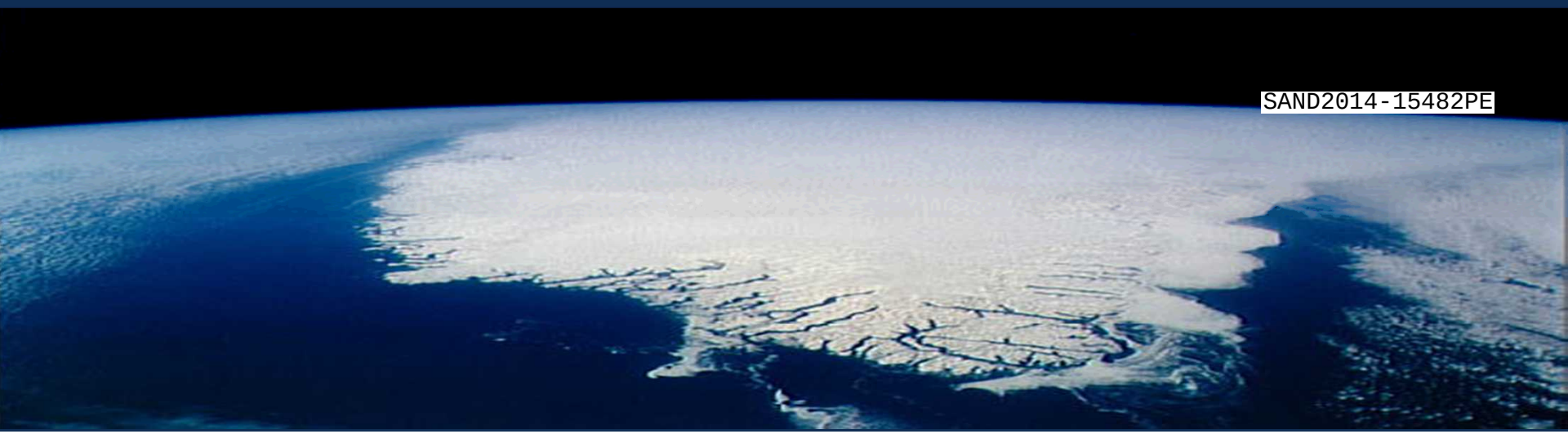
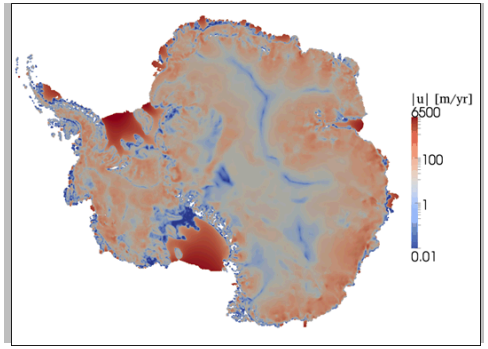


SAND2014-15482PE



A Performance-Portable Implementation of the Albany Ice Sheet Model: Kokkos Approach

Irina Demeshko, H. Carter Edwards, Michael A.Heroux, Eric T. Phipps, Andrew G. Salinger

Algorithms and Abstractions for Assembly in PDE Codes, July 10, 2014

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

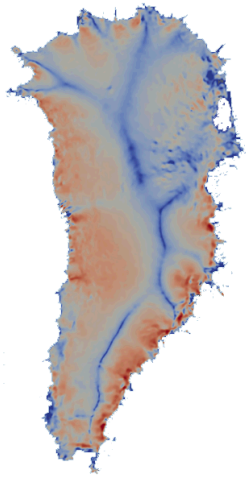
Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2011-XXXXP

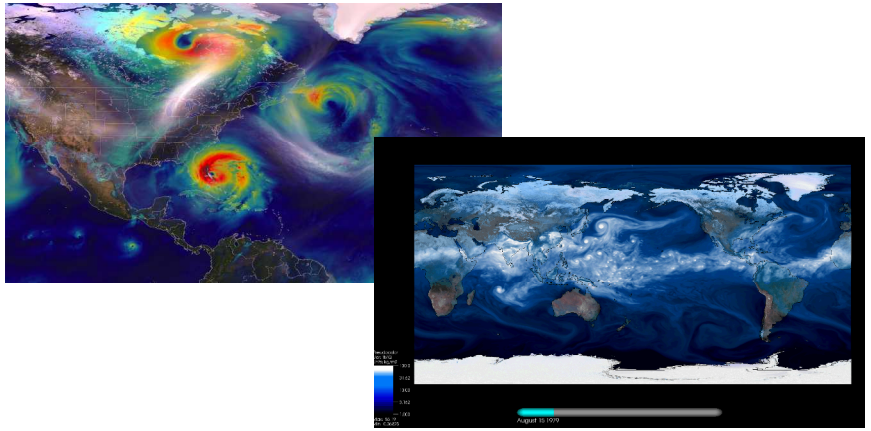
Outline

- Introduction
- Kokkos programming model
- Albany Greenland Ice Sheet Model (FELIX)
- Kokkos implementation of the FELIX model
- Performance results
- Conclusion and future plans

Introduction

Sandia National Laboratories





Tianhe-2 (National SuperComputer Center in Guangzhou)
Intel Xeon E5-2692+ Intel Xeon Phi

TITAN (ORNL)
Cray XK7 , Opteron 6274 NVIDIA K20x

Sequoia (DOE/NNSA/LLNL)
BlueGene/Q, Power BQC, Custom

Sandia National Laboratories

Performance portability has become a critical issue

parallel code needs to be executed correctly and performant despite variation in the architecture, operating system and software libraries.

