Kokkos Kernels: Overview

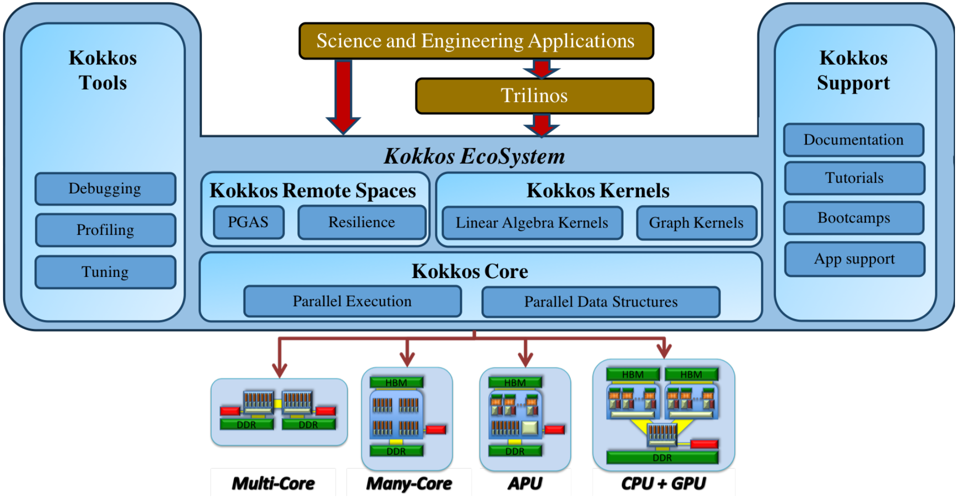
Luc Berger-Vergiat, Sandia National Laboratories

Kokkos User Group Meeting 2023

December 12, 2023

Overview

Kokkos Kernels in the Kokkos Ecosystem



The Team

Kokkos Kernels team:

- Siva Rajamanicakam *team lead*
- Luc Berger-Vergiat *team co-lead*
- Vinh Dang
- Nathan Ellingwood
- James Foucar
- Brian Kelley
- Kim Liegeois
- Carl Pearson
- Ernesto Prudencio

Components

- Batched (dense and sparse)
- BLAS
- Graph
- Lapack $\rightarrow$ New host base interface primarily wrapping TPLs (LAPACK, cuSOLVER, rocSOLVER, MAGMA)
- ODE $\rightarrow$ Device based implementation of Runge-Kutta and BDF time integrators
- Sparse

Latest development

BLAS improvements

BLAS completeness (no band/packed implementation)

- BLAS 1: complete
- BLAS 2: complete
- BLAS 3: need SYMM, HEMM and rank k/2k updates

Execution Space interface (stream/queue execution) supported for all kernels

- `KokkosBlas::myBlasKernels(const ExecutionSpace& space, ...)`

SYCL backend support

- Expending oneAPI MKL support, building/testing on Aurora

Example calling BLAS `gemm` with an execution space instance to run the kernel in a stream.

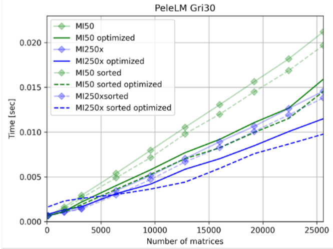
```cpp
using execution_space = Kokkos::Cuda;

auto instances =
    Kokkos::Experimental::partition_space(execution_space(), 1, 1);
KokkosBlas::gemm(instances[0], tA, tB, alpha, A1, B1, beta, C1);
KokkosBlas::gemm(instances[1], tA, tB, alpha, A2, B2, beta, C2);
Kokkos::fence(); // All results available after this point.
```

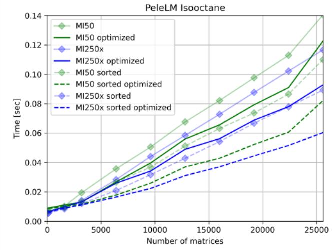
Sparse

- Sparse format conversion: coo2csr, csc2csr
- SpGEMM supports reuse and has improved TPL support (MKL, cuSPARSE, rocSPARSE)
- New incomplete factorization algorithms
  - parILUt algorithm (iterative computation of L and U)
  - MDF(0) reorders matrix rows to minimize discarded fill
  - Stream version of ILU(k) and SpTRSV
- CrsMatrix sort and merge (also graph version)
- SpMV improved for BsrMatrix

Sparse Batched algorithms

- Algorithms implemented
  - Linear algebra (SpMV, Vector operations...)
  - Iterative solver (CG, GMRES)
  - Preconditioner (Jacobi)
- Launch parameters tunned by architecture
  - NVIDIA V100
  - AMD MI50/MI250





ODE

New component for time integration algorithms (still experimental)

- Explicit integrators
  - Runge Kutta (orders 1 to 5)
  - Various schemes for stability (Fehlberg 45, Cash-Karp, Dormand-Prince)
  - Time adaptive
  - DOP 853 upcoming...
- Implicit integrators
  - BDF (orders 1 to 5)
  - Time and order adaptive implementation
  - Adams-Moulton method upcoming...
- Methods are implemented to be called in innermost parallel level (SerialInternal)

Upcoming work

Next features

Short term goals:

- Block version of ILU(K)
- Adding BLIS TPL for BLAS implementation
- LAPACK features and TPL support will expand greatly
- Improve stream preconditioners: balancing, reordering...
- Expend oneAPI MKL utilization

Next features

Longer term goals:

- automated architecture based tunning
- more performance monitoring using benchmarks
- documentation improvements
- handle refactor
- full BLAS layer

Collaborations

Currently Kokkos Kernels is primarily developed at Sandia.
More external collaborations are welcomed

- Join us on kokkosteam.slack.com in the #kokkos-kernels chanel
- Create an issue or pull request on github.com/kokkos/kokkos-kernels
- If desirable, an external collaboration meeting can be setup

Conclusion

Thank you for your attention! Any questions?