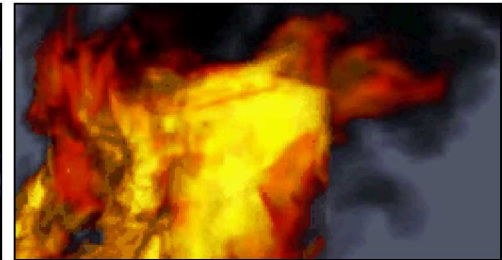




$$\partial_a^m J_{a,\sigma^2}(\xi_1) = \frac{(\xi_1 - a)}{\sigma^2} f_{a,\sigma^2}(\xi_1)$$
$$\int_{\mathbb{R}_+} T(x) \cdot \frac{\partial}{\partial \theta} f(x, \theta) dx = M \left( T(\xi) \cdot \frac{\partial}{\partial \theta} \ln U(\theta) \right)$$



## Kokkos: Present and Future

Unclassified Unlimited Release

**Christian R. Trott**, - Center for Computing Research  
Sandia National Laboratories/NM



Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and 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-NA-0003525.



# Transitioning To Community Project



- Core: 15 Developers (8 SNL)
- More code contributions from non-SNL
  - >50% of code reviews by ORNL
  - >50% of commits from non-Sandians
- Sandia leads API design
- Other labs lead backend implementations



Sandia  
National  
Laboratories



BERKELEY LAB

## Kokkos Core:

**C.R. Trott**, N. Ellingwood, D. Ibanez, J. Miles, D. Hollman, V. Dang, Jan Ciesko, J. Wilke, L. Cannada,  
H. Finkel, N. Liber, D. Lebrun-Grandie, B. Turcksin, J. Madsen, D. Arndt, J. Madsen, R. Gayatri  
former: **H.C. Edwards**, D. Labreche, G. Mackey, S. Bova, D. Sunderland,

## Kokkos Kernels:

**S. Rajamanickam**, L. Berger, V. Dang, N. Ellingwood, E. Harvey, B. Kelley, K. Kim, C.R. Trott, J. Wilke, S. Acer

## Kokkos Tools:

**D. Poliakoff**, S. Hammond, C.R. Trott, D. Ibanez, S. Moore, L. Cannada

## Kokkos Support:

**C.R. Trott**, G. Shipman, G. Lopez, G. Womeldorff,  
former: **H.C. Edwards**, D. Labreche, Fernanda Foertter



# Kokkos Uptake

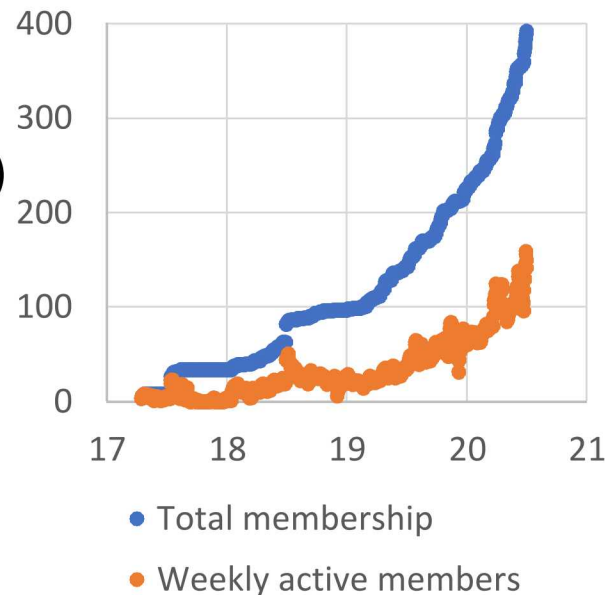
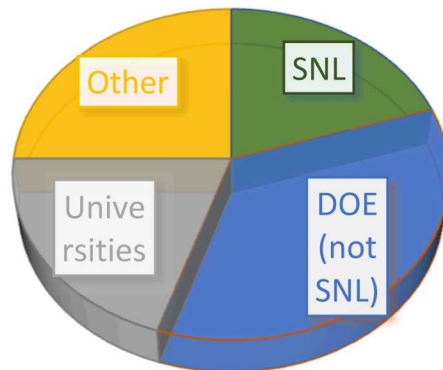


## ECP Critical Dependencies

|               |           |          |    |
|---------------|-----------|----------|----|
| MPI           | 60        | hypre    | 11 |
| LLVM          | 53        | Trilinos | 10 |
| C++           | 41        | DAV-SDK  | 9  |
| OpenMP        | 33        | ADIOS    | 8  |
| CUDA          | 22        | VTK-m    | 8  |
| HDF5          | 19        | FFT      | 7  |
| LAPACK        | 19        | Spack    | 7  |
| <b>Kokkos</b> | <b>18</b> | OpenACC  | 6  |
| Fortran       | 17        | MPI-IO   | 6  |
| BLAS          | 16        | PnetCDF  | 6  |
| C             | 14        | zfp      | 5  |
| ALPINE        | 12        | SUNDIALS | 5  |

