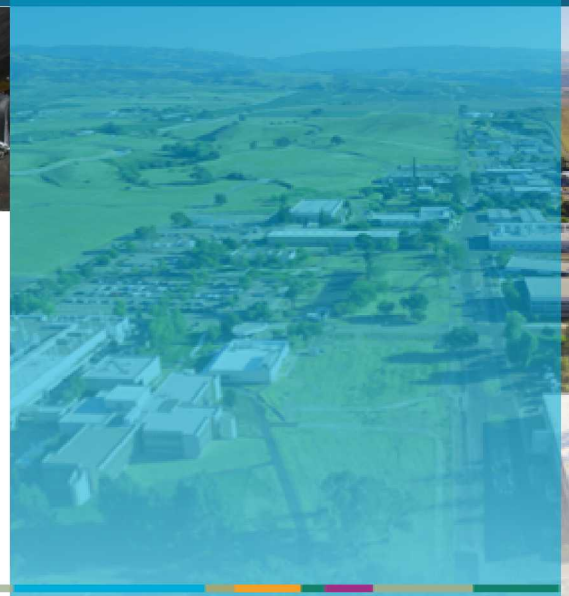
SAND2020-10300C

Sandia National Laboratories

On the performance portability of boundary conditions in Albany Land Ice



PRESENTED BY

Max Carlson

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

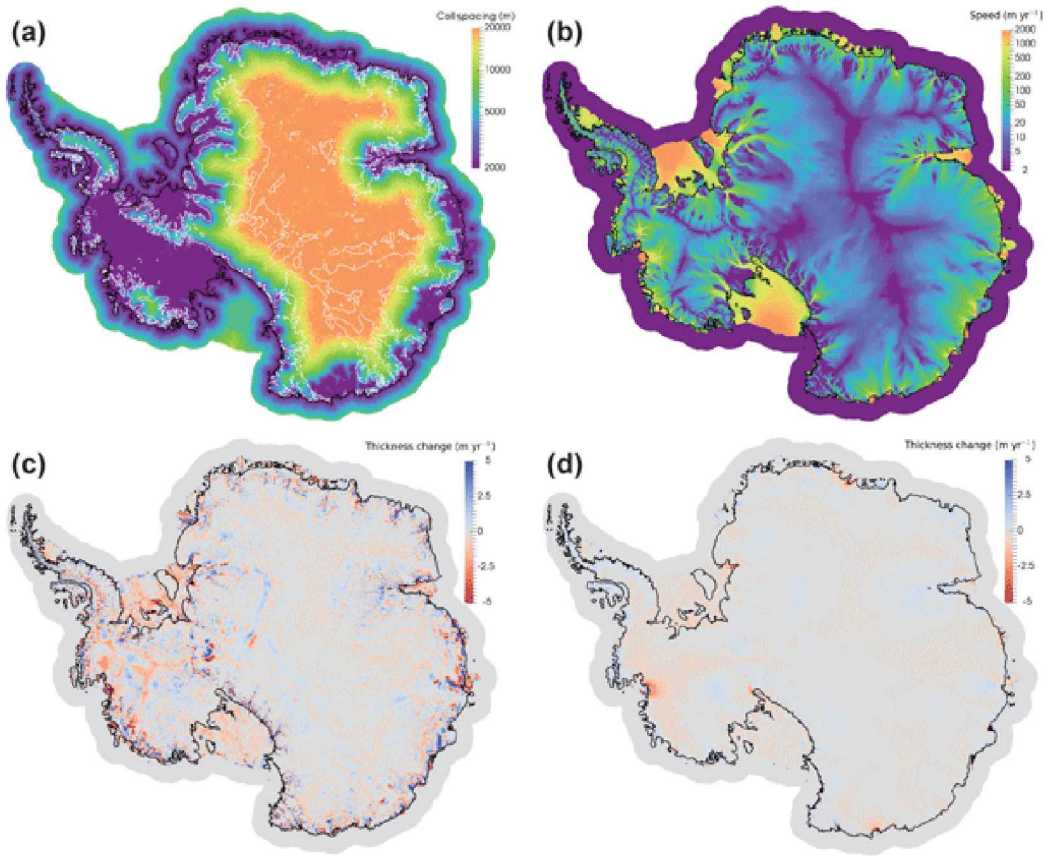
1) Albany Land Ice

2) Implementation Details

3) Performance Results

Albany Land Ice

Albany Land Ice



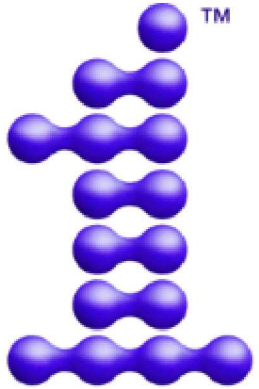
- Land ice modeling using first-order approximation of Stokes flow for glaciers and ice sheets
- Ice sheet velocity is modeled as system of steady state equations
- The velocity equations are coupled to dynamic equations for ice thickness and temperature
- ALI interfaces with E3SM through the MPAS framework (MALI)

Modernizing Albany Land Ice

- One of the ongoing goals of the Albany Land Ice project is to run the solver on high end GPU computing clusters such as Summit
- Work had been done previously to port the velocity problem assembly for GPUs
