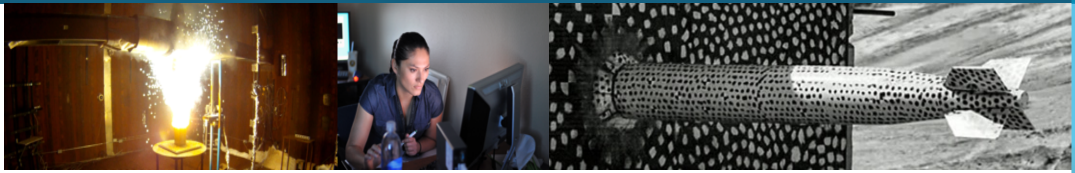
LAMMPS KOKKOS Package: The quest for performance portable MD



Stan Moore

2019 LAMMPS Workshop

Albuquerque, NM

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & 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-NA0003525.

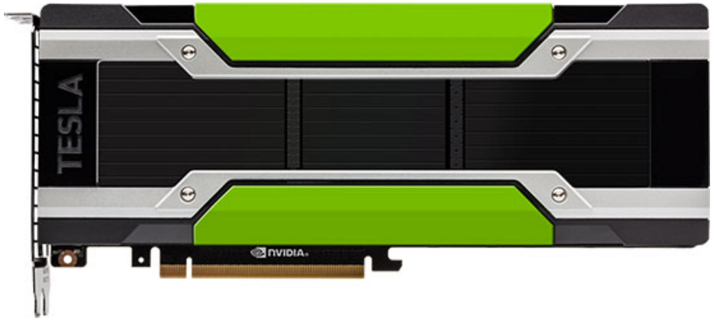
Supercomputer Hardware Trends

Currently, half of the top ten supercomputers use NVIDIA GPUs, one more has Intel Xeon Phi (many-core) accelerators, according to the June 2019 Top500 List (https://www.top500.org)

In the future, other large supercomputers will have accelerators or non-conventional hardware (NERSC Perlmutter—NVIDIA GPUs, ANL Aurora—Intel Xe, ORNL Frontier—AMD GPUs)

Special code (beyond regular C++ and MPI in LAMMPS) is required to run well on GPUs and many-core CPUs (e.g. CUDA, OpenMP; likely true for future hardware as well

Hardware and corresponding programming languages are ever changing, how to keep LAMMPS up to date?



http://www.nvidia.com/object/tesla-p100.html

Kokkos Performance Portability Library

Kokkos is an abstraction layer between programmer and next-generation platforms

Allows the same LAMMPS C++ code to run on multiple hardware (GPU, Xeon Phi, etc.)

Kokkos consists of two main parts:

1. Parallel dispatch—threaded kernels are launched and mapped onto backend languages such as CUDA or OpenMP
2. Kokkos views—polymorphic memory layouts that can be optimized for a specific hardware

Used on top of existing MPI parallelization (MPI + X)

Open-source, can be downloaded at https://github.com/kokkos/kokkos

In a nutshell, the goal of Kokkos is to future-proof LAMMPS to allow it to run on future hardware without total re-write (i.e. change Kokkos library for new hardware, not LAMMMPS)

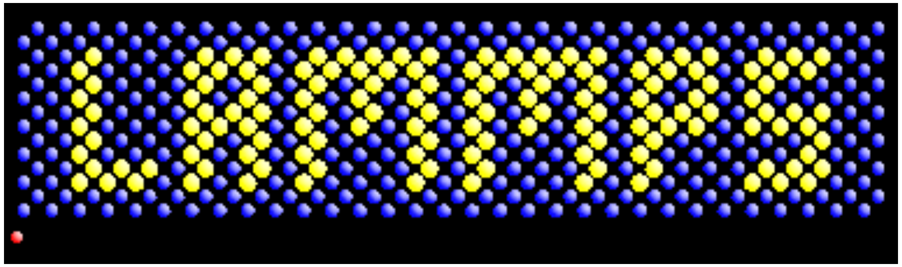
LAMMPS KOKKOS Package

Optional add-on package in LAMMPS

Developed by Stan Moore, Christian Trott, and others

Goal is that everything in LAMMPS (pair, fixes, computes, etc.) runs on the GPU , with minimal data transfer from GPU to CPU if possible

Different than the GPU package, which only runs the pair-style and a few other computations on the GPU



LAMMPS KOKKOS Package

8 atom styles: angle, atomic, bond, charge, dpd, full, molecular, sphere (along with hybrid)

44 pair styles: buck/coul/cut, buck/coul/long, buck, coul/cut, coul/debye, coul/dsf, coul/long, coul/wolf, dpd/fdt/energy, eam/alloy, eam/fs, eam, exp6/rx, gran/hooke/history, hybrid/overlay, lj/charmm/coul/charmm/implicit, lj/charmm/coul/charmm, lj/charmm/coul/long, lj/class2/coul/cut, lj/class2/coul/long, lj/class2, lj/cut/coul/cut, lj/cut/coul/debye, lj/cut/coul/dsf, lj/cut/coul/long, lj/cut, lj/expand, lj/gromacs/coul/gromacs, lj/gromacs, lj/sdk, morse, multi/lucy/rx, reaxc, snap, sw, table, table/rx, tersoff, tersoff/mod, tersoff/zbl, vashishta, yukawa, zbl