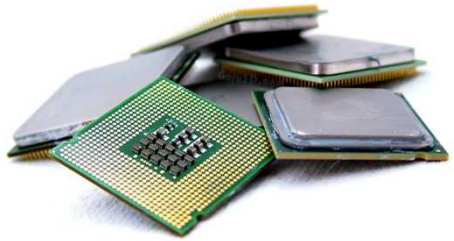
Approach: Kokkos programming model

C++ library, which provide performance portability across diverse devices with different memory models.


Kokkos

Kokkos* programming model

Sandia National Laboratories



- Kokkos - C++ library to provide scientific and engineering codes with an intuitive manycore performance portable **programming model**.

- ✓ **Standard C++, Not a language extension**
- ✓ **Uses C++ template meta-programming**
- ✓ **Provides portability across manycore devices (Multicore CPU, NVidia GPU, Intel Xeon Phi (potential: AMD Fusion))**
- ✓ **Abstract data layout for non-trivial data structures**

* Trilinos package

Kokkos* programming model

Sandia National Laboratories

- **Kokkos :**

✓ **Single code base**

✓ **Support for all current (and future) hardware**

✓ **Flexible run configurations MPI-Only**

- ✧ **MPI + Threads**
- ✧ **MPI + GPU**
- ✧ **MPI + GPU + Threads**

✓ **Close to optimal performance (i.e. performance of a specialized code)**

✓ **Use vendor compilers**

✓ **Simple code**

* Trilinos package

Kokkos* programming model

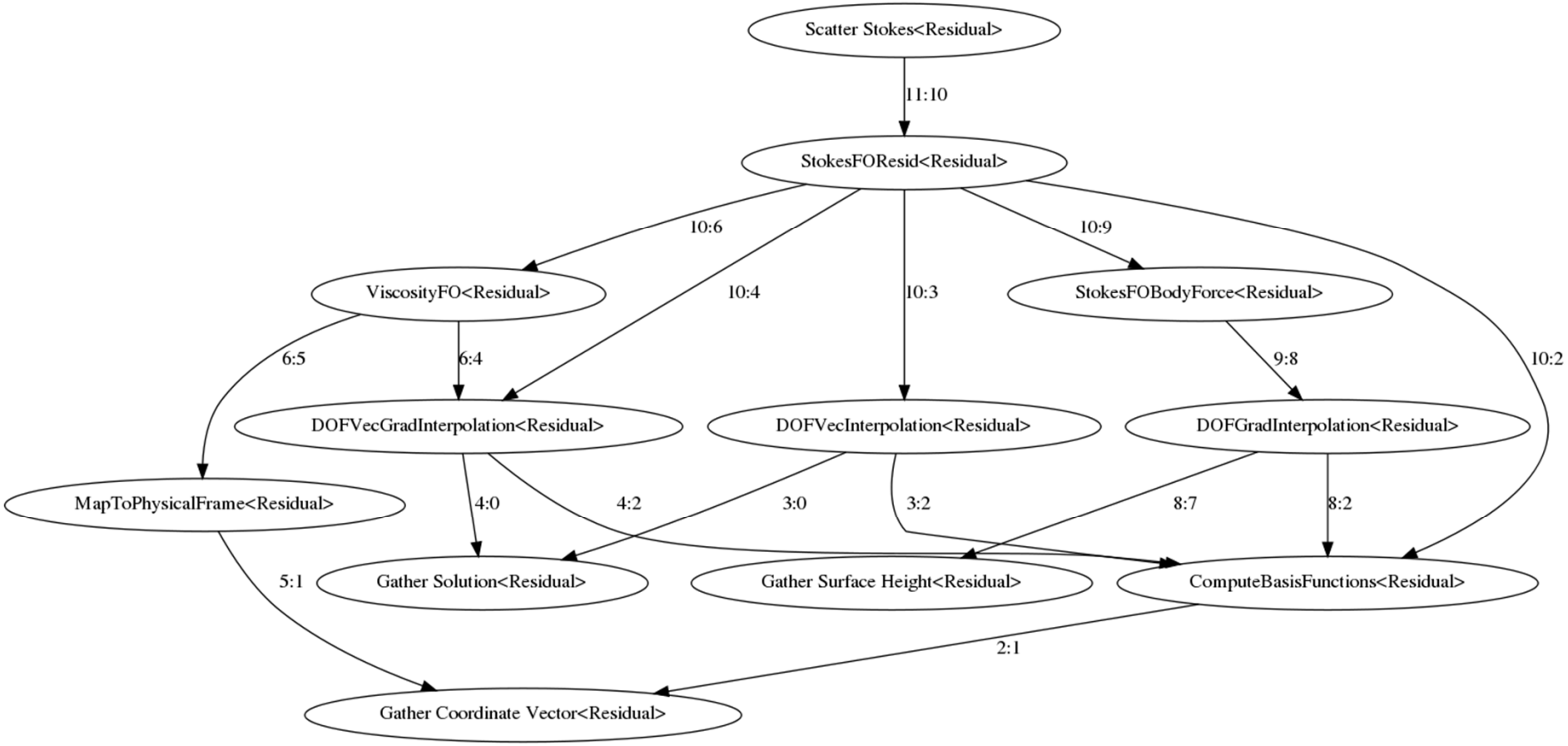
Sandia National Laboratories

A programming model with two major components:

