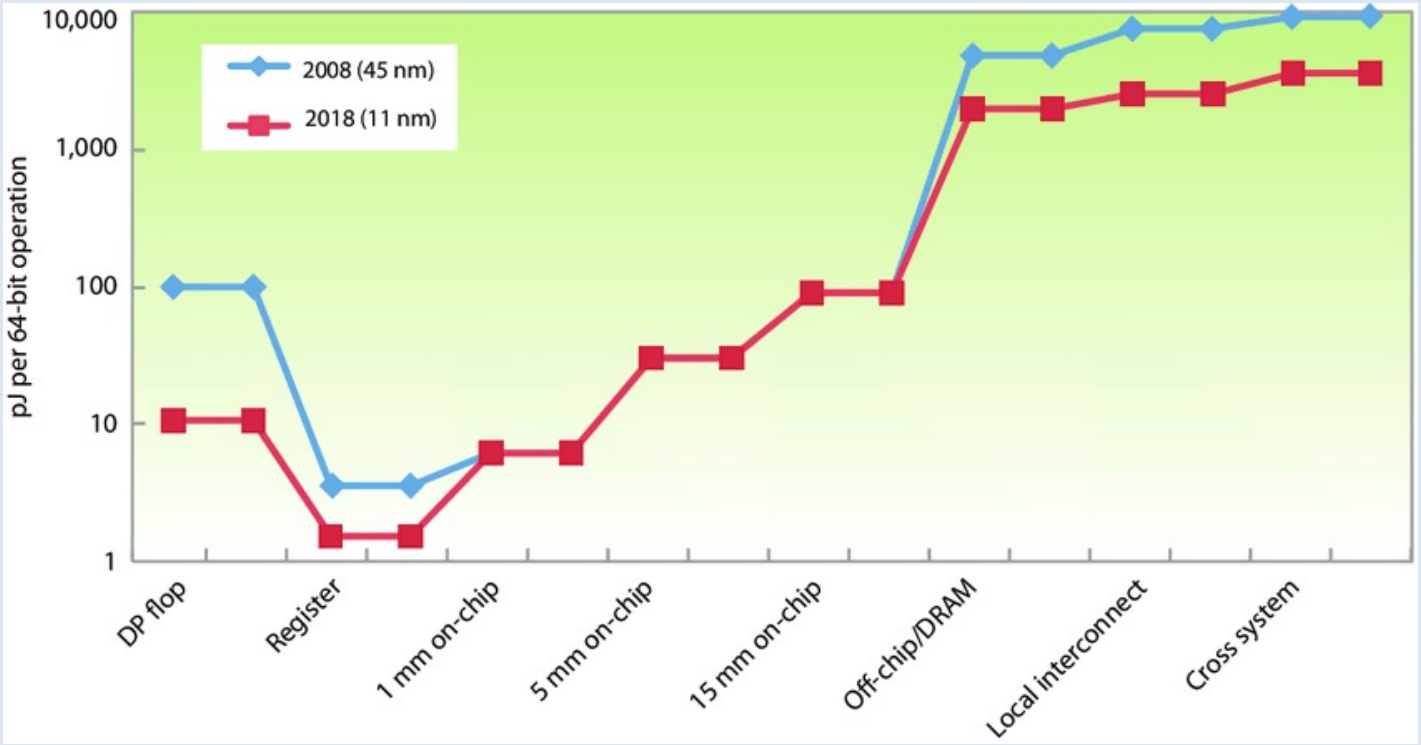
Energy-Efficient Applications are More About Data than Computation