## Kokkos Slack Users

- 440 registered users
  - 70 Institutions
  - Every continent
    - (-Antarctica)
- Doubles every year

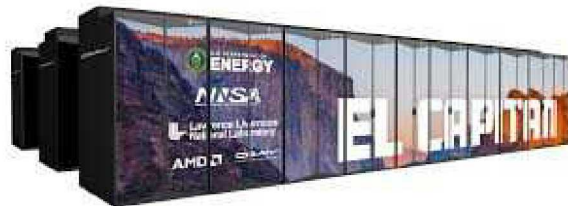


# Exascale Readiness



## ***Frontier/El Capitan: HIP***

- Primary development at ORNL
- Many Capabilities ready
  - Some Hierarchical parallelism is waiting for compiler bugs
- PR testing for Kokkos on AMD GPUs in place
- ArborX, Cabana, LAMMPS (partially) working



## ***Aurora: DPC++ and OpenMP 5.0***

- DPC++ blocked by compiler
  - Working with Intel on it
- OpenMP 5.0 similar state as HIP

*Kokkos 3.3 (Nov 2020): OpenMP 5 and HIP expected to be largely feature complete*



# Updates: Training Material



- Developed The Kokkos Lectures
  - 8 lectures covering most aspects of Kokkos
  - 14 hours of recordings
  - > 500 slides
  - >20 exercises
- Hosted by ECP
  - Module 8 this Friday
- Module 1: Introduction
  - Introduction, Basic Parallelism, Build System
- Module 2: Views and Spaces
  - Execution and Memory Spaces, Data Layout
- Module 3: Data Structures and MDRangePolicy
  - Tightly Nested Loops, Subviews, ScatterView,...
- Module 4: Hierarchical Parallelism
  - Nested Parallelism, Scratch Pads, Unique Token
- Module 5: Advanced Optimizations
  - Streams, Tasking and SIMD
- Module 6: Language Interoperability
  - Fortran, Python, MPI and PGAS
- Module 7: Tools
  - Profiling, Tuning , Debugging, Static Analysis
- Module 8: Kokkos Kernels
  - Dense LA, Sparse LA, Solvers, Graph Kernels

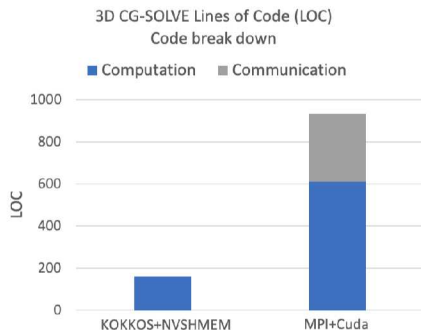
<https://github.com/kokkos/kokkos-tutorials/wiki/Kokkos-Lecture-Series>



# Updates: Remote Spaces and Tooling



- Tools Support is growing
  - More Native support e.g. Tau
- Connectors to Timemory etc.
- Nsight Systems does more useful stuff with connectors
  - Rename Kernels



- Remote Spaces beta now released
  - <https://github.com/kokkos/kokkos-remote-spaces>
- Support for NVSHMEM, MPI, SHMEM
- Working on Caching, aggregation etc.
- Potentially huge productivity benefits

TAU: ParaProf: Statistics for: node 0, thread 0 - exami

| Name                                                                   | Exclusive TIME |
|------------------------------------------------------------------------|----------------|
| .TAU application                                                       | 0.143          |
| Comm::exchange                                                         | 0.001          |
| Comm::exchange_halo                                                    | 0.001          |
| Comm::update_halo                                                      | 0.004          |
| Kokkos::parallel_for CommMPI::halo_update_pack [device=0]              | 0.002          |
| Kokkos::parallel_for CommMPI::halo_update_self [device=0]              | 0.003          |
| Kokkos::parallel_for CommMPI::halo_update_unpack [device=0]            | 0.002          |
| MPI_Irecv()                                                            | 0.001          |
| MPI_Send()                                                             | 29.268         |
| MPI_Wait()                                                             | 0.001          |
| OpenMP_Implicit_Task                                                   | 0.041          |
| OpenMP_Parallel_Region parallel_for<Kokkos::RangePolicy<CommMPI::Ta    | 0              |
| OpenMP_Parallel_Region parallel_for<Kokkos::RangePolicy<CommMPI::Ta    | 0.08           |
| OpenMP_Parallel_Region void Kokkos::parallel_for<Kokkos::RangePolicy<I | 0.001          |
| OpenMP_Sync_Region_Barrier parallel_for<Kokkos::RangePolicy<CommMF     | 0.489          |
| OpenMP_Sync_Region_Barrier parallel_for<Kokkos::RangePolicy<CommMF     | 0.875          |
| OpenMP_Sync_Region_Barrier void Kokkos::parallel_for<Kokkos::RangePol  | 0.58           |