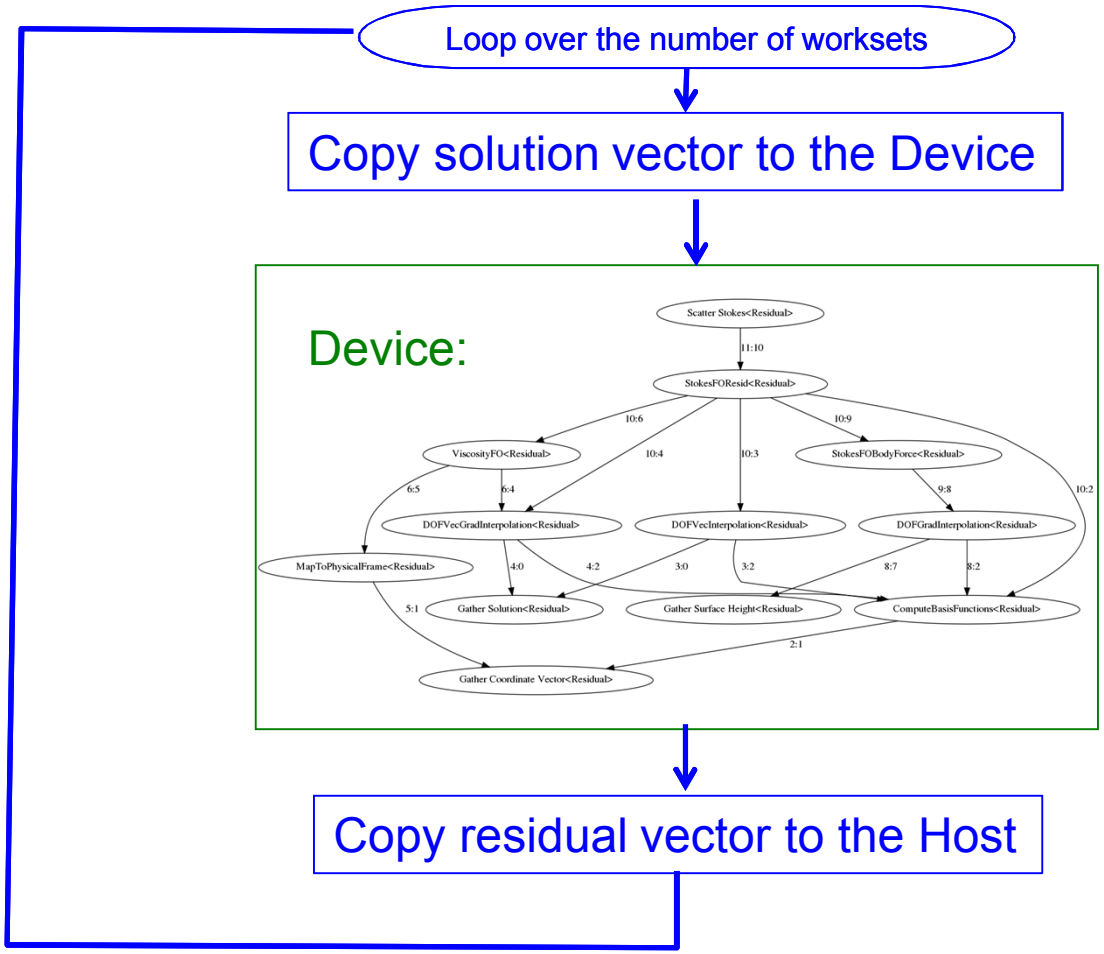
- **Data access abstraction**
  - ✓ Change data layout at compile time without changing access syntax => Optimal access pattern for each device
  - ✓ Data padding and alignment is transparent
    Access traits for portable support of hardware specific load/store units
- **Parallel dispatch**
  - ✓ Express algorithms with **parallel_for**, **parallel_reduce** etc.
    Using functor concept (functor: construct allowing an object to be invoked or called as if it were an ordinary function)
  - ✓ Transparently mapped onto back-end languages
    (e.g. OpenMP, CUDA, Pthreads …)

Albany Greenland Ice Sheet model

Greenland Ice-Sheet model



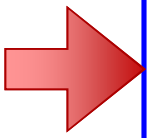
Greenland Ice-Sheet model (Kokkos implementation)



Kokkos_functor example: Jacobian

Sandia National Laboratories

```
template<typename EvalT, typename Traits>
Void Jacobian<EvalT, Traits>::evaluateFields(typename
Traits::EvalData d)
{
for(int cell = 0; cell < worksetNumCells; cell++) {
    for(int qp = 0; qp < numQPs; qp++) {
      for(int row = 0; row < numDims; row++){
        for(int col = 0; col < numDims; col++){
          for(int node = 0; node < numNodes; node++){
            jacobian(cell, qp, row, col) +=
                    coordVec(cell, node, row)
                    *basisGrads(node, qp, col);
          } // node
        } // col
      } // row
    } // qp
  } // cell
}
```