Mary Hall
University of Utah
September, 2024

Energy Cost of Data Movement



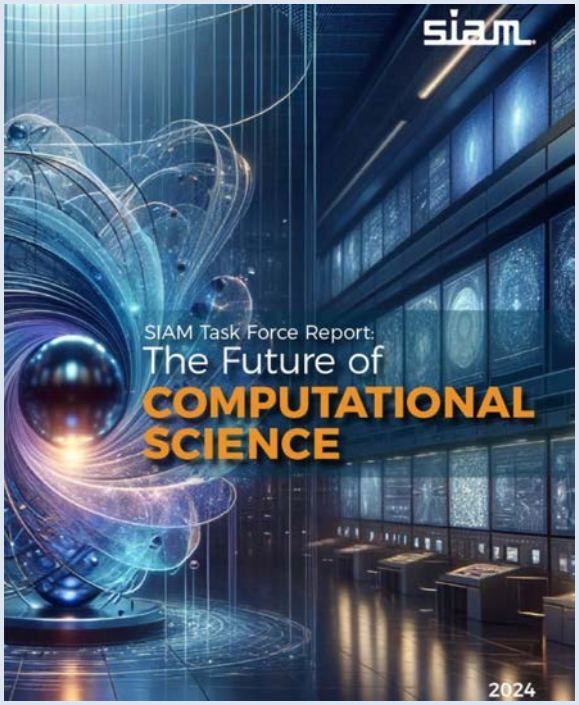
Source: P. Kogge and J. Shalf, "Exascale Computing Trends: Adjusting to the "New Normal"' for Computer Architecture," in *Computing in Science & Engineering, Nov.-Dec. 2013.*

Takeaway Message: Reduce energy costs by avoiding unnecessary data movement.

Starting Point: How Do We Know if Our Applications are Energy Efficient?

- Measuring power and energy is not a standard practice in our community
  - Instrumentation impact and what is the right granularity to measure
  - Repeatability
  - How to attribute costs
- Instead, typical proxies for energy usage
  - Time-to-solution
  - Amount of data movement at each level of the memory hierarchy

Future of Computational Science



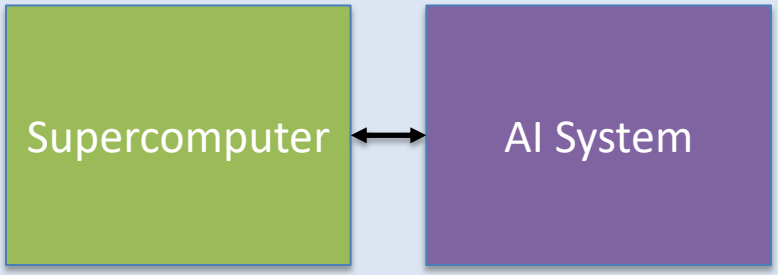
Bruce Hendrickson (Chair), *Alejandro Aceves*, *Elie Alhajjar*, *Mario Bañuelos*, *David Brown*, *Karen Devine*, *Qiang Du*, *Omar Ghattas*, *Roscoe Giles*, *Mary Hall*, *Tanzima Islam*, *Kirk Jordan*, *Lin Lin*, *Alex Pothen*, *Padma Raghavan*, *Robert Schreiber*, *Craig Thalhauser*, *Alyson Wilson*

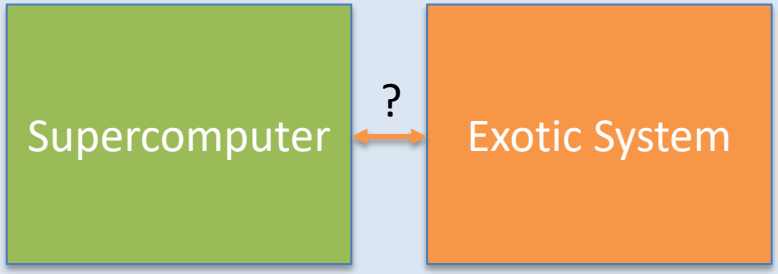
We talk about CPU+GPU systems as being heterogeneous, but they are "homogeneous clusters of heterogeneous nodes". Future supercomputers will be truly heterogeneous.
-- During the task force discussion

"The computational science community can adapt to foreseeable changes in architectures, but the lack of clarity about future machines is an enormous challenge… So, the community must develop algorithms and software today that will be capable of running on future computers, even without clarity about what those computers will look like."

