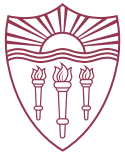


CASPER:
Compiler Abstractions Supporting high Performance on Extreme-scale Resources

Connor Imes, Alexei Colin, Naifeng Zhang,
Ajitesh Srivastava, Viktor Prasanna, John Paul Walters

ESPM2 @ Supercomputing – November 11, 2020

Goals and Challenges for HPC Application Developers

Challenges

Scale

Diversity

Heterogeneity

Dynamics

Goals

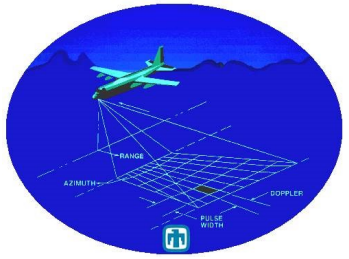
Efficiency

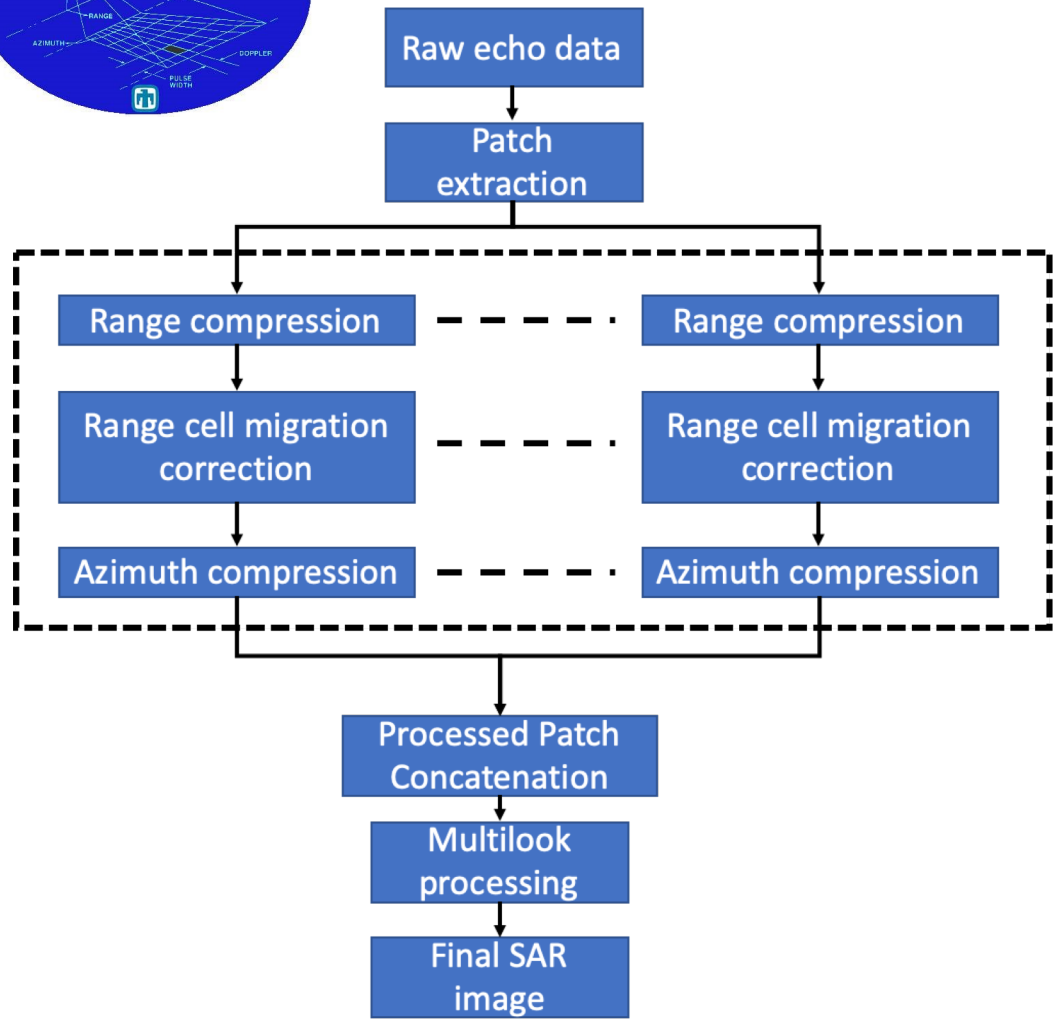
Portability

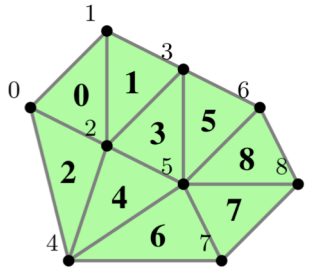
Productivity

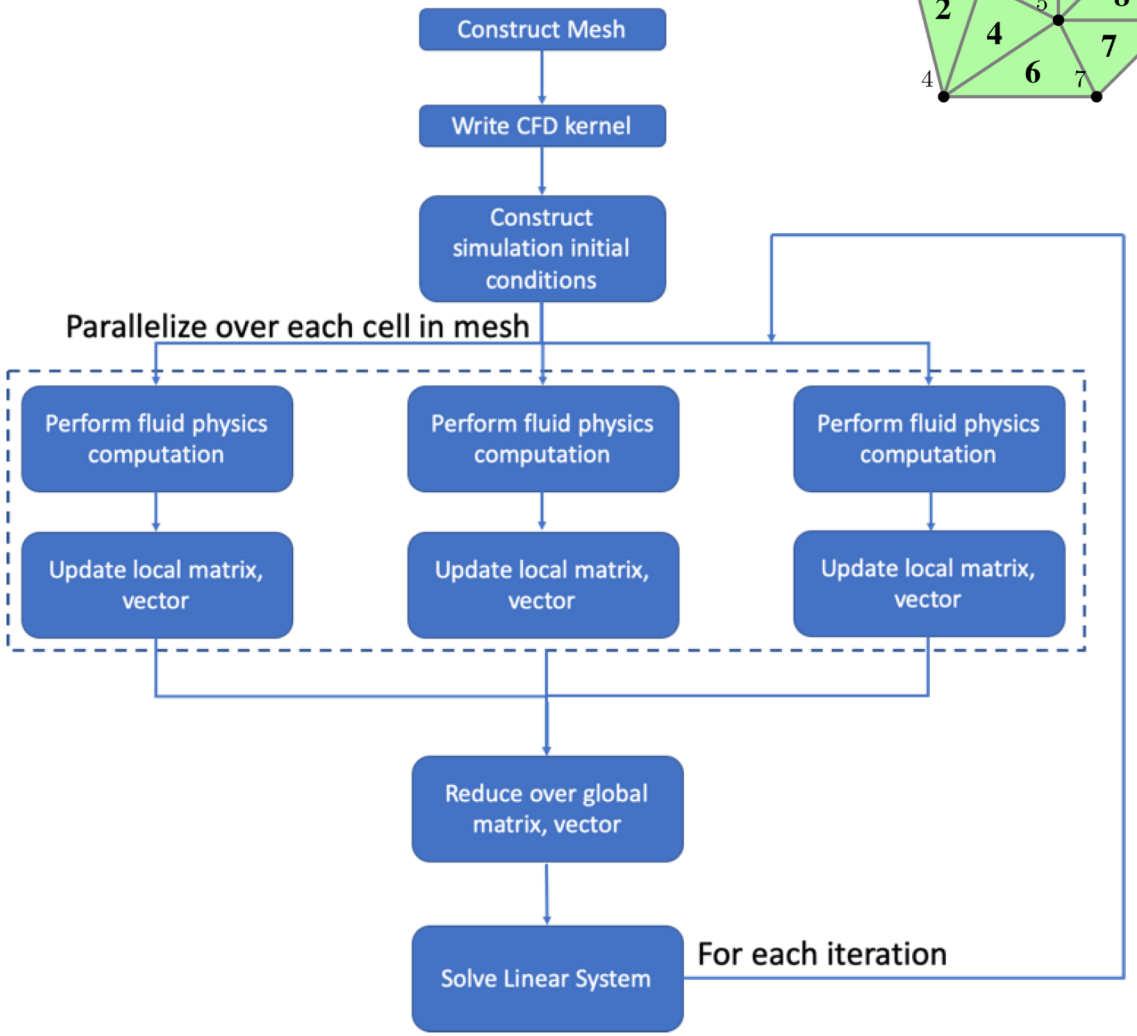
