Performance Portability
Performance Portability
Performance Portability

EXASCALE SYSTEM

- new architecture
- new libraries
- new programming models
Kokkos
Albany: agile component-based parallel unstructured mesh application

- A finite element based application development environment containing the "typical" building blocks needed for rapid deployment and prototyping of analysis capabilities
  - A Trilinos demonstration application, built almost exclusively from reusable libraries. Albany leverages 100+ packages/libraries.
  - Open-source

**Strategic Goal:**
To enable the Rapid development of new Production codes embedded with Transformational capabilities.
supports a wide variety of application physics areas

- Heat transfer
- Fluid dynamics
- Quantum device modeling
- Structural mechanics
- Climate modeling
Albany team is Rapidly Developing Several New Component-Based Applications

1. Turbulent CFD for nuclear energy [NE]
2. Computational mechanics R&D [ASC]
3. Quantum device design [LDRD]
4. Extended MHD [ASCR]
5. CRADA partner’s in-house code [CRADA]
6. Peridynamics solver [ASC]
7. Biogeochemical element cycling: climate [SciDAC]
8. Fuel rod degradation modeling [NE]
9. Ice Sheet dynamics [SciDAC]
10. Atmospheric Dynamics [LDRD]

+ Impacting Many Others

Codes are born: parallel, scalable, robust, with sensitivities, optimization, UQ … and ready to adopt: embedded UQ, multicore kernels, adaptivity, code coupling, ROM
Our goal:
To create an architecture-portable version of Albany by using Kokkos library.
manages dependencies between different components of the Albany and manages data in the code.

Phalanx

Tpetra

Kokkos

Intrepid

Trilinos

Piro

MueLu

Library of interoperable tools for compatible discretizations of Partial Differential Equations.

implements linear algebra objects, including sparse graphs, sparse matrices, and dense vectors.
• A new Albany-Kokkos implementation:
  • has Kokkos::Views at the base layer
  • has Kokkos::Vew –like temporary data
  • has Kokkos kernels in replacement of original nested loops
  • is a single code base that runs and is performant on diverse HPC architectures
FELIX: Albany Greenland Ice Sheet model
Albany FELIX project

- An unstructured-grid finite element ice sheet code for land-ice modeling (Greenland, Antarctica).

**Project objective:**

- Provide sea level rise prediction
- Run on new architecture machines (hybrid systems).
  - 50% time spent in FE Assembly
  - 50% time spent in Linear Solves

**Funding Source:** SciDAC

**Collaborators:** SNL, ORNL, LANL, LBNL, UT, FSU, SC, MIT, NCAR

**Sandia Staff:** A. Salinger, I. Kalashnikova, M. Perego, R. Tuminaro, J. Jakeman, M. Eldred
Phalanx graph for the Greenland Ice-Sheet model
Kokkos implementation (Greenland Ice-Sheet model)

Loop over the number of worksets

Copy solution vector to the Device

Device:

Copy residual vector to the Host
Kokkos functor example in Albany

Initial code:

```cpp
template<
    typename EvalT
>
void CoordGrad<EvalT>::evaluateFields()
{
    
    // Outer loop over a Workset of Elements
    for(int cell = 0; cell < NumCells; cell++) {
        for(int qp = 0; qp < numQPs; qp++) {
            for(int row = 0; row < numDims; row++){
                for(int col = 0; col < numDims; col++){
                    coordGrad[cell][qp][row][col] +=
                        coordVec[cell][nd][row]
                        * basisGrads[nd][qp][col];
                }
            }
        }
    }
} // cell loop
```

Kokkos Functor implementation:

```cpp
template<
    typename EvalT
>
void CoordGrad<EvalT>:: evaluateFields()
{
    // Outer loop over a Workset of Elements
    Kokkos::parallel_for (NumCells, *this);
}

template<
    typename EvalT
>
KOKKOS_INLINE_FUNCTION
void CoordGrad<EvalT>:: operator ()
    (const int cell) const
{
    for(int qp = 0; qp < numQPs; qp++) {
        for(int row = 0; row < numDims; row++){
            for(int col = 0; col < numDims; col++){
                for(int nd = 0; nd < numNodes; nd++){
                    coordGrad(cell, qp, row, col) +=
                        coordVec(cell, nd, row)
                        * basisGrads(nd, qp, col);
                }
            }
        }
    }
} // eval()
FELIX Performance results

Evaluation environment:

**Shannon:**
32 nodes:
Two 8-core Sandy Bridge Xeon E5-2670 @ 2.6GHz (HT deactivated) per node,
128GB DDR3 memory per node,
2x NVIDIA K20x/k40 per node

Serial=2 MPI processes
OpenMP=16 OpenMP threads
CUDA=1 Nvidia K80 GPU
UVM for CPU-GPU data management
FELIX performance results

Evaluation environment:
TITAN:
18,688 AMD Opteron nodes:
- 16 cores per node,
- 1 K20X Kepler GPUs per node,
- 32GB + 6GB memory per node
• Next generation global atmosphere model.

• Numerics are similar to the Community Atmosphere Model - Spectral Elements (CAM-SE)

• Model development: shallow water, X-Z hydrostatic, 3D hydrostatic, clouds, 3D non-hydrostatic
Aeras performance results

Evaluation environment:

Shannon:
32 nodes:
Two 8-core Sandy Bridge Xeon E5-2670 @ 2.6GHz (HT deactivated) per node, 128GB DDR3 memory per node, 2x NVIDIA K20x/k40 per node
Aeras performance results

Evaluation environment:
**TITAN:**
- 18,688 AMD Opteron nodes:
  - 16 cores per node,
  - 1 K20X Kepler GPUS per node,
  - 32GB + 6GB memory per node
Conclusion

• New version of Albany provides architecture-portability;

• Our numerical experiments on two climate applications implemented in Albany show that:
  
  (1) a single code can execute correctly in several evaluation environments (MPI, OpenMP, CUDAUVVM), and

  (2) reasonable performance is achieved across the different architectures without implicit data management: speed-ups using OpenMP and GPUs can be achieved over an MPI-only run;
Acknowledgments

I would like to thank:

• C. R. Trott and H.C. Edwards for their help with Kokkos,
• Adam V. Delora for his work on Intrepid,
• Eric T. Phipps, Eric C. Cyr and Andrew Bradley for their help with Trilinos and Albany,
• Steve Price and Matt Hoffman and Mauro Perego for providing the data used in the FELIX land-ice runs.
Thank you!

irina@lanl.gov