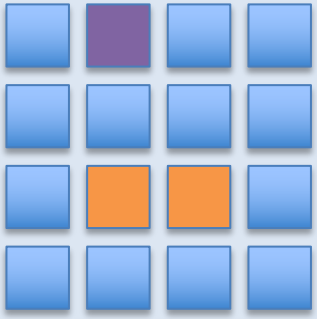
Multi-Scale Heterogeneity

System-Level Heterogeneity

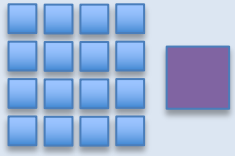




Chip-Level Heterogeneity



Heterogeneous Functional Units (e.g., Tensor Cores)



Heterogeneous Memory

Software for heterogeneous hardware

- Partition computation between different systems/tiles/units
- Marshal data to appropriate compute or storage resource

Lessons from AI Accelerators

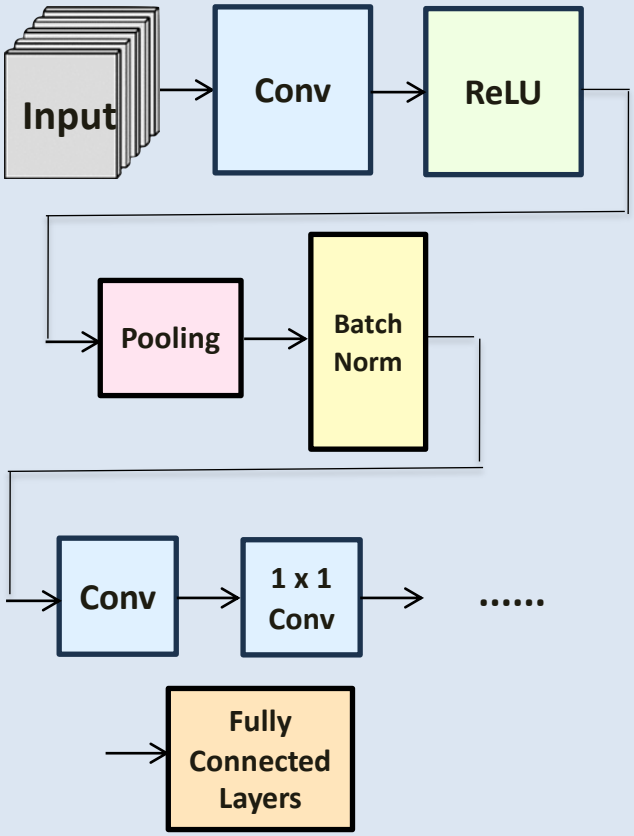
Argonne ALCF AI Testbed

Takeaway Message: Deep learning accelerators typically do not expose a more general software ecosystem.

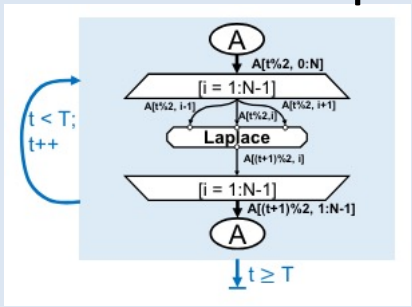
From the Future of Computational Science Report: *Recommendation 3.3: A facility or facilities should be established that will ensure that computational science researchers have access to emerging hardware, programming models, and heterogeneous systems to enable assessment of their utility for scientific applications.*

