SAND2017-0880C

Kokkos: Performance Portability for C++ Codes

Sandia National Laboratories
H.C. Edwards, C.R. Trott, D. Sunderland, N. Ellingwood, G.E. Mackey, S.D. Hammond
(Research, Development, and Support)

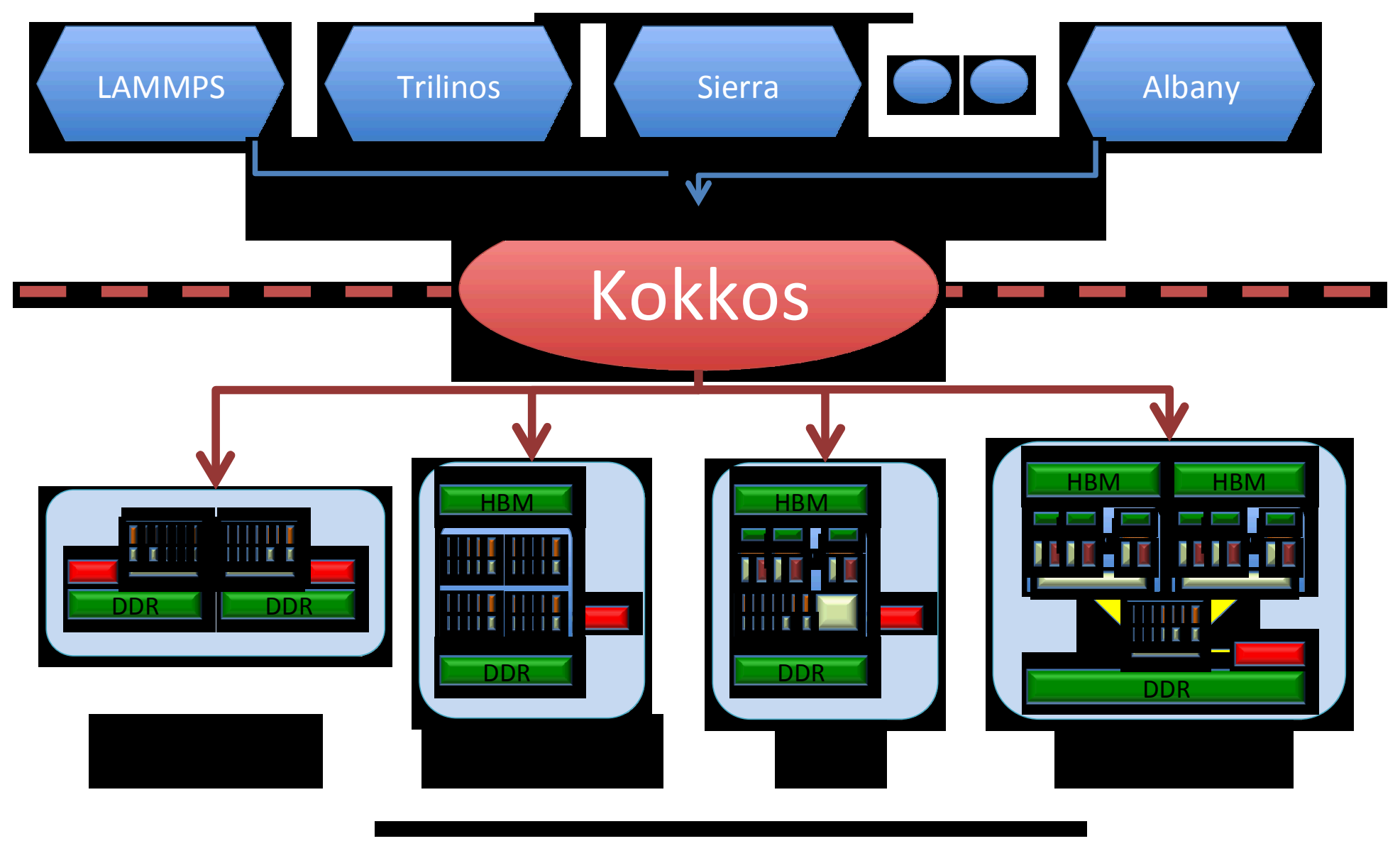
Los Alamos National Laboratories
G. Shipman (Kokkos-Support)

Oak Ridge National Laboratories
F. Foertter (Kokkos-Support)

ECP 1.3.1.05 Research & Development
ECP 1.3.1.12 Application Support

Why another Programming Model?