Domain Specific Languages + Extensions

Application Domains: SAR and CFD

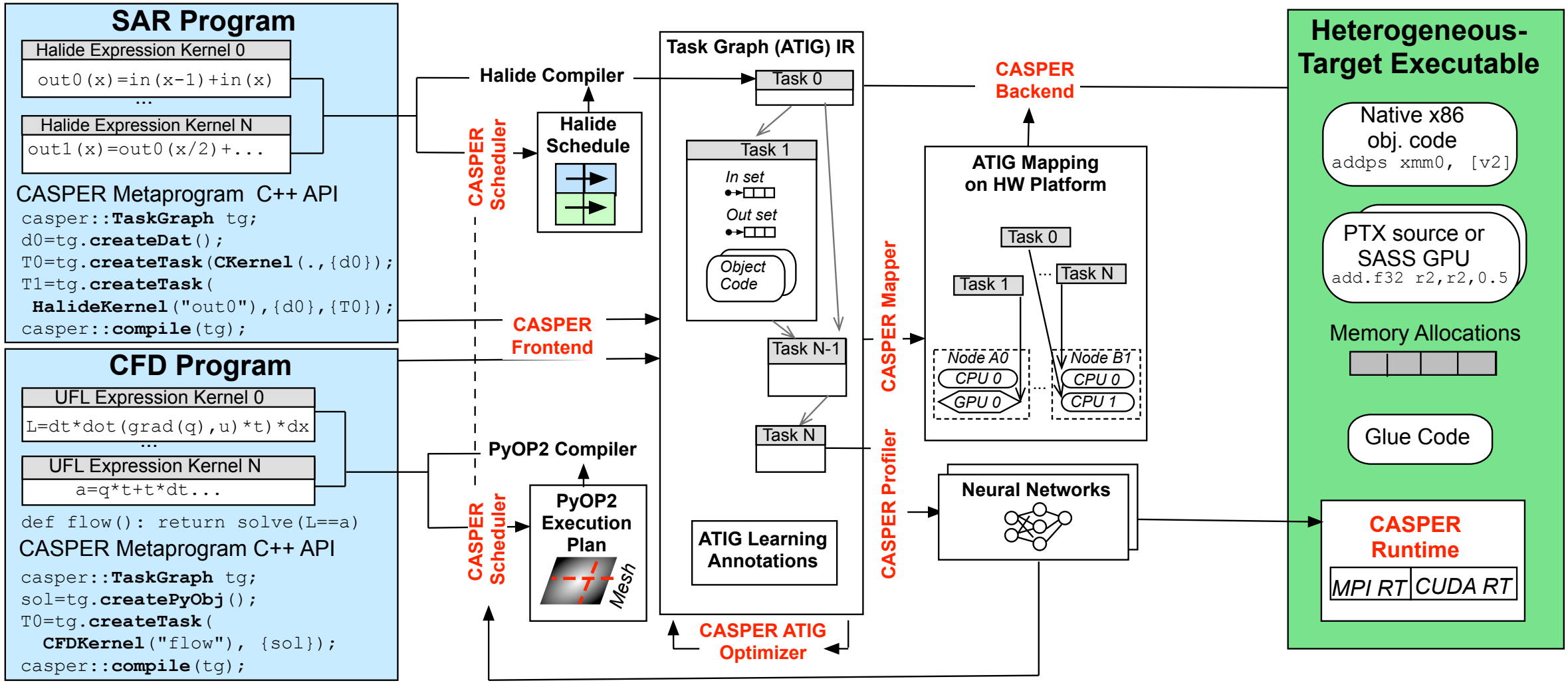








CASPER Compiler Architecture



ATIG Optimization: Variant Selection

- Implementations of tasks expose some “knobs” that can be tuned affecting performance without affecting the program output.