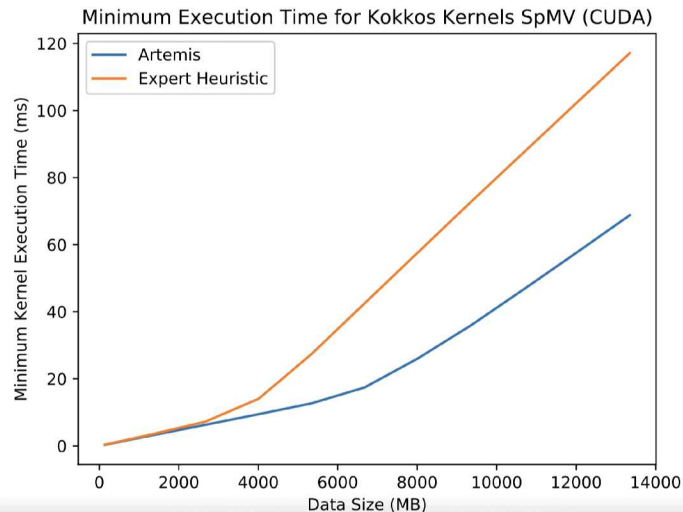
# New Capabilities: Auto Tuning



- Part of Kokkos 3.2 (released last week)
- Tuning Interface + Tools
  - Same as other hooks: they are always there, but act as no-ops without a tool
- Multi Input – multi Output tuning
  - Inputs describe problem space
  - OutputTypes describe search space
    - Sets, Ranges, Categorical
    - Logarithmic, linear
- Tuning scopes can include multiple kernels
- Tuning of internal variables in 3.3 or 3.4

## ***Apollo Tuner for SPMV tuning:***

- Rows per team
- Team Size
- Vector Length





# New Capabilities: Static Analysis



- Can we catch violations of Kokkos semantics even if code would compile/run?
  - kokkos-llvm: fork of LLVM with Kokkos aware clang-tidy variant
- Three types of violating patterns:
  - compile with some backends but not others.
  - run correct with some backends but crash on others.
  - run correct with some backends but have wrong results with others!

## Example: Missing function markup

```
void fooOOPS(int i) { printf("%i\n", i); }

int main(int argc, char **argv) {
    Kokkos::initialize();
    Kokkos::parallel_for(15, KOKKOS_LAMBDA(int i) {
        fooOOPS(i);
    });
    Kokkos::finalize();
}
```

```
>clang-tidy -checks=--,kokkos-* file.cpp
<main.cpp:7:5> warning: Function 'fooOOPS' called in
a lambda was missing KOKKOS_X_FUNCTION annotation.
fooOOPS(i);
^
<main.cpp:2:1> note: Function 'fooOOPS' was declared here
void fooOOPS(int i) { printf("%i\n", i); }
```



# Upcoming Capabilities: Graph Interface



- Build static graphs of kernels
  - Can use CUDAGraphs as backend
  - Allows repeated dispatch
- Helps with Latency Limited codes
  - Cuts down on launch latency
  - Can leverage streams to overlap work
  - Infers overlapping from dependencies
- Prototype release planned as part of Kokkos 3.3 (November)

```
const auto graph = Kokkos::Experimental::create_graph(  
  [=](auto builder) {  
    auto root = builder.get_root();  
    auto f1 = root.then_parallel_for(  
      Kokkos::RangePolicy<>(0, 1), KOKKOS_LAMBDA(long) {...});  
    auto f2a = f1.then_parallel_for(  
      Kokkos::RangePolicy<>(0, 1), KOKKOS_LAMBDA(long) {...});  
    auto f2b = f1.then_parallel_for(  
      Kokkos::RangePolicy<>(0, 1), KOKKOS_LAMBDA(long) {...});  
    builder.when_all(f2, f3).then_parallel_for(  
      Kokkos::RangePolicy<>(0, 1), KOKKOS_LAMBDA(long) {...});  
  });  
  
for (int i = 0; i < repeats; ++i) {  
  graph.submit();  
  graph.get_execution_space().fence();  
}
```

