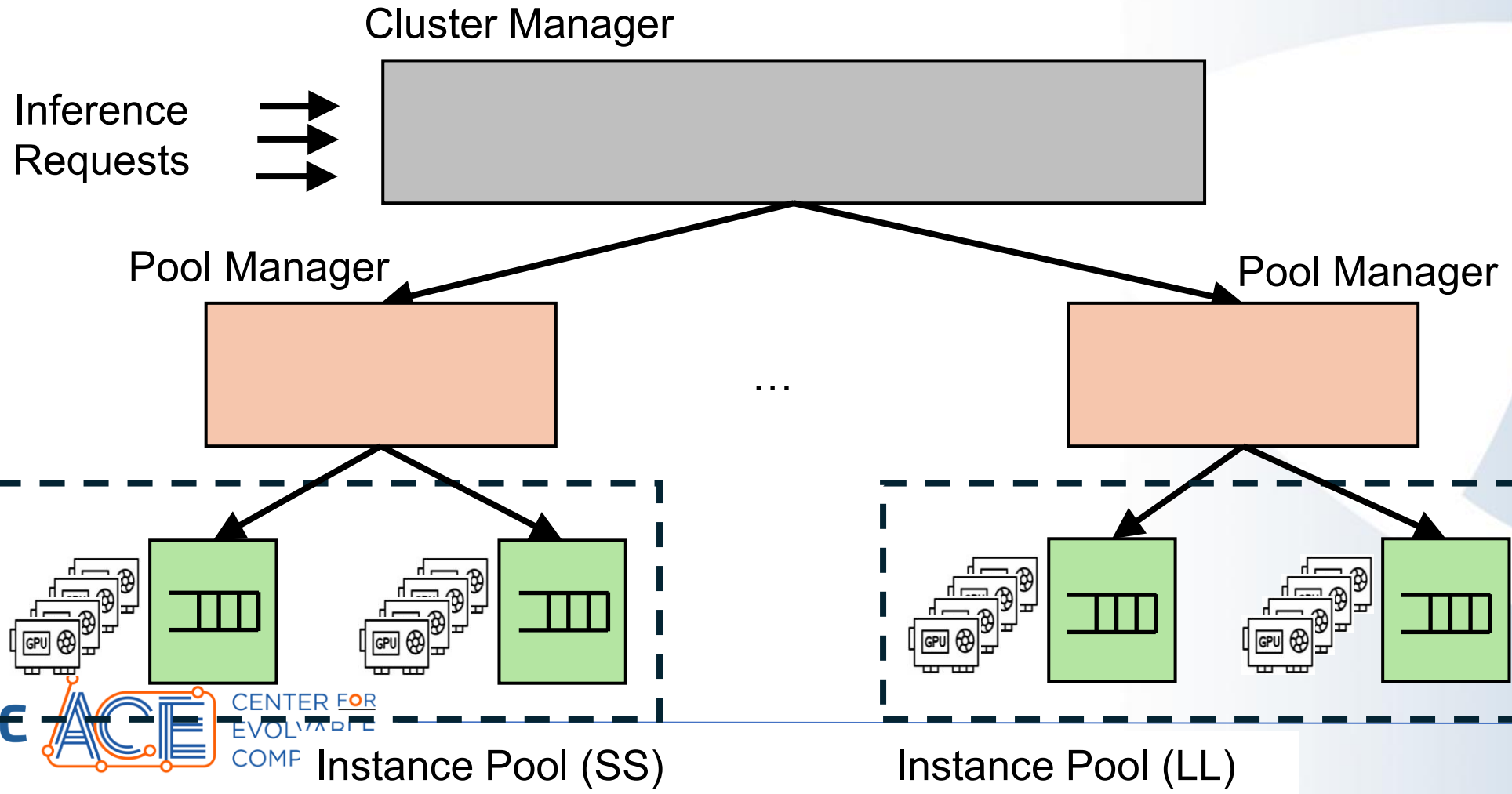
DynamoLLM: Energy-Management Framework for LLM Inference Clusters