- This summer, we extended this work to the enthalpy problem assembly and the boundary condition evaluation
- While the main goal of this project is to get the fully coupled problem working on GPUs, it is also the goal to achieve a high degree of performance portability

Performance Portability

- Improvements to raw processing power has slowed considerably, giving rise to specialized computing architectures such as GPUs, FPGAs, and some CPUs
- Writing GPU code requires learning an entirely new programming model and maintaining a performant GPU code in parallel with CPU code is infeasible
- Domain experts shouldn’t have to be exposed to performance concerns



oneAPI™

hipSYCL

RAJA

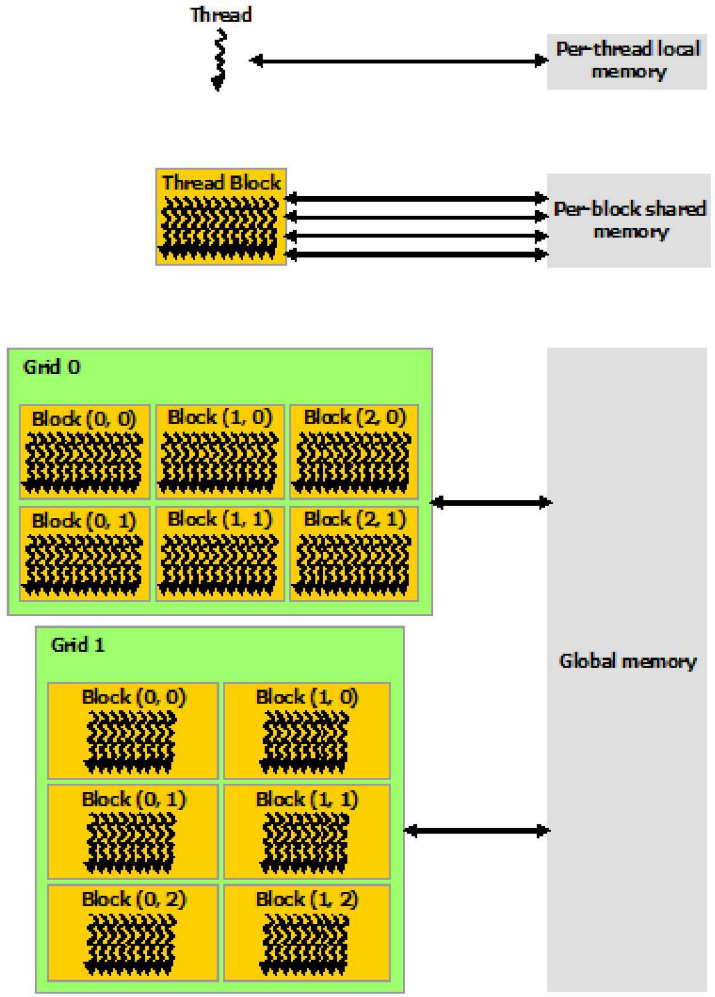
GPU Performance – Memory Hierarchy

- Device memory is further broken up into global, shared, and local memory

- Moving data from global memory into share or local memory typically is an expensive operation