- **Variant Selection Problem:** For a given <task, hardware> pair, identify the set of parameters, i.e., settings of the knobs, that result in the optimal performance (execution time).
  - Need to be able to evaluate performance given the program inputs and knobs.
    - Halide: tile size, vectorization width.
    - MPI: number of processes, max threads per process.

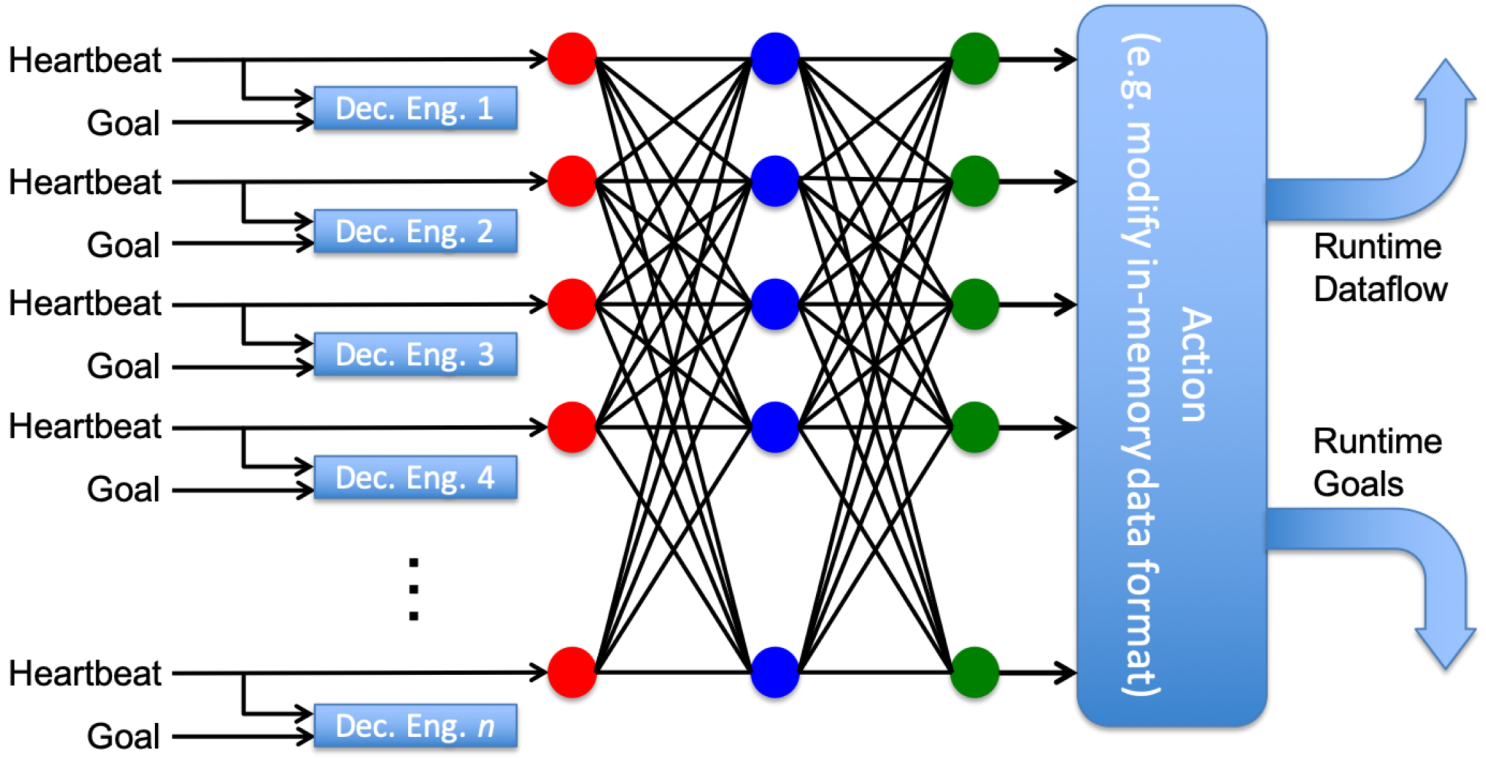
- **Performance prediction for each variant:** Our compact neural networks (< 100 weights) can accurately predict the execution time for various program inputs and knobs for given <task, hardware> pair.
  - Small size ensures fast training/retraining and fast inference at runtime.