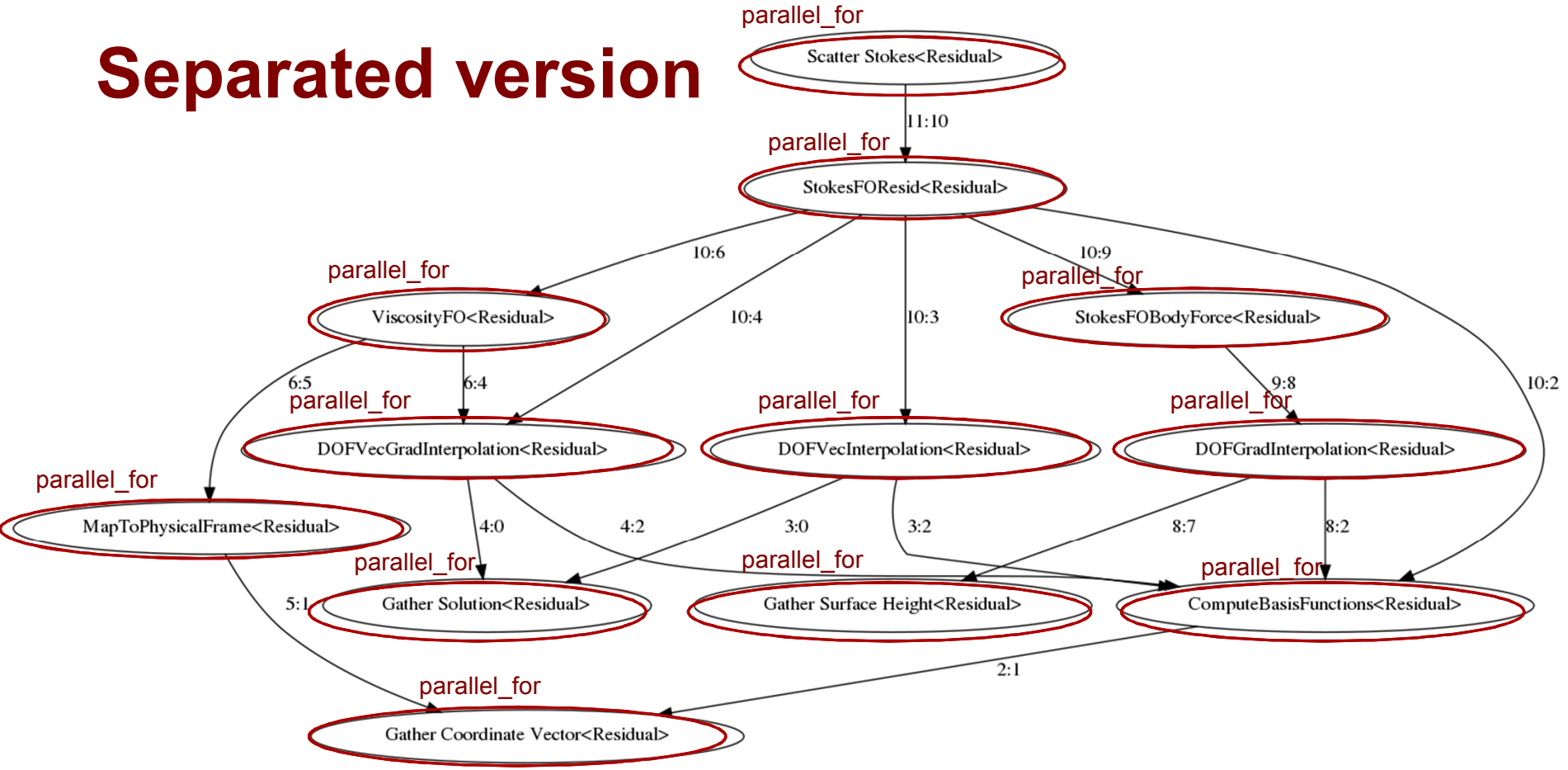
```
**********************************************
template<typename EvalT, typename Traits>
KOKKOS_INLINE_FUNCTION
void Jacobian<EvalT, Traits>:: operator () (const int i) const
 {
   for(int qp = 0; qp < numQPs; qp++) {
     for(int row = 0; row < numDims; row++){
       for(int col = 0; col < numDims; col++){
         for(int node = 0; node < numNodes; node++){
           jacobian(cell, qp, row, col) +=
                    coordVec(cell, node, row)
                    *basisGrads(node, qp, col);
         } // node
       } // col
     } // row
   } // qp

}
//**********************************************
template<typename EvalT, typename Traits>
Void Jacobian<EvalT, Traits>::evaluateFields(typename
Traits::EvalData d)
{
  Kokkos::parallel_for (worksetNumCells, *this);
}
```