- Thread blocks are designed to load data from global memory in contiguous blocks

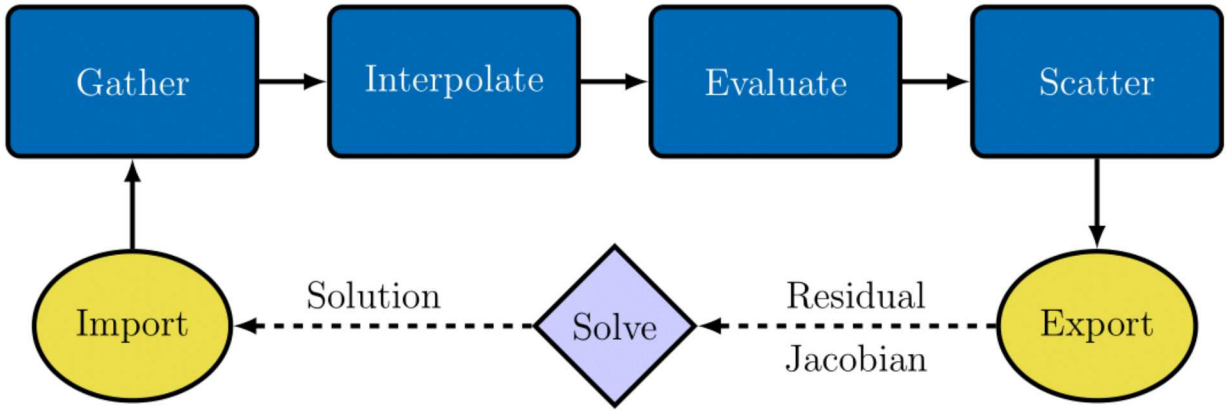
- Efficient access of global memory is referred to as “coalesced”



Implementation Details

Albany

- Albany is an object-oriented, parallel, C++ code for discretizing and solving PDEs
- Uses finite element method on unstructured grids
- Utilizes a number of libraries from the Trilinos project
- Albany constructs a system and then hands it to Trilinos to solve



Volume Refactor

- Intermediate data structures converted to Kokkos views for accessibility on device
- Evaulators were parallelized across number of cells in a workset
- Volume evaluators were either completely data parallel, or at least data parallel with respect to a cell
- Readability was preserved for ease of future implementation

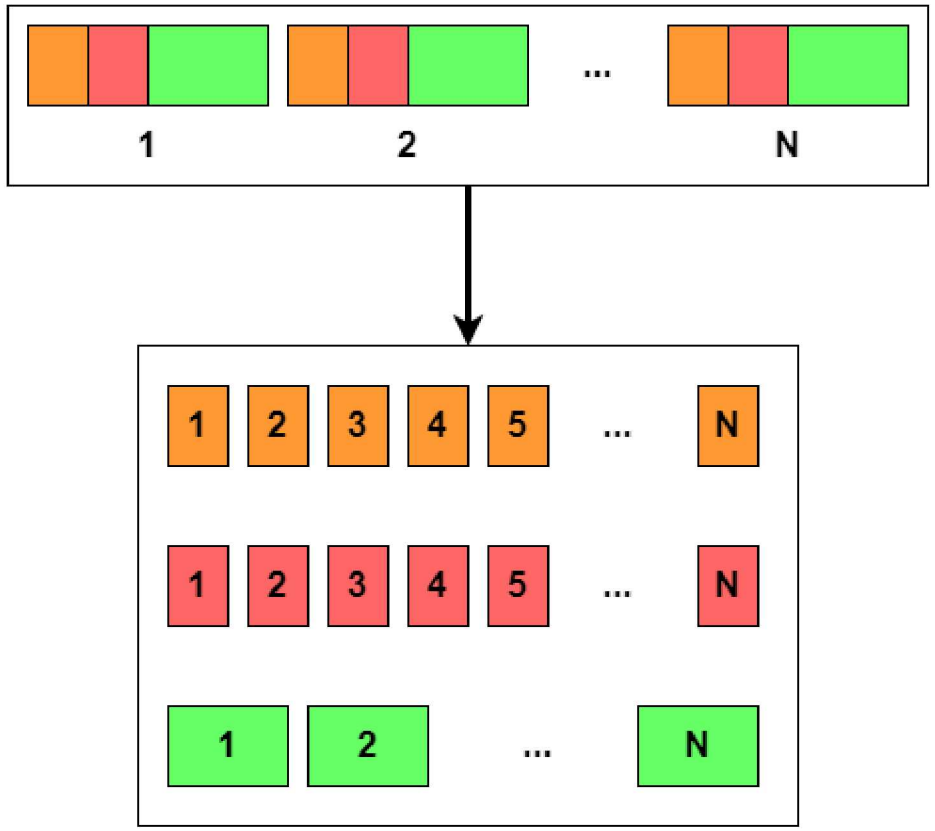
```
void Dissipation<EvalT,Traits>::
evaluateFields(typename Traits::EvalData workset) {
    for (std::size_t cell = 0; cell < workset.numCells; ++cell)
        for (std::size_t qp = 0; qp < numQPs; ++qp)
            diss(cell,qp) = 1.0/scyr * 4.0 * mu(cell,qp) * epsilonSq(cell,qp);
}
```

```
KOKKOS_INLINE_FUNCTION
void Dissipation<EvalT,Traits>::
operator() (const int &cell) const {
  for (int qp = 0; qp < numQPs; ++qp) {
    diss(cell,qp) = 1.0/scyr * 4.0 * mu(cell,qp) * epsilonSq(cell,qp);
  }
}

void Dissipation<EvalT,Traits>::
evaluateFields(typename Traits::EvalData workset) {
  Kokkos::parallel_for(Dissipation_Policy(0, workset.numCells), *this);
}
```