Runtime Optimization

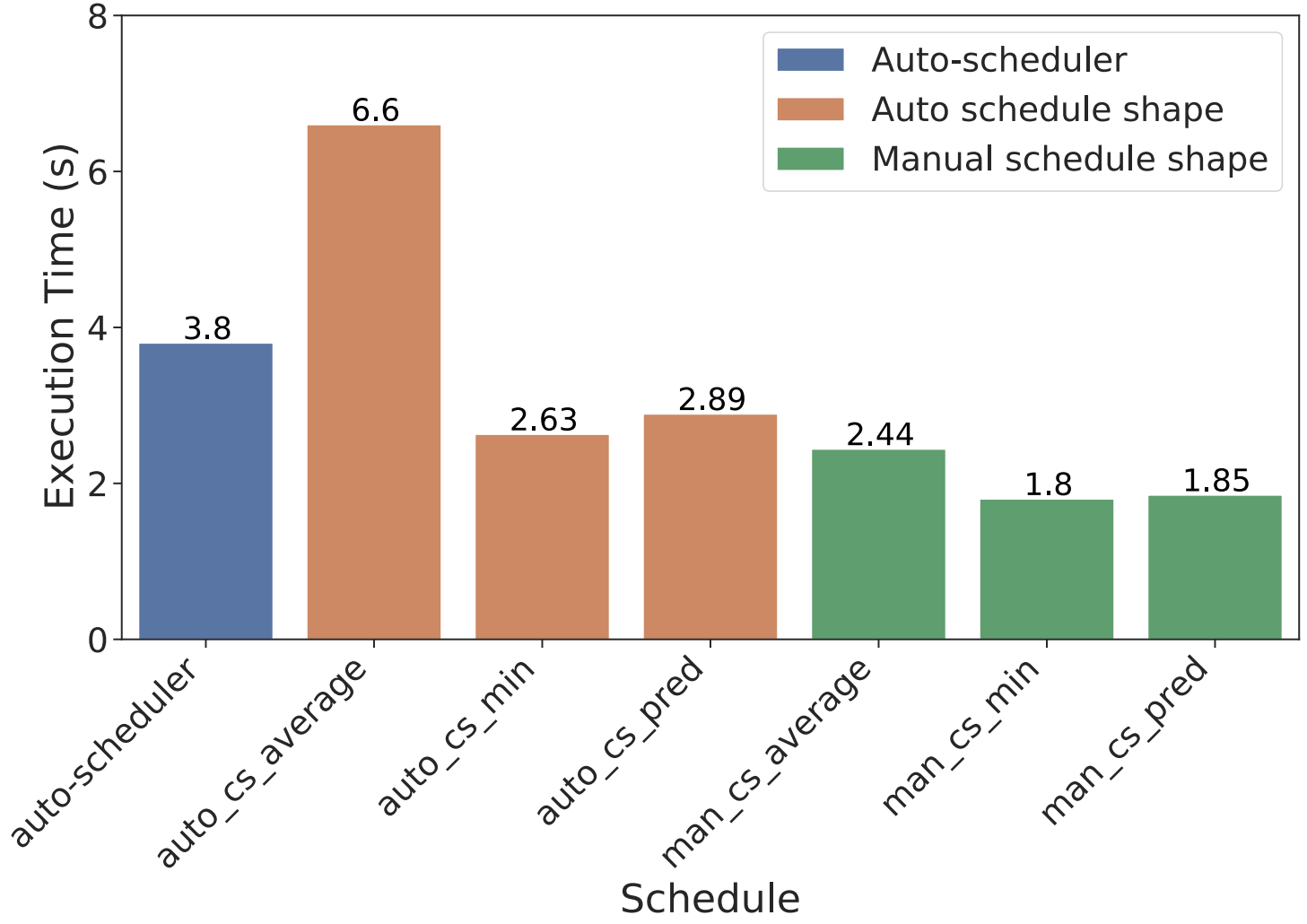
Manage application dataflow and resource utilization