Programming applications for performance and portability across modern computing architectures is extremely challenging for many application developers. Hardware-native programming models may provide much higher performance but may only run on one vendor or even one machine. Standardized cross-platform models provide greater porting potential but often much lower performance. With a C++ abstraction based model you can have the best of both worlds.



github.com/kokkos

- Code Repository
- Issue Tracking
- Tools Repository
- Tutorials
- Documentation

Collaborators

Users

Sandia National Laboratories
ARL
Los Alamos National Laboratory EST. 1943
U.S. Naval Research Laboratory
Oak Ridge National Laboratory
The University of Utah

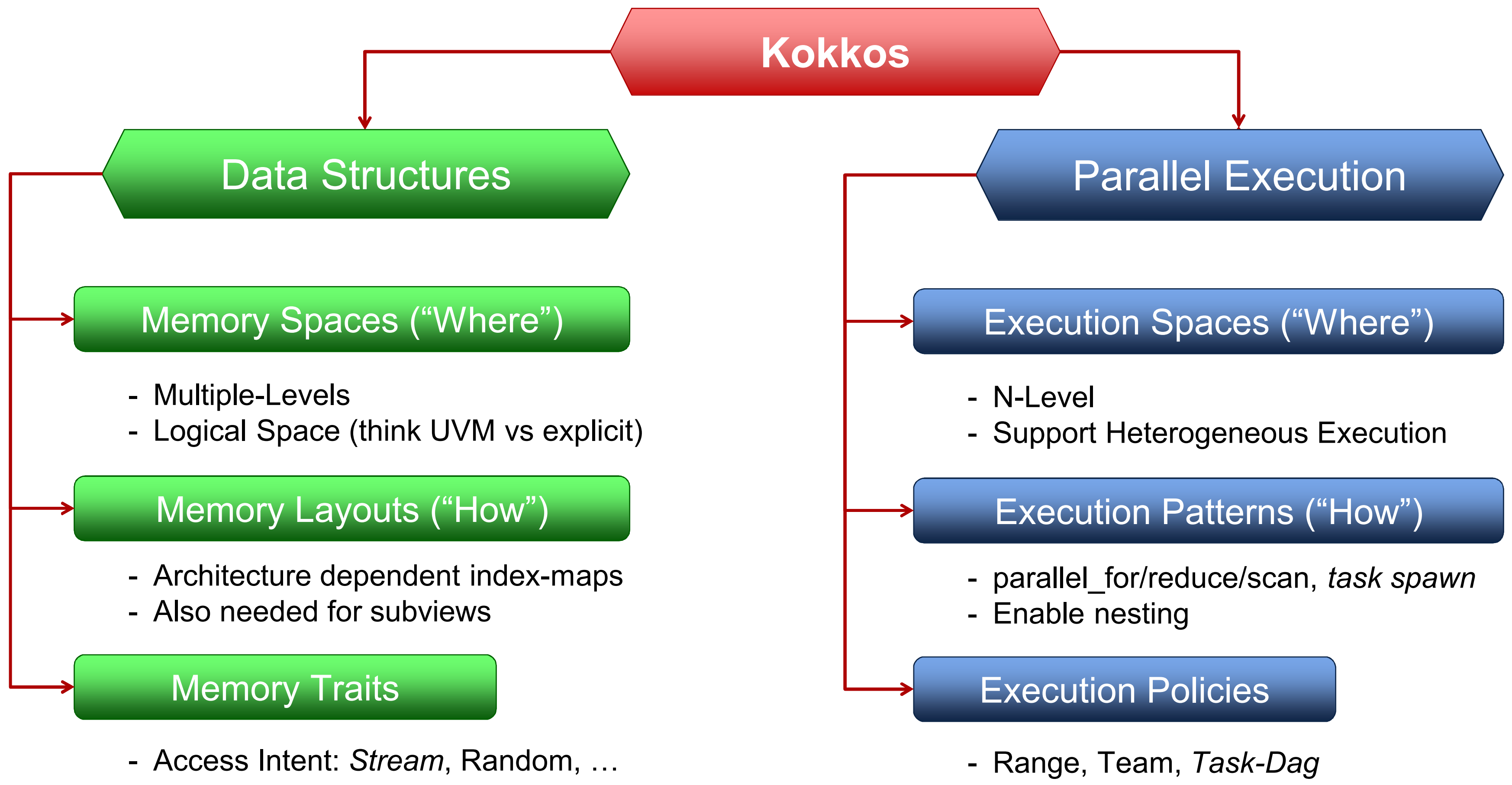
Backend Optimization

intel
IBM
NVIDIA
AMD

Applications/Libraries

Trilinos
Uintah
RAPTOR
LAMMPS
ALU LOW MACH
Sierra Mechanics

Abstraction Concepts



Kokkos

Data Structures

Memory Spaces (“Where”)
- Multiple-Levels
- Logical Space (think UVM vs explicit)

Memory Layouts (“How”)
- Architecture dependent index-maps
- Also needed for subviews

Memory Traits
- Access Intent: *Stream*, Random, …
- Access Behavior: Atomic
- Enables special load paths: i.e. texture

Parallel Execution

Execution Spaces (“Where”)
- N-Level
- Support Heterogeneous Execution

Execution Patterns (“How”)
- parallel_for/reduce/scan, *task spawn*
- Enable nesting

Execution Policies
- Range, Team, *Task-Dag*
- Dynamic / Static Scheduling
- Support non-persistent scratch-pads

Capabilities

Concept	Example
Parallel Loops	parallel_for(N, KOKKOS_LAMBDA (int i) { ...BODY... });

Parallel Reduction	parallel_reduce(RangePolicy<ExecSpace>(0,N), KOKKOS_LAMBDA (int i, double& upd) {
	  ...BODY...
	  upd += ...
	}, result);

Tightly Nested Loops (exp)	parallel_for(MDRangePolicy<Rank<3> > ({0,0,0},{N1,N2,N3},{T1,T2,T3}, KOKKOS_LAMBDA (int i, int j, int k) {...BODY...});