A Paradigm Shift to Dataflow

ONNX Deep Learning Model

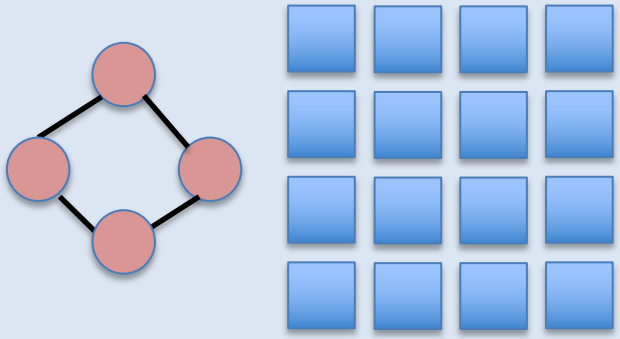


DaCe Stateful Dataflow Graph

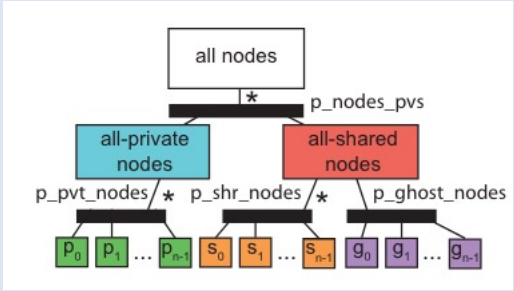


Ben-Nun et al., SC19

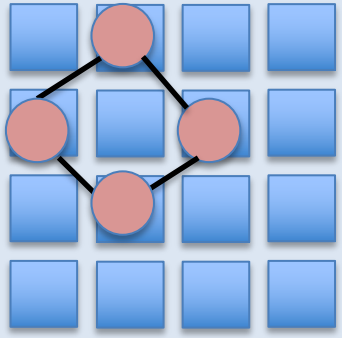
Mapping dataflow to tiles



Legion

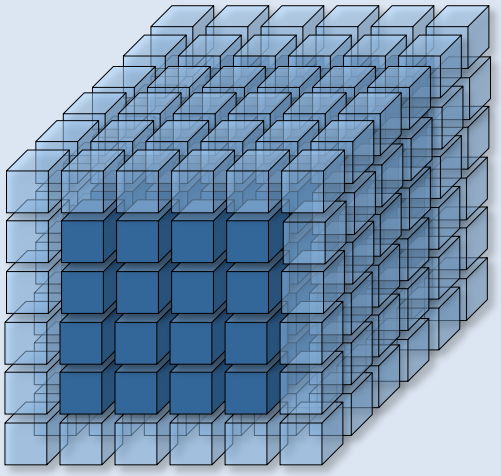


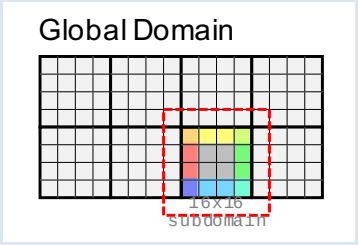
Bauer et al., SC12

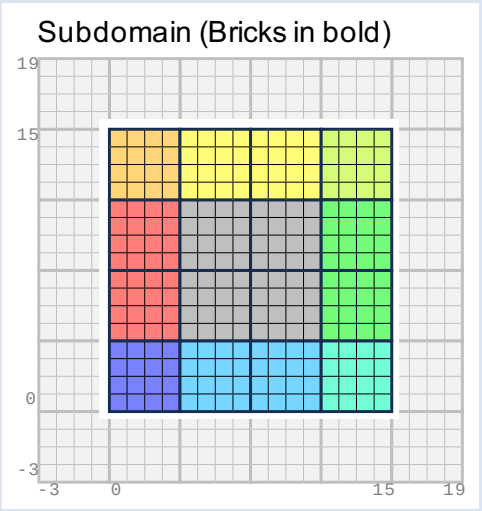


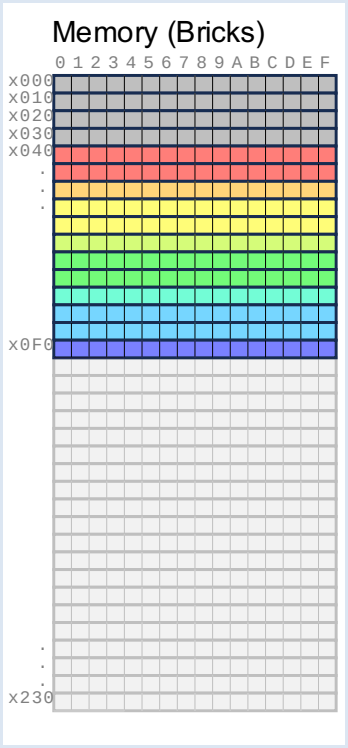
1. Data Layout/Movement











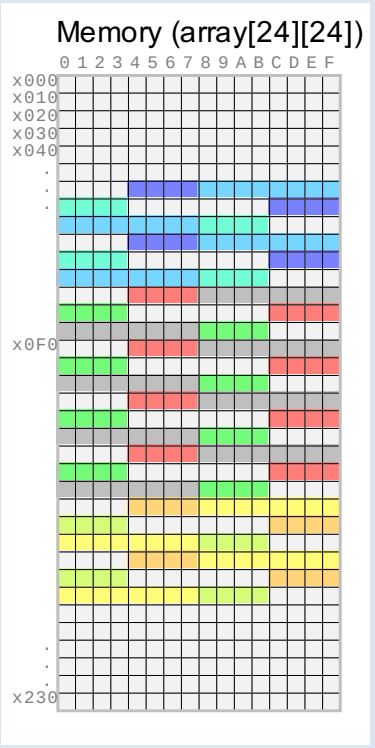
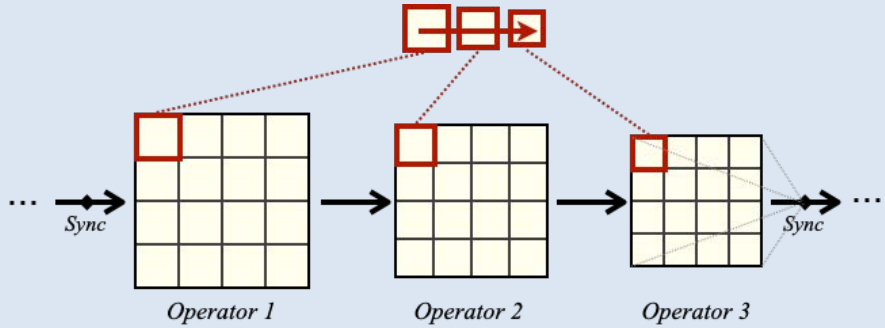


Figure Courtesy: Samuel Williams, LBNL

- Layout data in the order you want to traverse or communicate it, e.g., Bricks
- May potentially require reorganization at boundaries