2 GIS Kokkos implementations:

Sandia National Laboratories

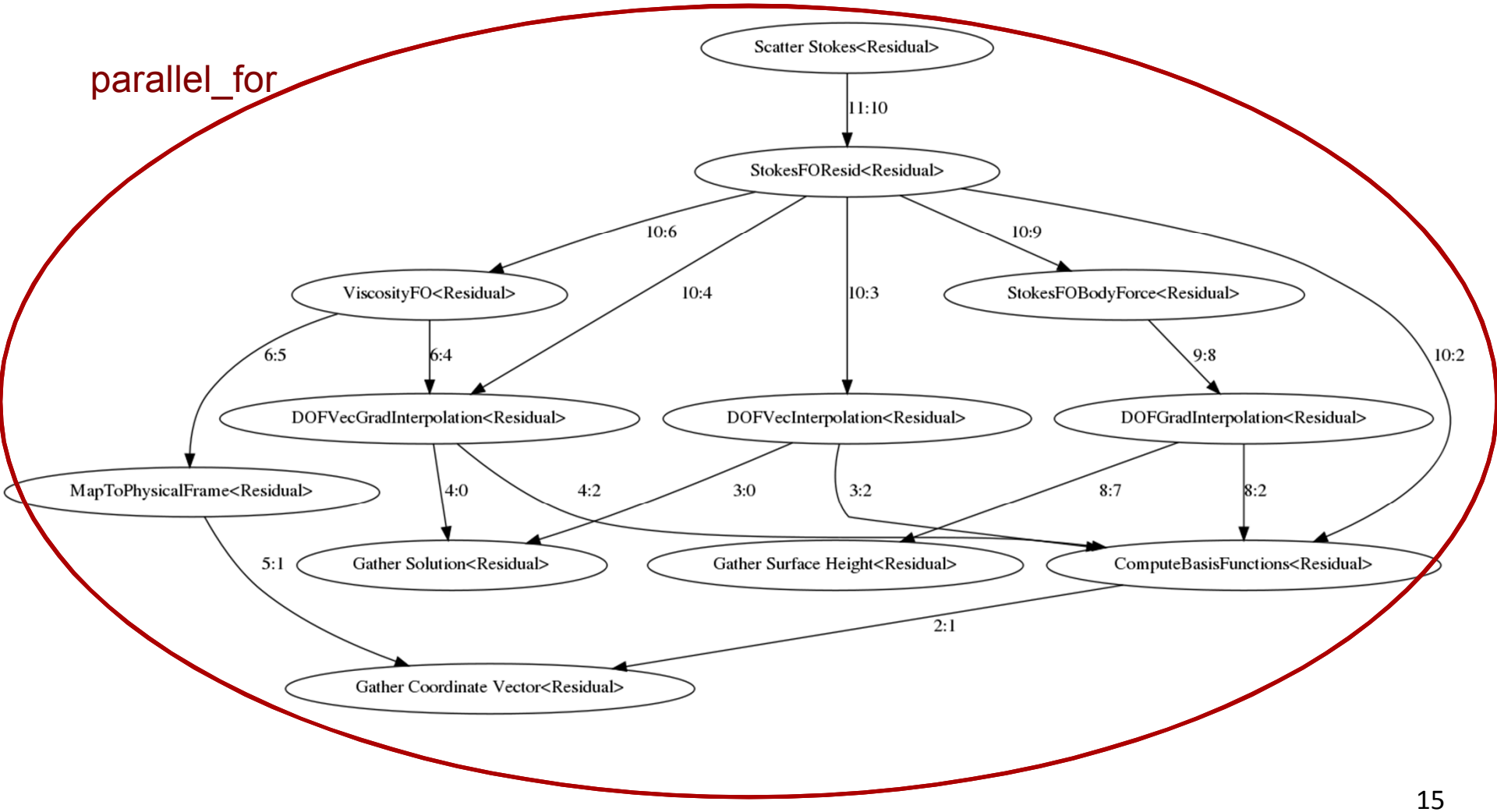
Separated version



2 GIS Kokkos implementations:

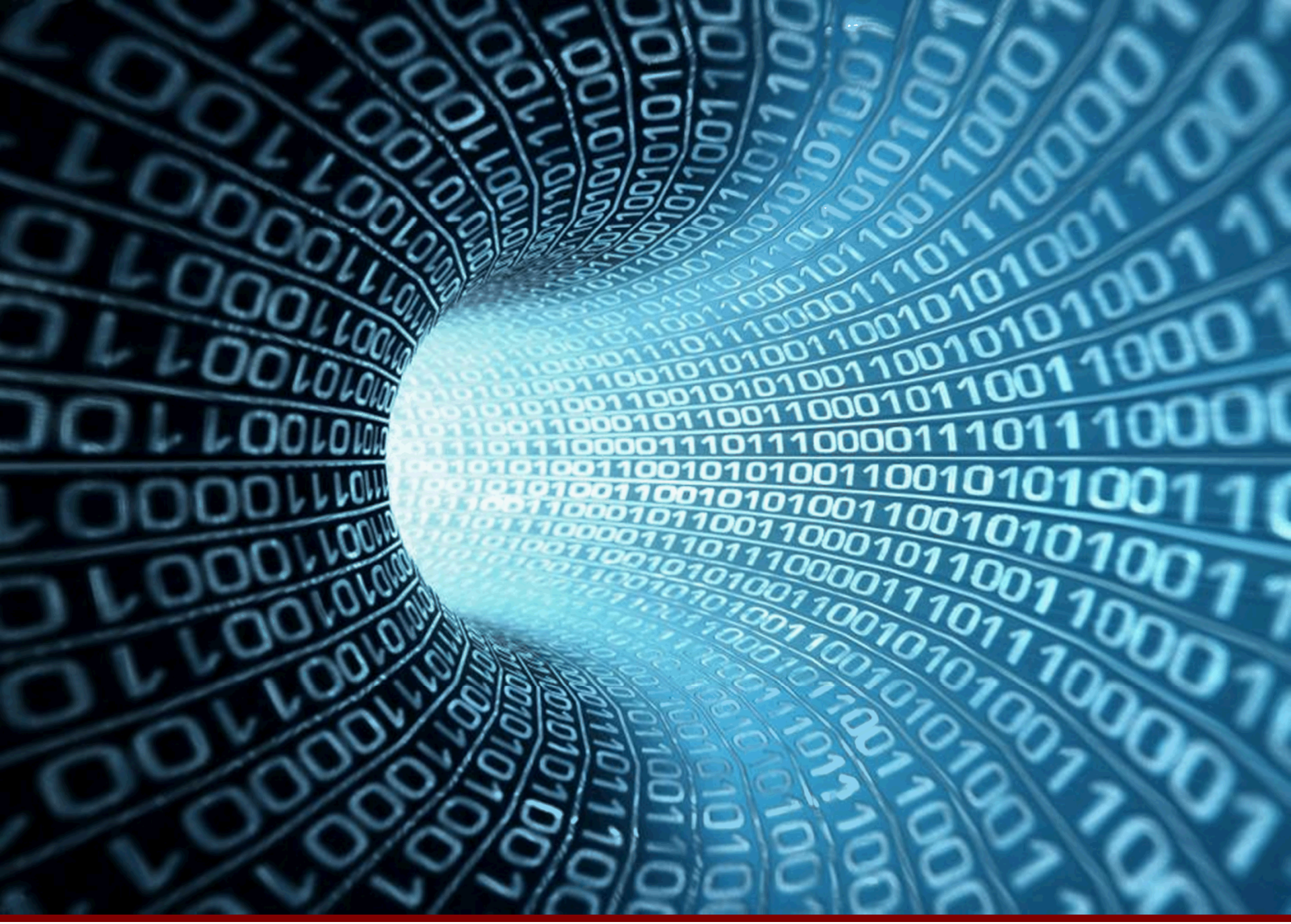