Within a given scope:

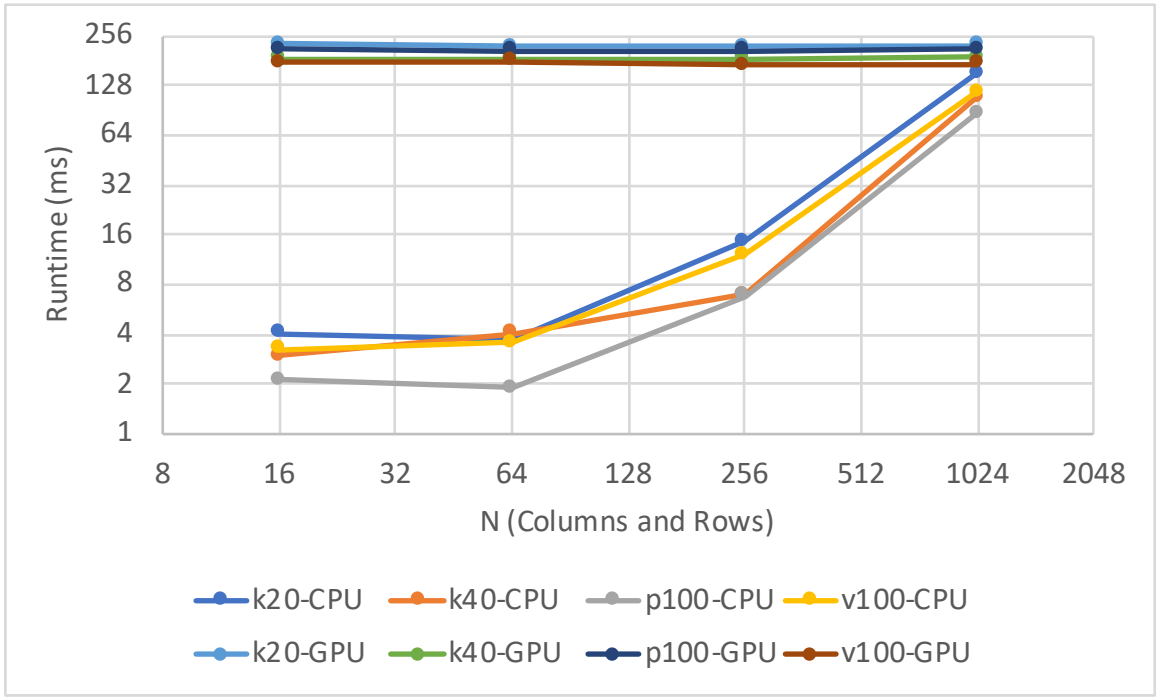
- Dataflow Remapper
  - Optimizes dataflow for current scope using Deep Reinforcement Learning
- Decision Engines [n >= 0]
  - Configured for a dataflow
  - Tune HW/SW knobs to meet goals