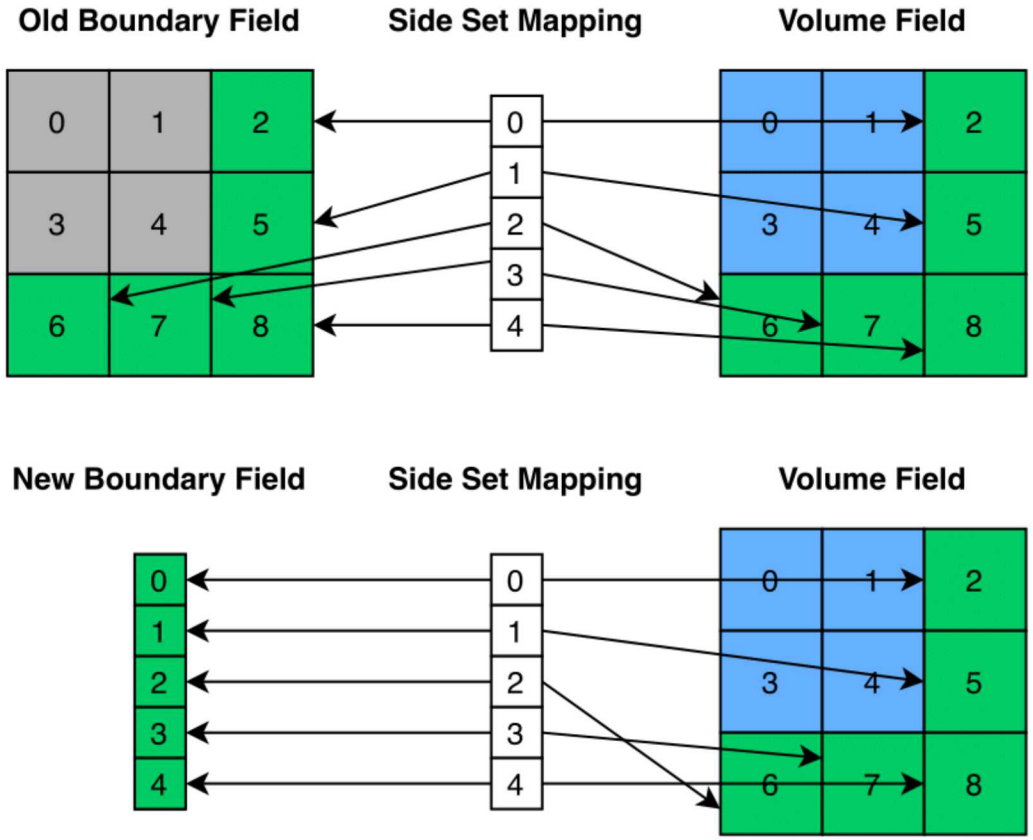
Boundary Condition Refactor

- Boundary index information originally stored as array of structures
- Replaced with structure of Kokkos views for device access
- This structure enables coalesced access of boundary information
- Boundary fields are not coalesced