Fused version

parallel_for



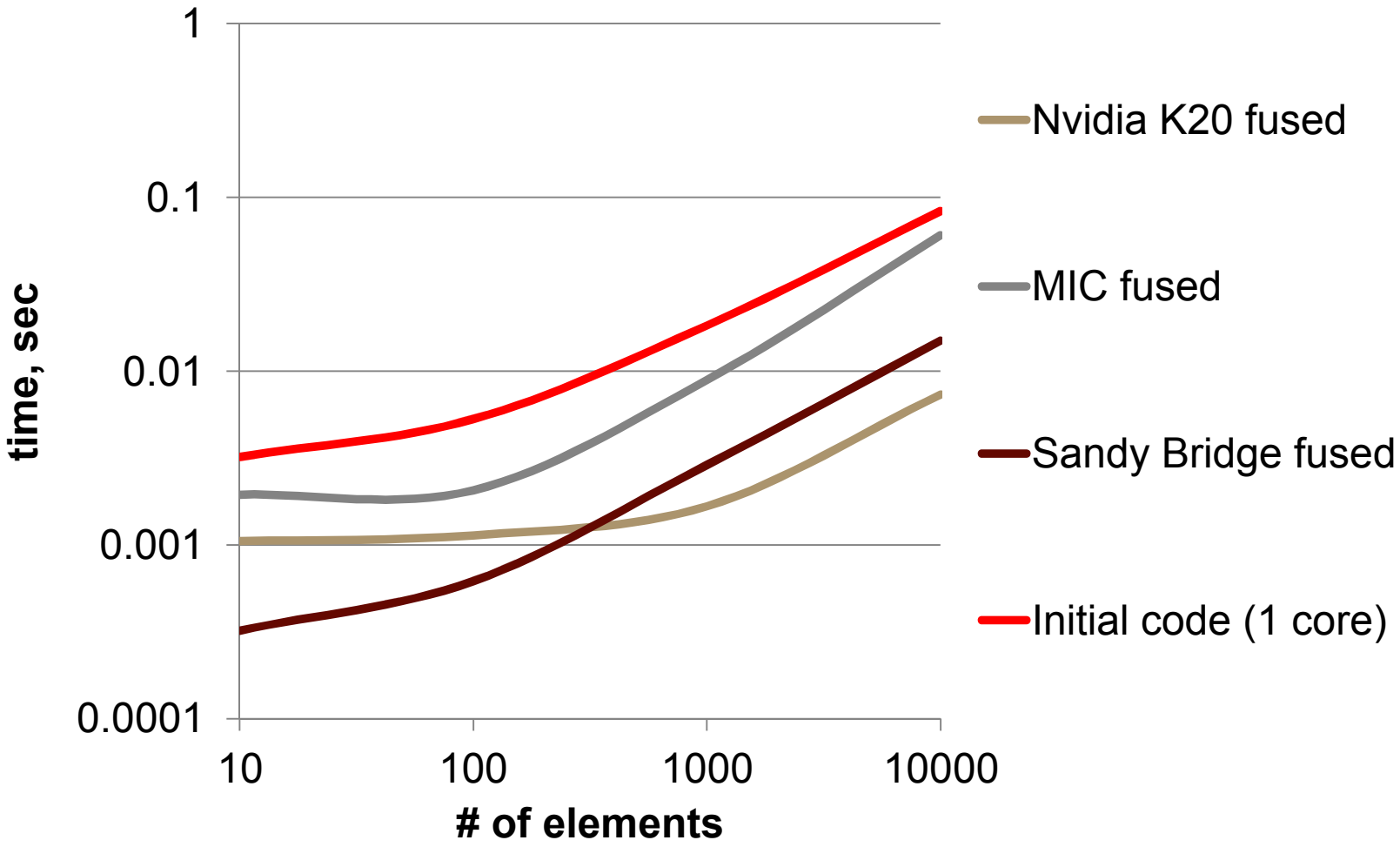
Sandia National Laboratories

Performance results



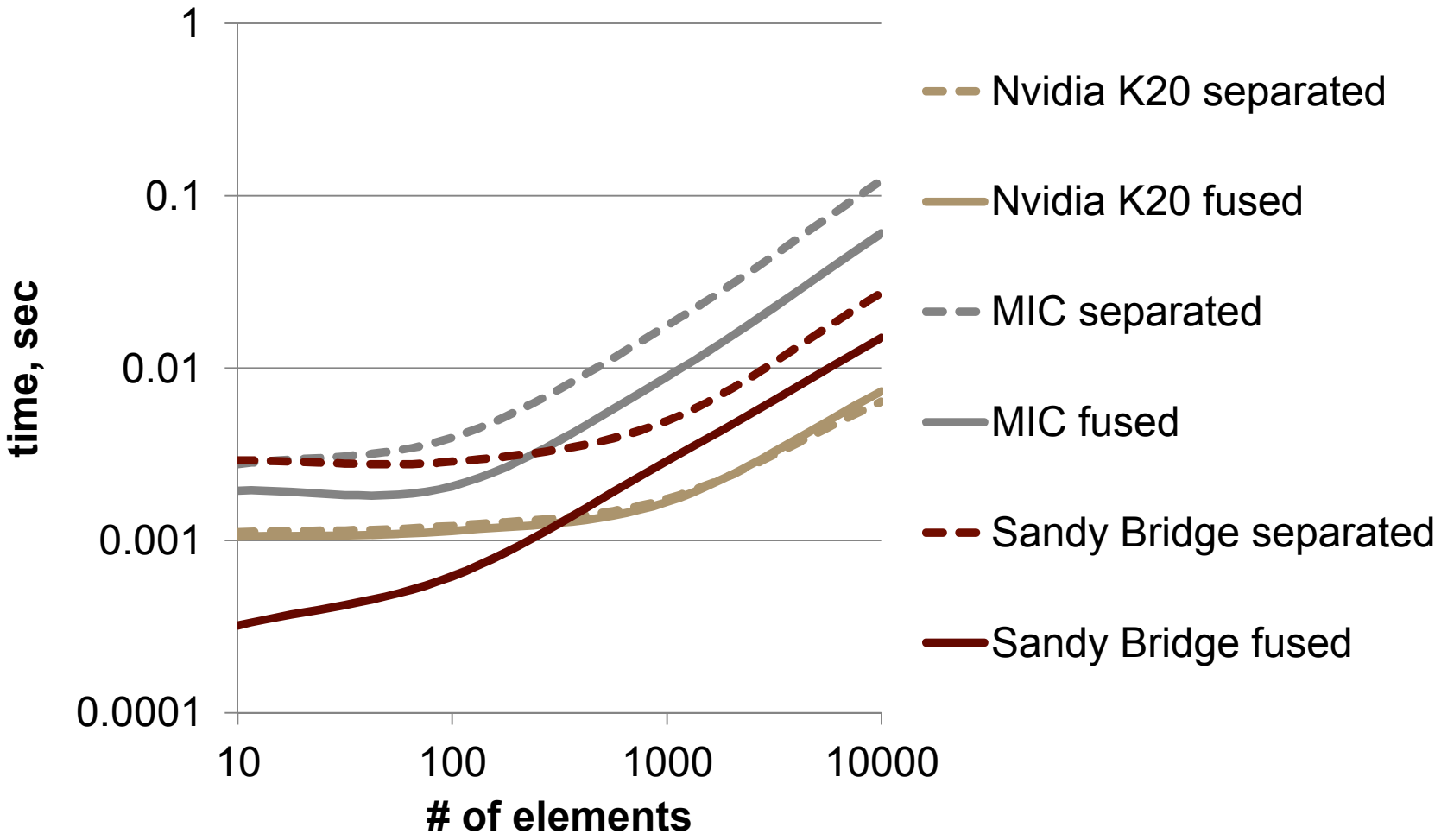
Performance results (fused version)

Sandia National Laboratories



- ** “Nvidia K20 GPU” and “Initial Implementation” results were obtained on Shannon, “Intel MIC” were obtained on Compton