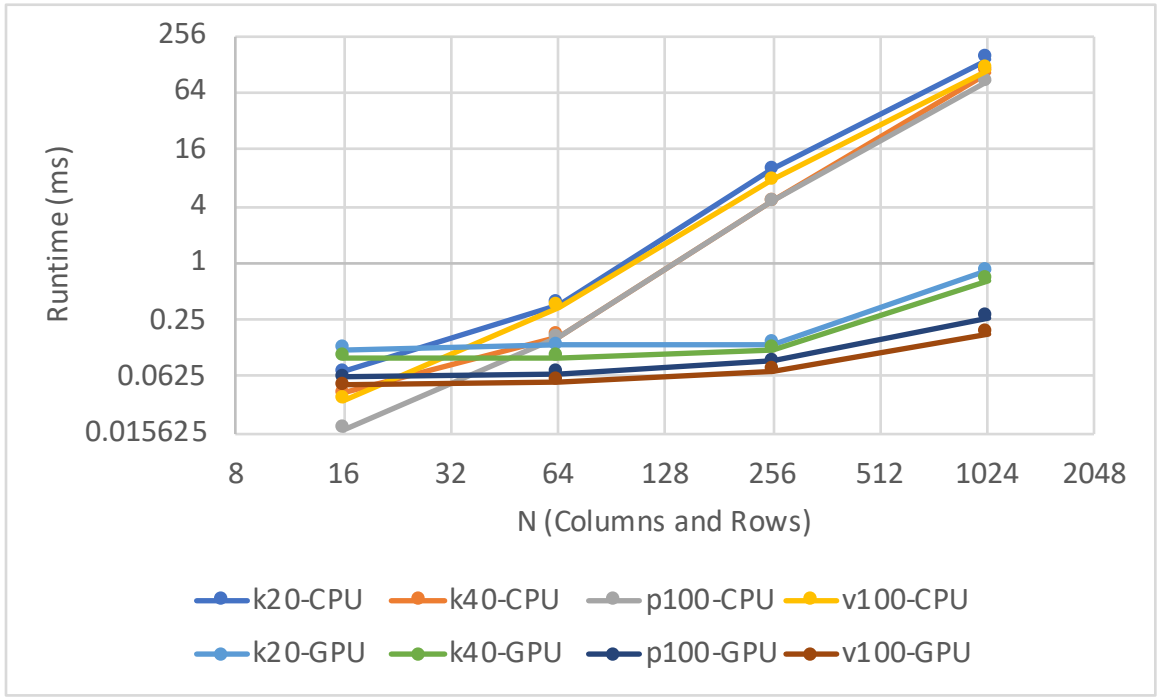
Schedule Optimization



SAR: FFTs on CPU vs GPU

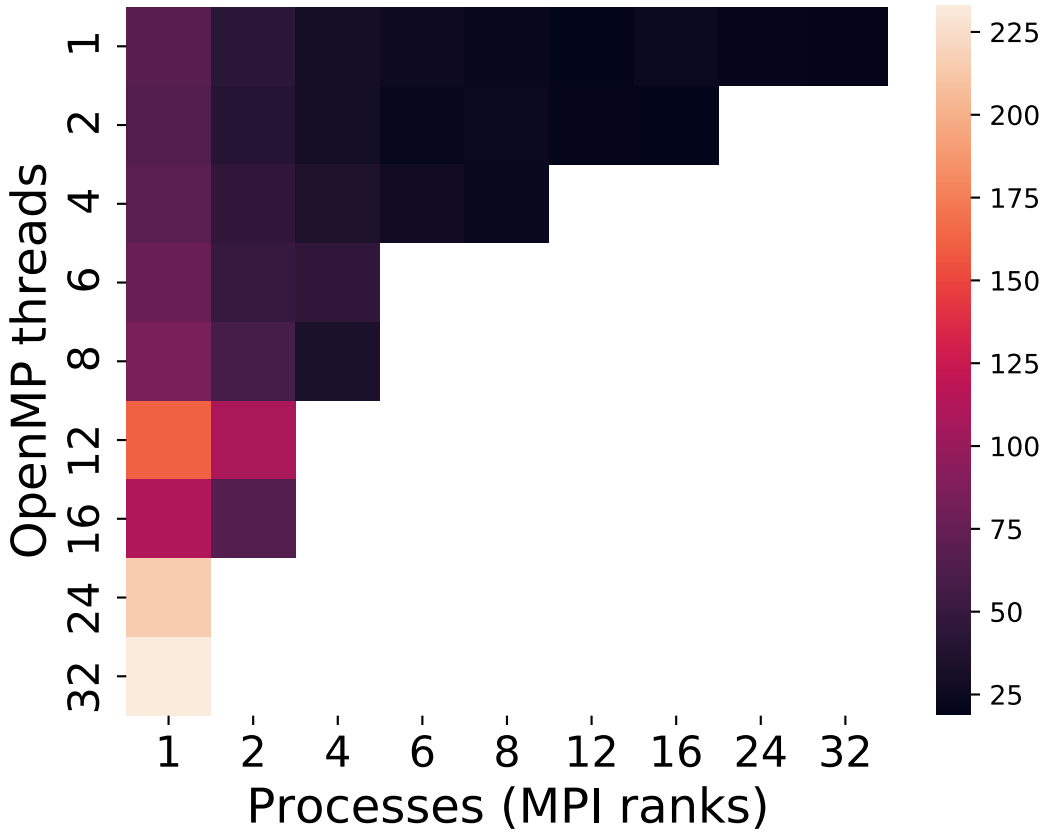


Total runtime – FFTW (CPU) vs cuFFT (GPU)