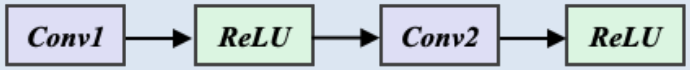
1. More Data Layout/Movement

4D Bricks for Graph-Level Optimizations of Deep Learning Workloads

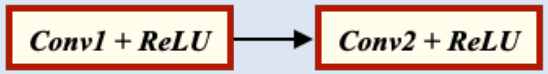


- Partition input graph into subgraphs
- Decompose feature maps into bricks
- Merge the execution of (fused) bricks across operators in each subgraph

Input Graph:



Operator Fusion:

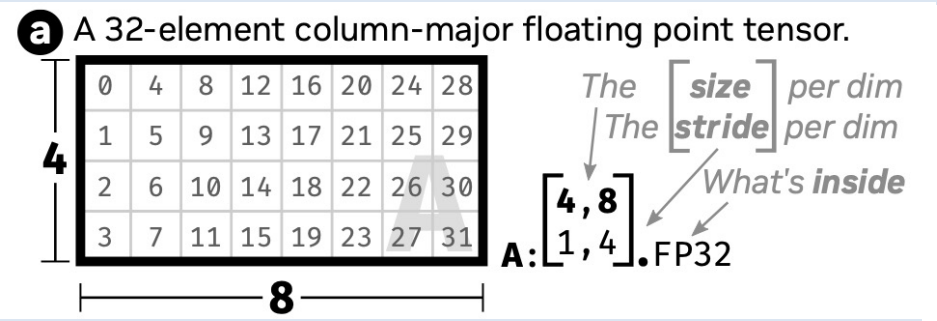


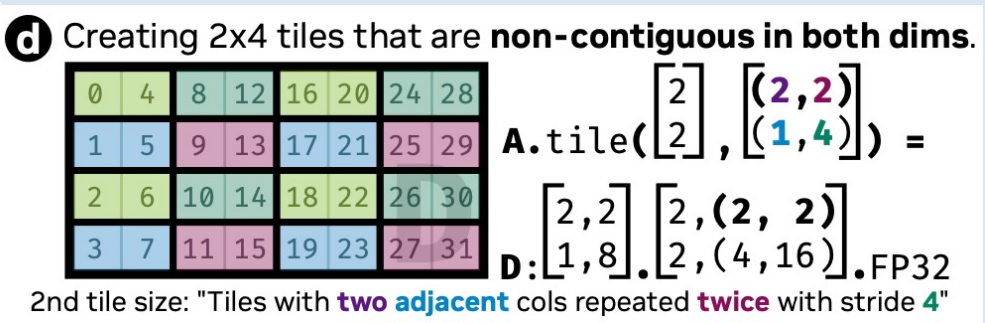
Merged Execution:

M. Lakshminarasimhan, M. Hall, S. Williams, and O. Antepara. BrickDL: Graph-Level Optimizations for DNNs with Fine-Grained Data Blocking on GPUs. In Proceedings of the 53rd International Conference on Parallel Processing (ICPP '24).