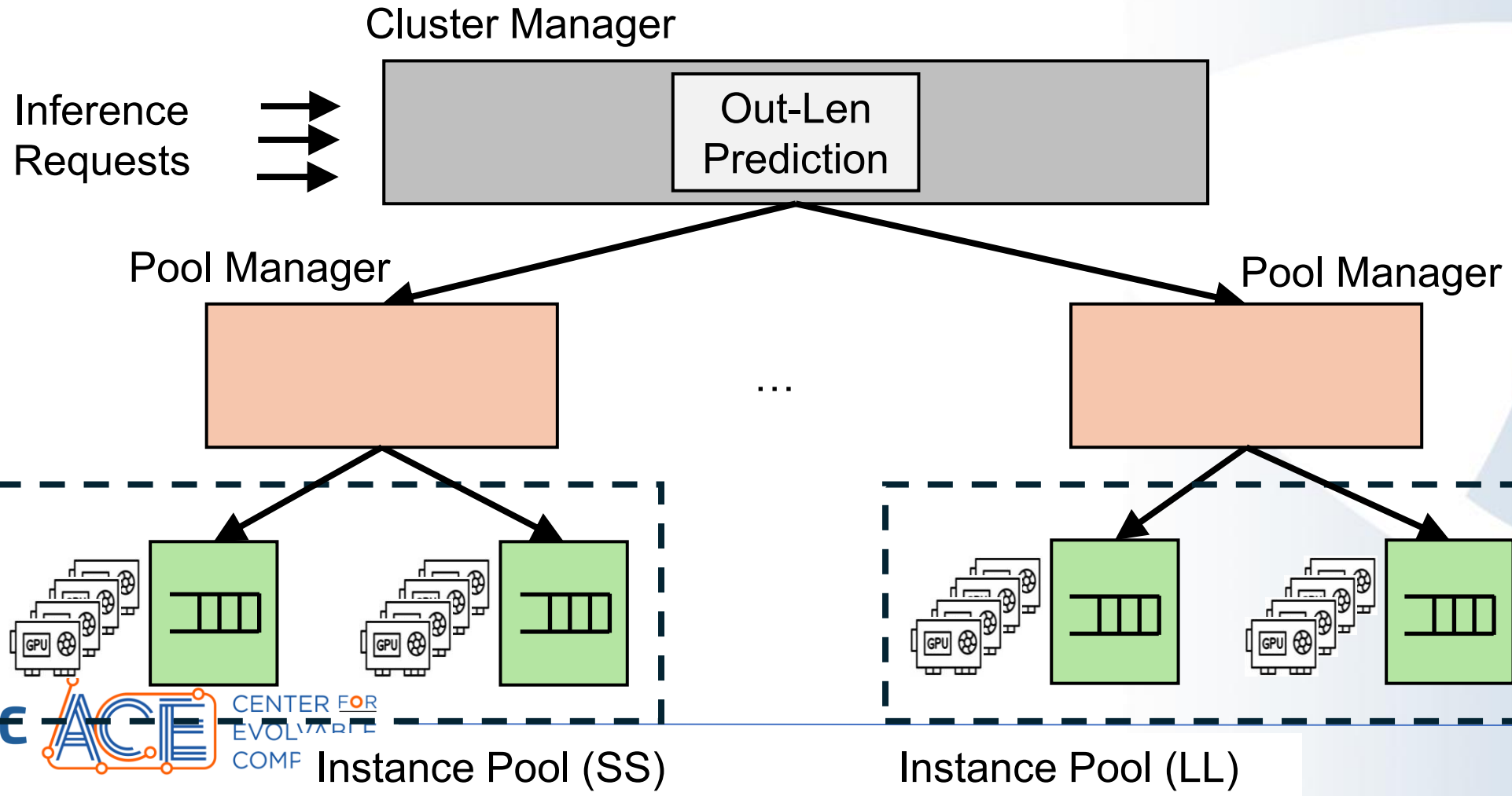
DynamoLLM:
1. Profile-Driven Energy Configuration Setting



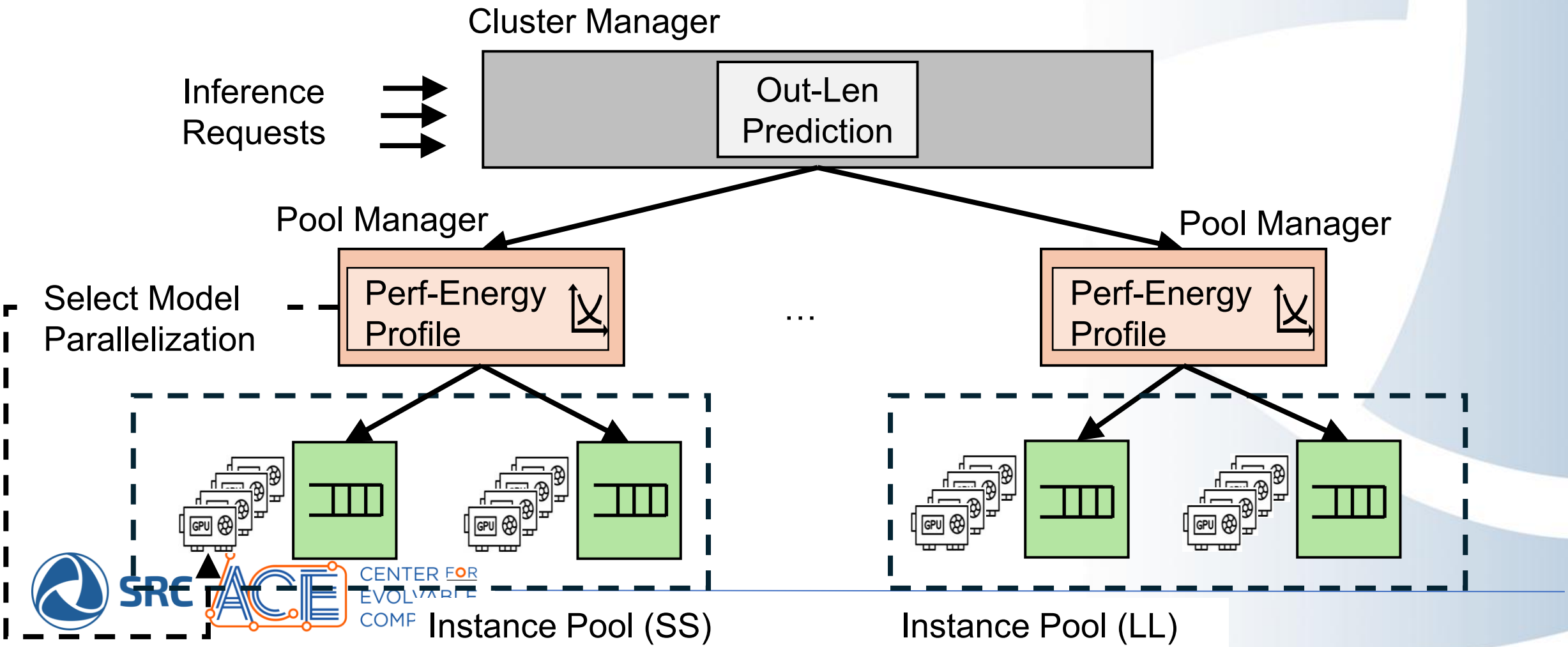
DynamoLLM:
2. Instance Pools for Diverse Workload