GPU-Friendly Boundary Data

- Boundary fields had the same layout as volume fields and were accessed using the side set mapping
- Matching the boundary field layout to side set mapping layout results in coalesced access for both
- Access to the volume field is not coalesced but happens only once instead of many times



GPU-Friendly Boundary Data (continued)

- Albany separates work into smaller worksets when problem size is very large
- In order to handle this, the boundary mapping structure was split into a global structure and a local structure.
- The global structure contains all boundary data for all worksets
- The local structure contains Kokkos subviews specific to a given workset
- This also helps avoid a performance corner case when initializing a mesh by minimizing Kokkos view initializations

Performance Results

Experiment Setup

- We solved the enthalpy problem on a variable resolution (1k to 10km) mesh of the Greenland ice sheet
- The following node configurations were used for performance testing
  - 4x Nvidia V100 GPUs per node
  - Dual-socket POWER9 CPU with 40 cores per node (20 cores per socket)
  - Dual-socket Haswell CPU with 32 cores per node (16 cores per socket)
  - Single-socket Knight’s Landing CPU with 64 cores per node
- Three branches of code were profiled: Original, Volume, Volume+BCs
- Each branch of code was built using either CUDA, OpenMP, or Serial Kokkos backends

Performance Speedup

	CUDA-original	CUDA-volume	Speedup
Total Fill Time	183.12 seconds	3.8 seconds	48.2x

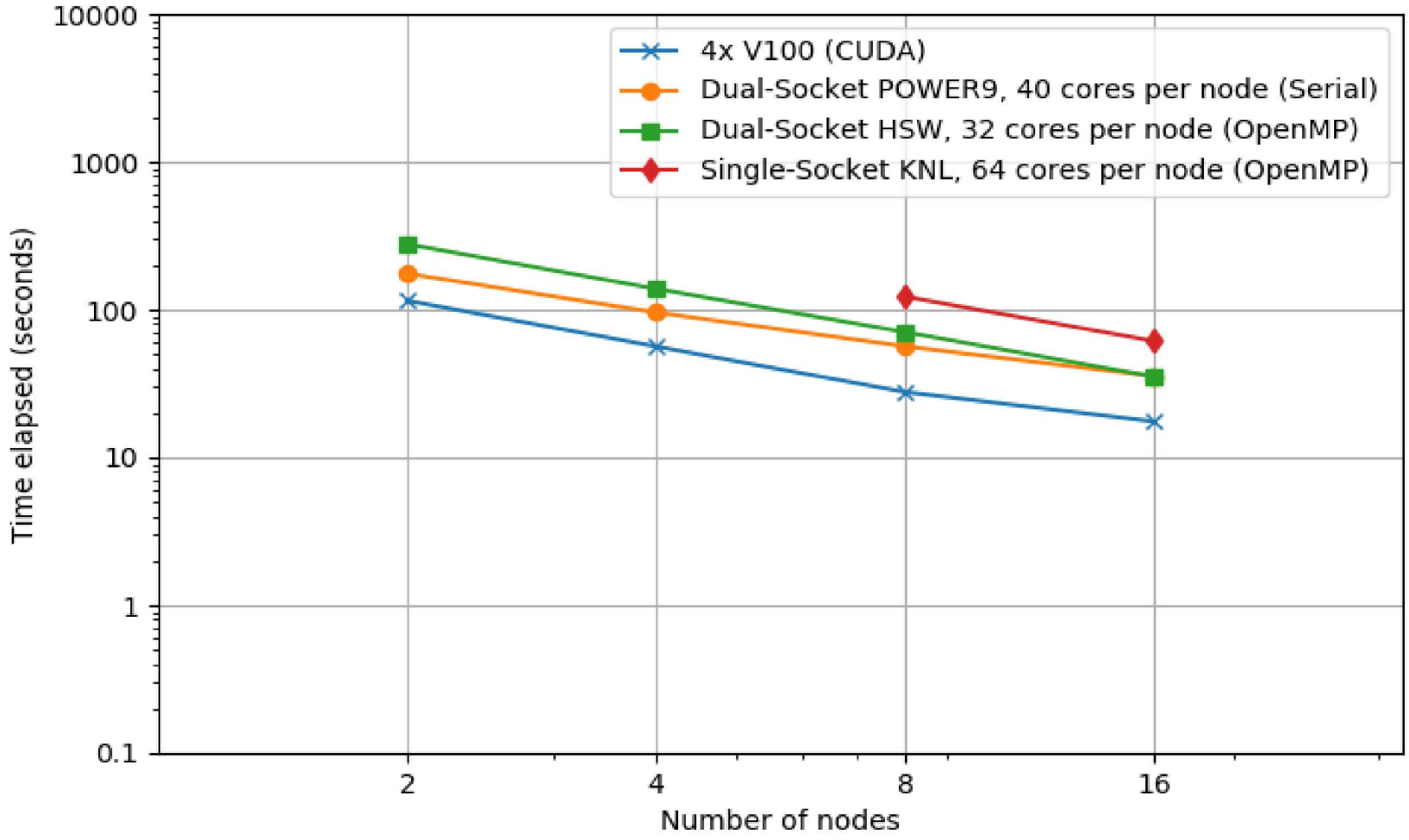
	Serial-original	CUDA-volume	Speedup
Total Fill Time	16.75 seconds	3.8 seconds	4.4x