1. More Data Layout/Movement

- Approach
  - Embed data layout into compiler
  - Compose w/ domain decomposition and thread mapping
  - Requires mapping between logical and physical layout
- Example: CuTE Algebra for NVIDIA CUTLASS

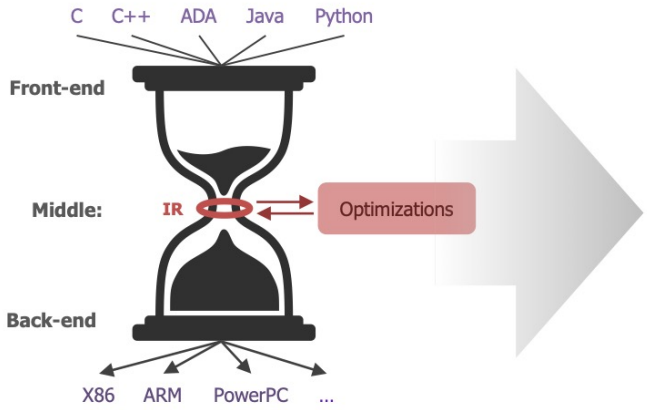




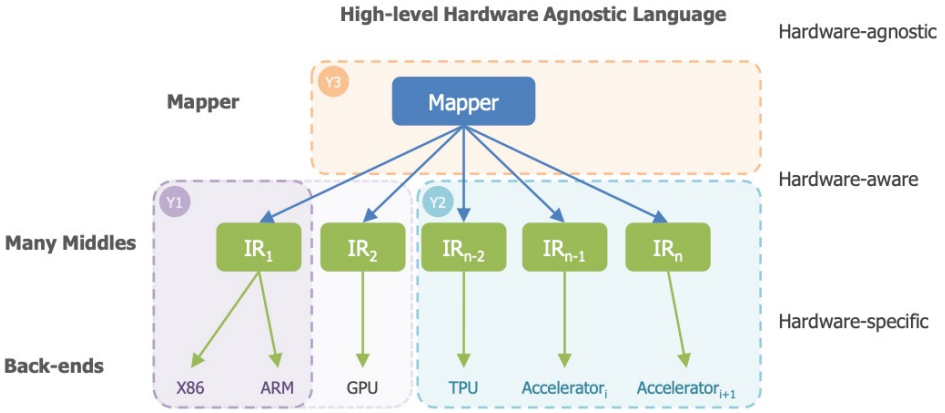
B. Hagedorn, B. Fan, H. Chen, C. Cecka, M. Garland, and V. Grover. Graphene: An IR for Optimized Tensor Computations on GPUs. In Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3 (ASPLOS 2023).

2. Compiler Structure: DARPA MOCHA BAA

Today's compiler architecture



MOCHA



Apply ML across the entire compiler pipeline

- Support multiple compute architectures simultaneously
- Enable holistic optimization across passes