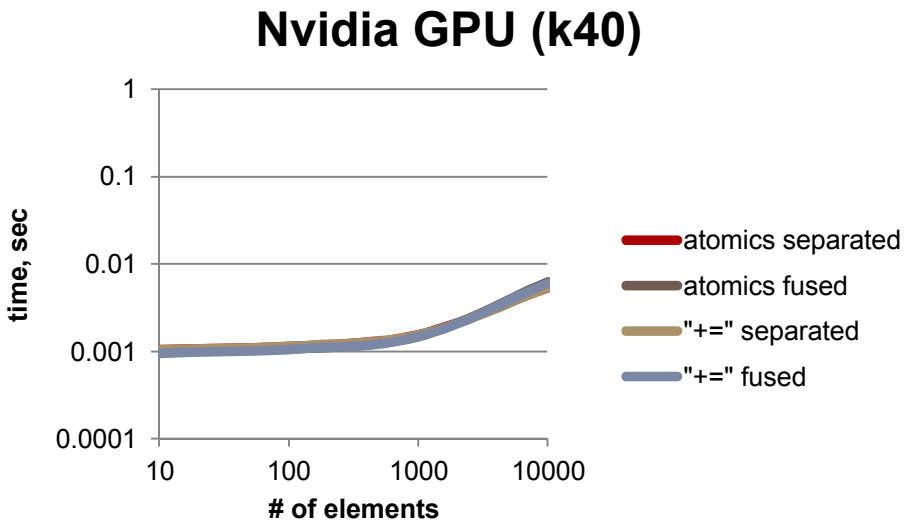
Separated vs Fused versions

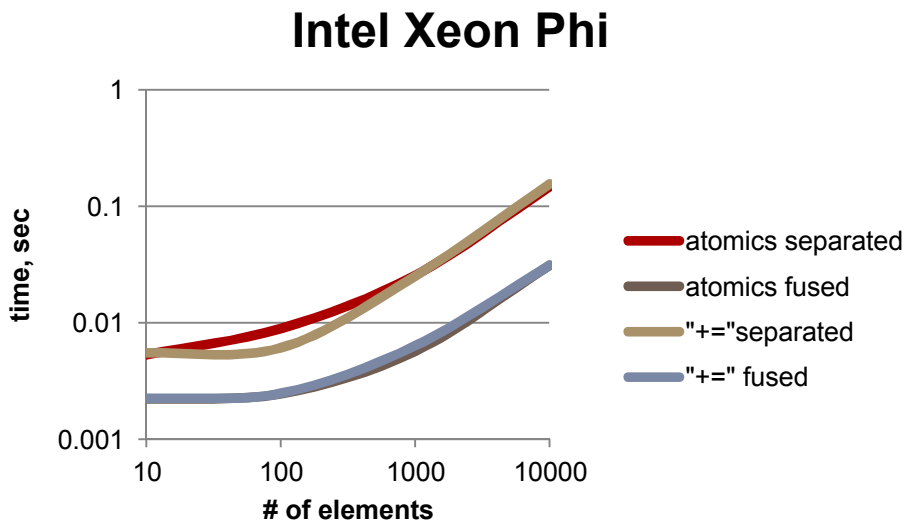


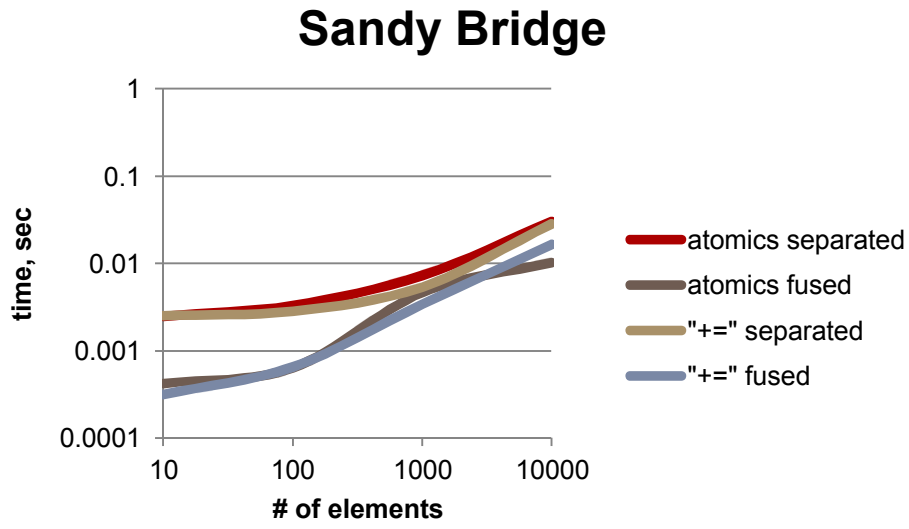
- ** “Nvidia K20 GPU” and “Initial Implementation” results were obtained on Shannon, “Intel MIC” were obtained on Compton

Atomics vs. “+=”

Sandia National Laboratories

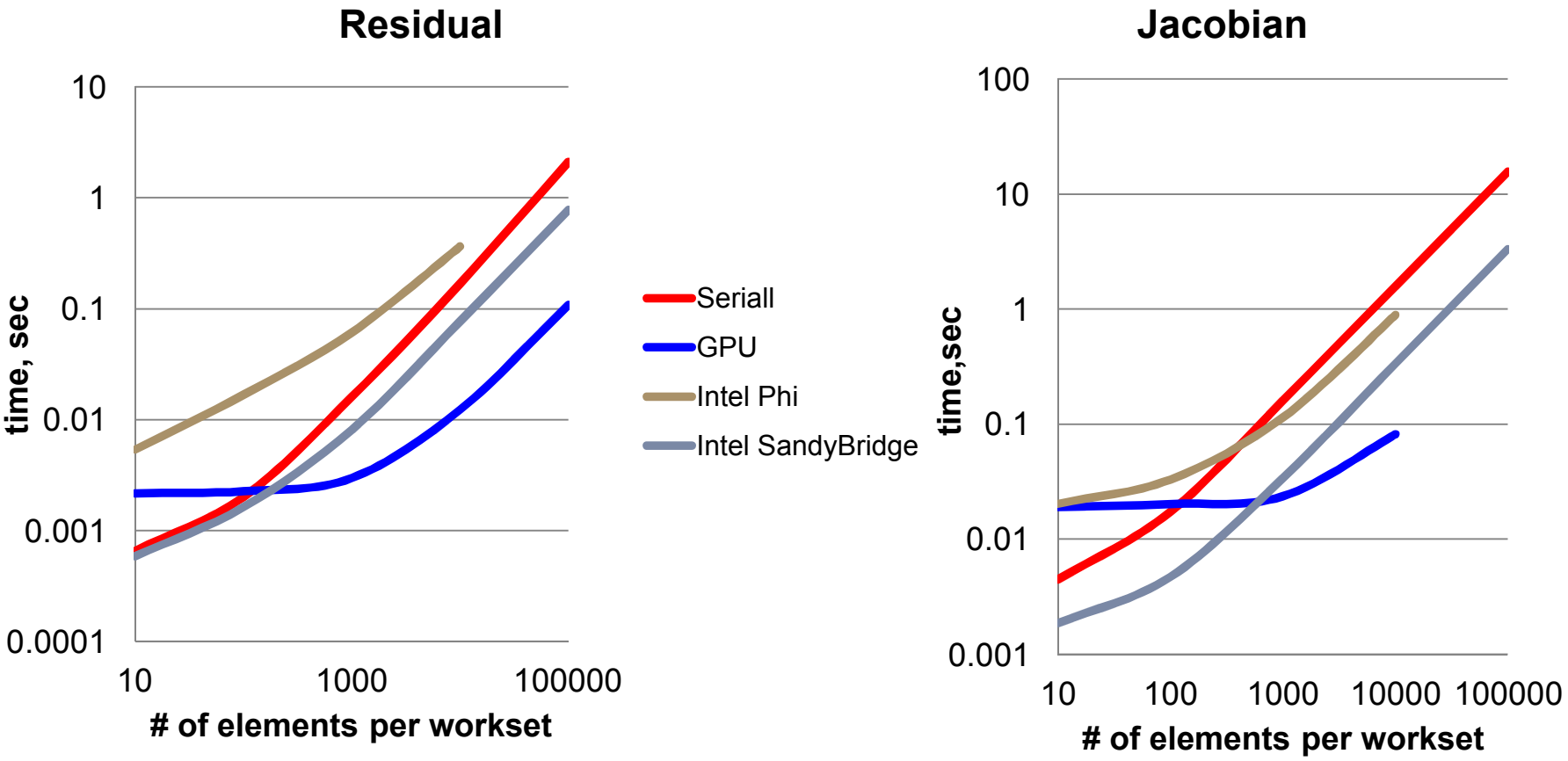






Another Example: Aeras code* (Albany-based)

Sandia National Laboratories



* Shallow Water test

Conclusion and future work

Sandia National Laboratories

Conclusions:

- Kokkos Ice Sheet model (FE Assembly):
  - shows good performance
  - portable across modern HPC architecture

- Using “Atomics” approach does not introduce significant overheads.

Ongoing work:

- Porting entire Albany application development environment to Kokkos .
- Porting Trilinos Linear Algebra Libraries to Kokkos