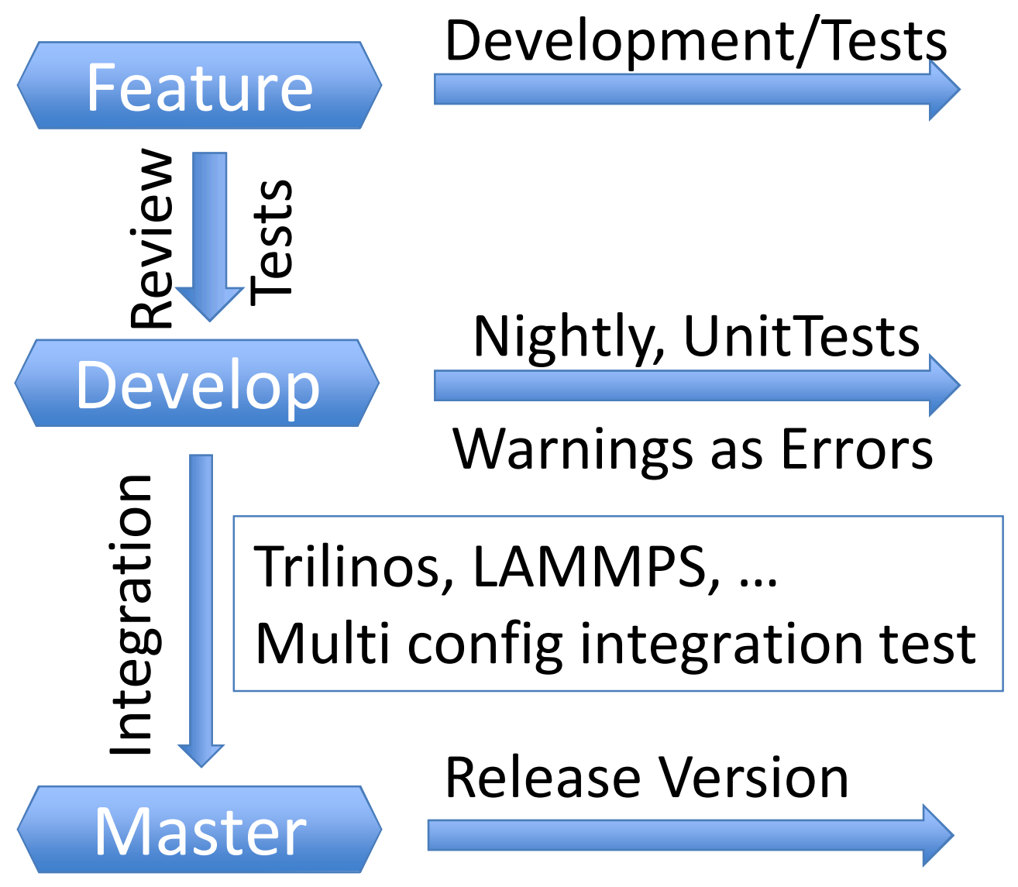
Non-Tightly Nested Loops	parallel_for(TeamPolicy<Schedule<Dynamic>>(N, TS), KOKKOS_LAMBDA (Team team) {
	  ... COMMON CODE 1 ...
	  parallel_for(TeamThreadRange(team, M(N)), [&] (int j) { ... INNER BODY... });
	  ... COMMON CODE 2 ...
	});

Task Dag (exp)	task_spawn(TaskTeam(scheduler , priority), KOKKOS_LAMBDA (Team team) { ... BODY });
Data Allocation	View<double**, Layout, MemSpace> a(“A”,N,M);
Data Transfer	deep_copy(a,b);

Sibling Projects

Kokkos-Tools	Profiling and Debuggin Tools. Can be used on release builds. No overhead if not loaded. Interface to third party tools (VTune, NVProfi etc.).
Kokkos-Kernels	BLAS, Sparse, and Graph Kernels. Takes Kokkos Views. Can internally call vendor libraries if data type / layout matches.

Testing and Software Quality



Feature
Development/Tests
Review
Tests
Develop
Nightly, UnitTests
Warnings as Errors
Integration
Trilinos, LAMMPS, …
Multi config integration test
Master
Release Version

New Features are developed on forks, and branches. Limited number of developers can push to develop branch. Pull requests are reviewed/tested.

Compilers	GCC (4.8-6.0), Clang (3.6-4.0), Intel (15.0-17.1), IBM (13.1.5), PGI (17.1), NVIDIA (7.0-8.0)
Hardware	Intel Haswell, Intel KNL, ARM v8, IBM Power8, NVIDIA K80, NVIDIA P100
Backends	OpenMP, Pthreads, Serial, Cuda

Each merge into master is minor release. Extensive integration test suite ensures backward compatibility, and catching of unit-test coverage gaps.

Kokkos Support – An ECP Project

“ECP Applications effective use of Kokkos to achieve performance portability across exascale architectures”

Tutorials

Extensive tutorials available at: github.com/kokkos/kokkos-tutorials
Regularly given at conferences such as SC and GTC.

Online Support

Online user forum is being considered through ECP 1.3.1.12.
Public, active Github issues list including questions and planning.
Note: Sensitive questions (e.g. with respect to export controlled codes) are handled outside of public github.

Bootcamps

Annual bootcamps with tutorials and hackathons will be organized at Oak Ridge and Sandia National Laboratories. Priority for participation given to ECP applications and libraries.

On-Site Support

On-site (SNL, ORNL, LANL) consulting arranged on a case by case basis.

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.