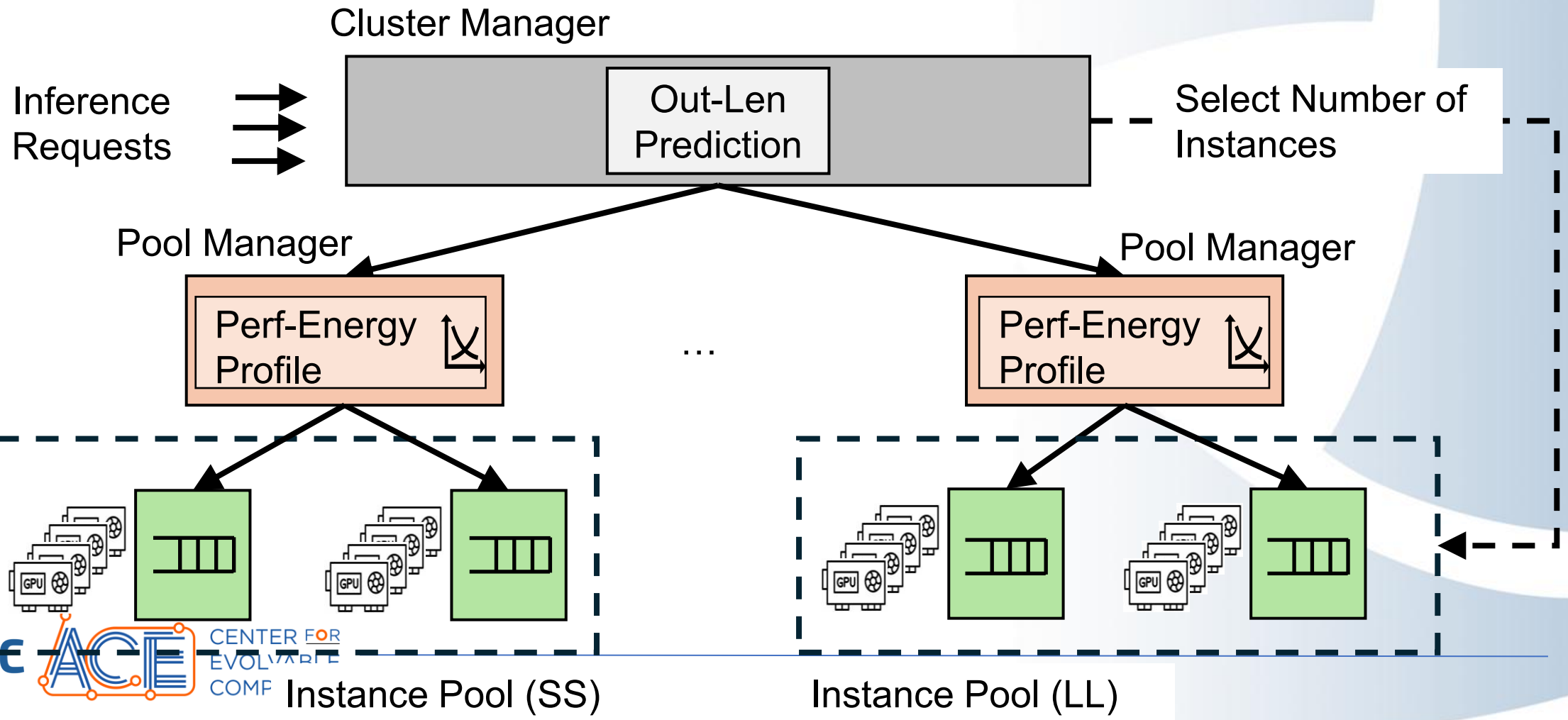
DynamoLLM:
2. Instance Pools for Diverse Workload



DynamoLLM:
2. Instance Pools for Diverse Workload



DynamoLLM:
3. Hierarchical Control for Dynamic Load



DynamoLLM:
3. Hierarchical Control for Dynamic Load