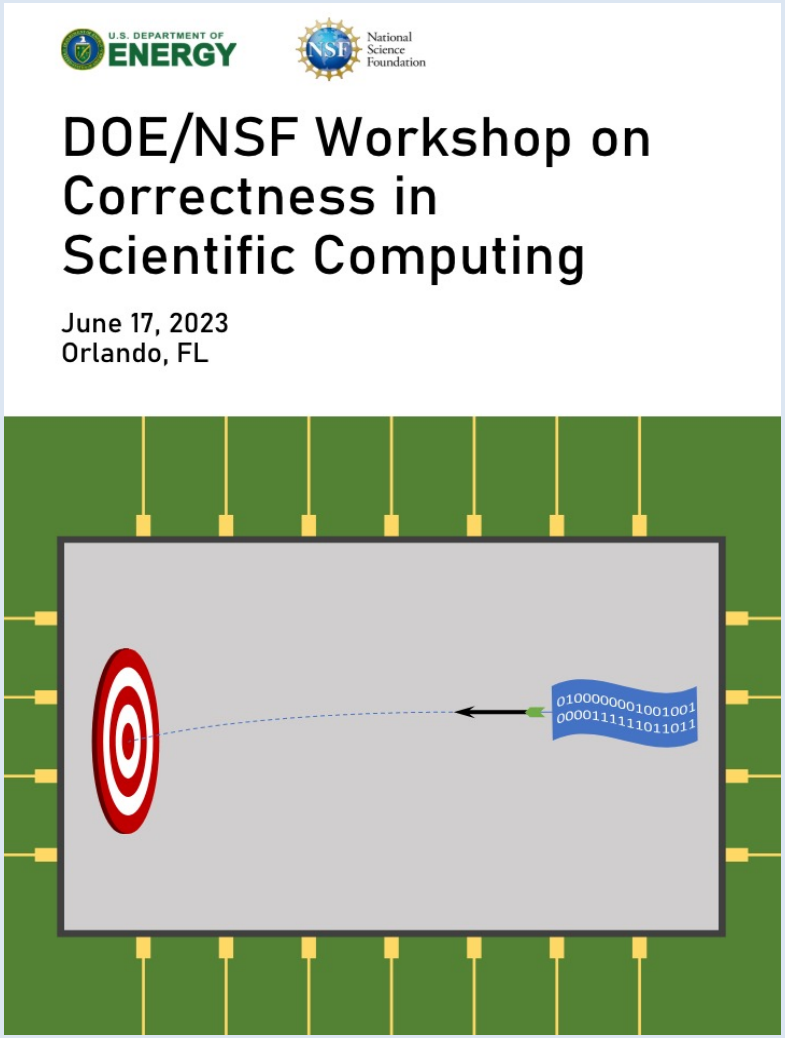
IR: Intermediate Representation
CPU: Central Processing Unit
GPU: Graphical Processing Unit
TPU: Tensor Processing Unit

https://www.darpa.mil/attachments/MOCHA%20Proposers%20Day%20Slides.pdf

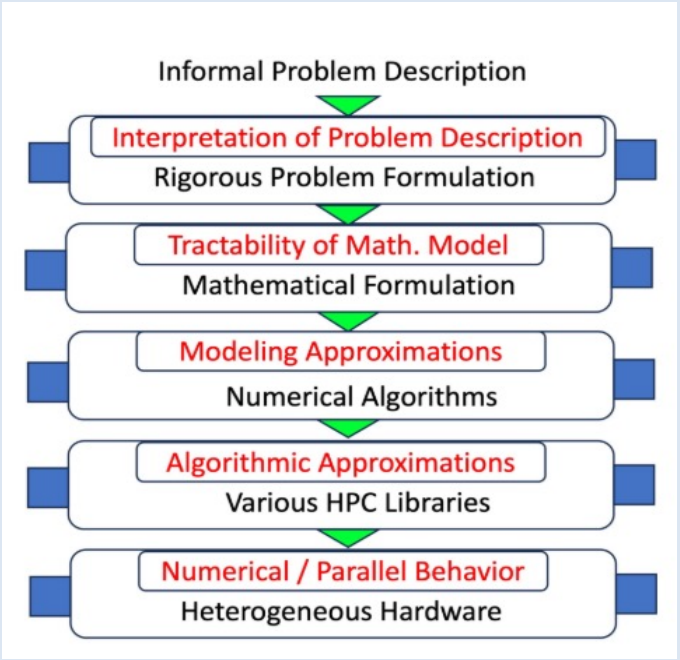
3. Produce/Integrate “Code” for New Compute Devices and Interfaces

??????????

4. Correctness



Increasing hardware heterogeneity together with partially documented hardware and libraries present serious correctness challenges.



Maya Gokhale, Ganesh Gopalakrishnan, Jackson Mayo, Santosh Nagarakatte, Cindy Rubio-Gonzalez, Stephen F. Siegel

4. Other Correctness Issues

- Lossy data compression and alternate data representations
- Sparsifying data such as selectively zeroing out weights in convolutions
- Associative reordering of computation

Path Forward, Beyond ECP Software

1. A paradigm shift making data layout and data movement a central consideration of code generation, communication, and parallel applications
2. Incorporating support for new accelerators and chiplets into compilers in a nimble way
3. New approaches to produce the “code” that will execute on fundamentally new compute devices
4. Validating the correctness of the resulting solution

Some Questions for Discussion

- Are novel architectures integrated into existing systems? What can we say about the interfaces?
- What does “code” look like for novel architectures?
- How much software remains in a co-designed architecture?
- What will it take to move the culture to measuring energy/power and software responding?