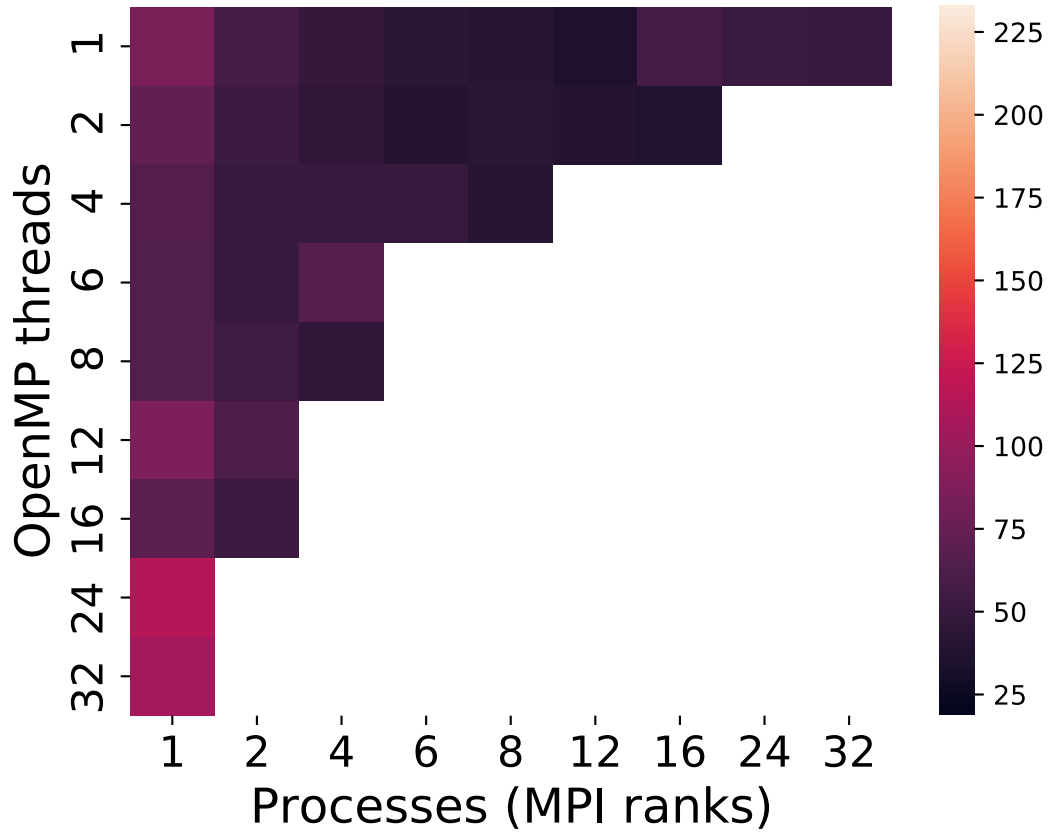


Kernel runtime – FFTW (CPU) vs cuFFT (GPU)

CFD: Process and Thread Counts



MUMPS



SuperLU_Dist

Conclusions and Summary

- CASPER is a domain-specific compiler and runtime framework to enable domain scientists to *productively* and *portably* write *efficient* and *scalable* HPC applications.
- CASPER uses Annotated Task Interaction Graphs (ATIGs) to efficiently map kernels to *diverse* and *heterogeneous* resources.
- The CASPER runtime supports adaptation in *dynamic* operating environments.
- We have demonstrated the need for CASPER with:
  - The benefits of using ATIGs to optimize resource mappings
  - Challenges in determining a priori the resources and knob settings for common SAR and CFD operations on different hardware and inputs