	Serial-original	CUDA-BCs	Speedup
Total Fill Time	16.75 seconds	2.79 seconds	6x

	KNL-original	KNL-volume	Speedup
Total Fill Time	28.12 seconds	20.14 seconds	1.4x

	Serial-original	KNL-volume	Speedup
Total Fill Time	32.4 seconds	20.14 seconds	1.6x

Strong Scaling



Volume Evaluator Profiling

(Placeholder) Final presentation will have table with volume evaluator performance profiling measurements

- Volume evaluators are bandwidth bound when problem size per MPI rank is sufficiently large, become latency bound as number of nodes increases

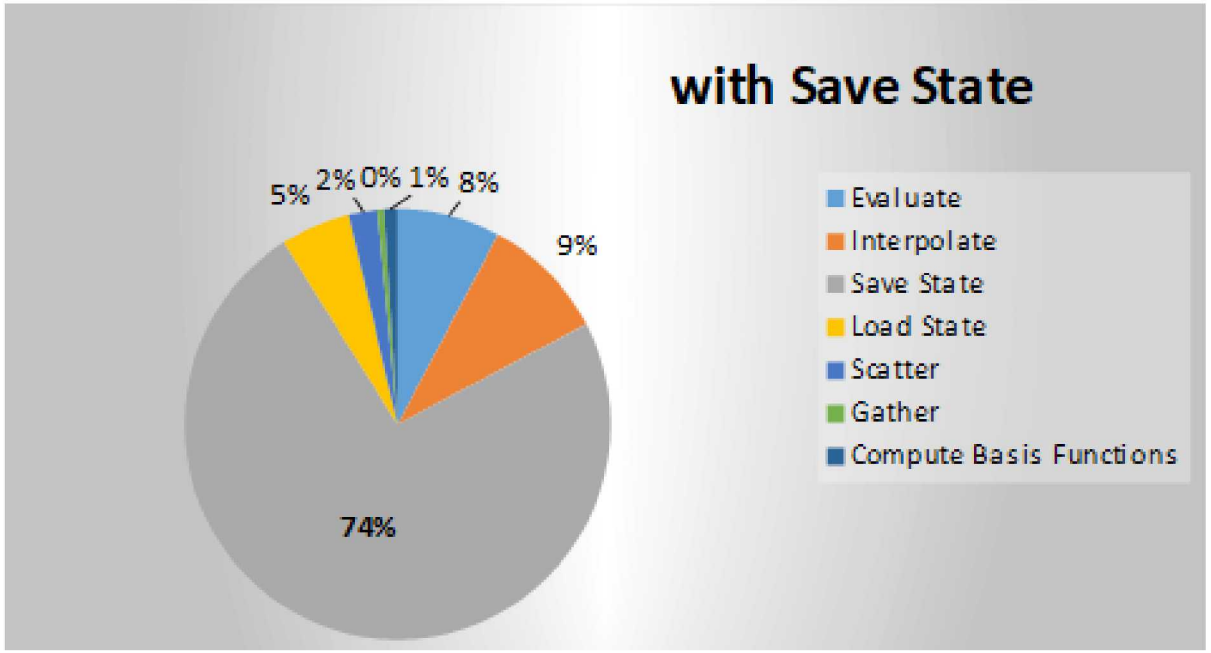
Boundary Evaluator Profiling

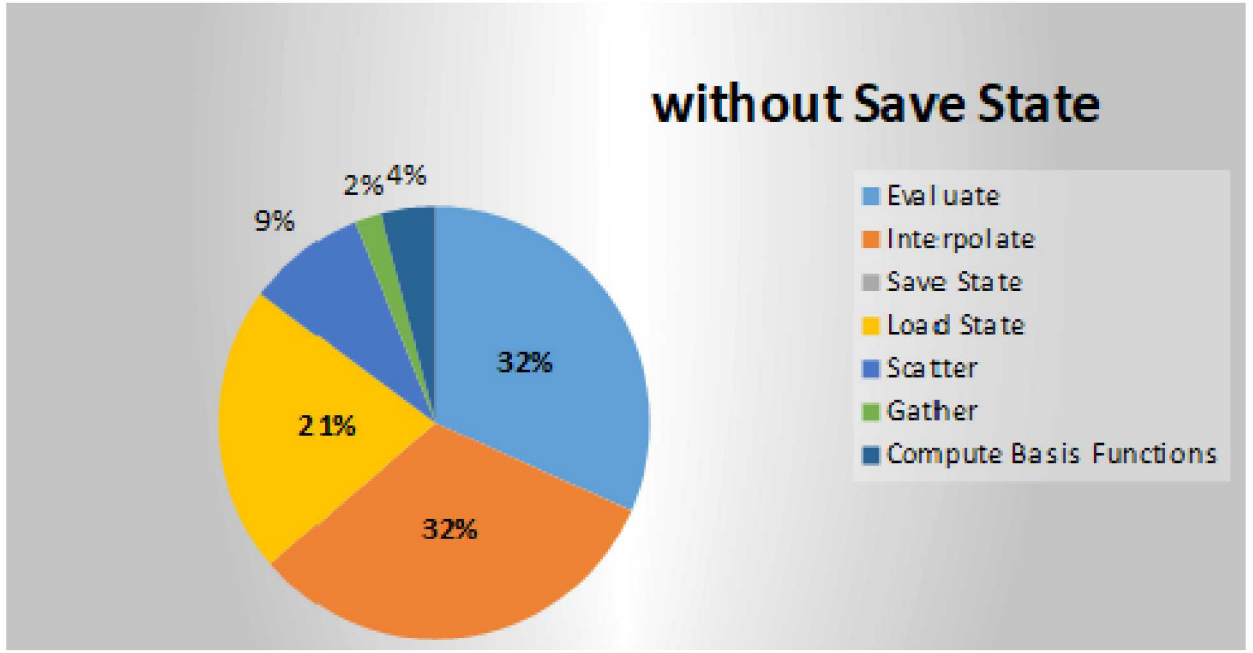
(Placeholder) Final presentation will have table with boundary condition evaluator performance profiling measurements

- Boundary condition evaluators are, as expected, latency bound due to small volume of data to be processed. Regardless, there is still performance improvement over host-side processing

Performance – Timing Breakdown

- Save and Load State categories are evaluators that prepare data for I/O operation, can't be ported to GPU, but can potentially be made unnecessary
- Evaluate and Interpolate are the categories that needed to be ported to GPU for this problem





Future Work

- The work done for the enthalpy boundary condition evaluation is going to be extended to the velocity problem

- Utilize hierarchical parallelism or CUDA/Kokkos graphs to schedule multiple latency bound kernels to run concurrently

- Solving the system still needs work to be fully supported by GPUs. This involves porting of code in Trilinos and will be the next major step towards full end-to-end GPU solves