22 fix styles: deform, dpd/energy, enforce2d, eos/table/rx, freeze, gravity, langevin, momentum, neigh/history, nph, npt, nve, nve/sphere, nvt, property/atom, qeq/reax, reaxc/bonds, reaxc/species, rx, setforce, shardlow, wall/lj93, wall/reflect

1 compute style: temp

3 bond styles: class2, fene, harmonic

4 angle styles: charmm, class2, cosine, harmonic

3 dihedral styles: charmm, class2, opls

2 improper style: class2, harmonic

1 kspace style: pppm

Compiling and Running KOKKOS Package

Kokkos library is already included with LAMMPS, no need to download:

- In lammps/src directory, “make yes-kokkos”
- Build with /src/MAKE/OPTIONS/Makefile.kokkos_omp or Makefile.kokkos_cuda_mpi
- Must use a c++11 compatible compiler (gcc 4.7.2 or higher, intel 14.0 or higher, CUDA 7.5 or higher)
- Also CMake option, see docs

No changes to input script needed, just add a few command line args:

- Run with 4 MPI tasks and 4 GPUs: “mpiexec -np 4 ./lmp_exe -in in.lj **-k on g 4 -sf kk**”
- Run with 4 OpenMP threads: “./lmp_exe -in in.lj **-k on t 4 –sf kk**”

See Kokkos docs:
https://lammps.sandia.gov/doc/Speed_kokkos.html

Recent Performance Work

2x improvement of **small Lennard-Jones systems** (~1000 atoms) on a single V100 GPU

5x improvement of **SNAP potential** on V100 GPUs (regular CPU version is also over 2x faster on CPUs than before)

2x improvement of **PPPM long-range electrostatics** on a V100 GPU (to be released soon)

Improved **OpenMP threading performance** by adding data duplication option (helped several pair styles, from LJ to ReaxFF)

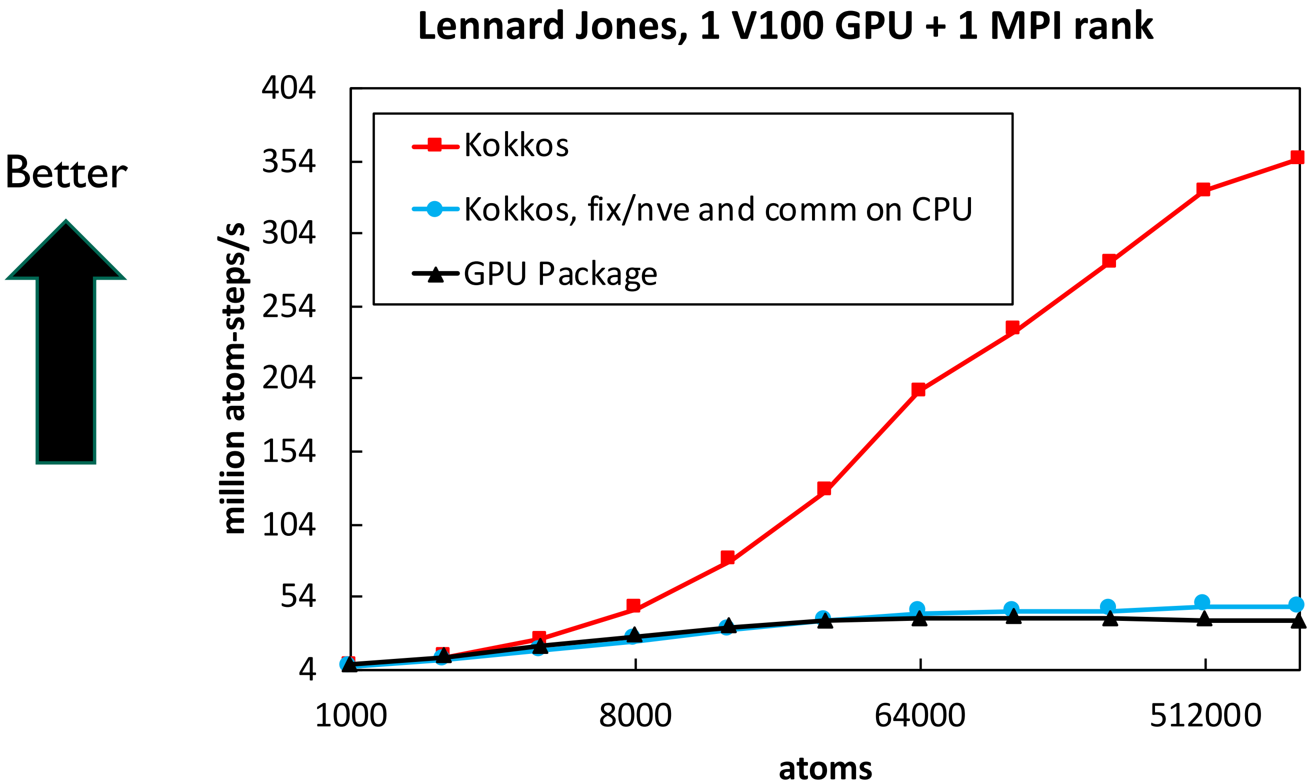
Performance comparison with GPU package

Double-precision only

Kokkos uses special fused MPI comm kernel when running on a single GPU

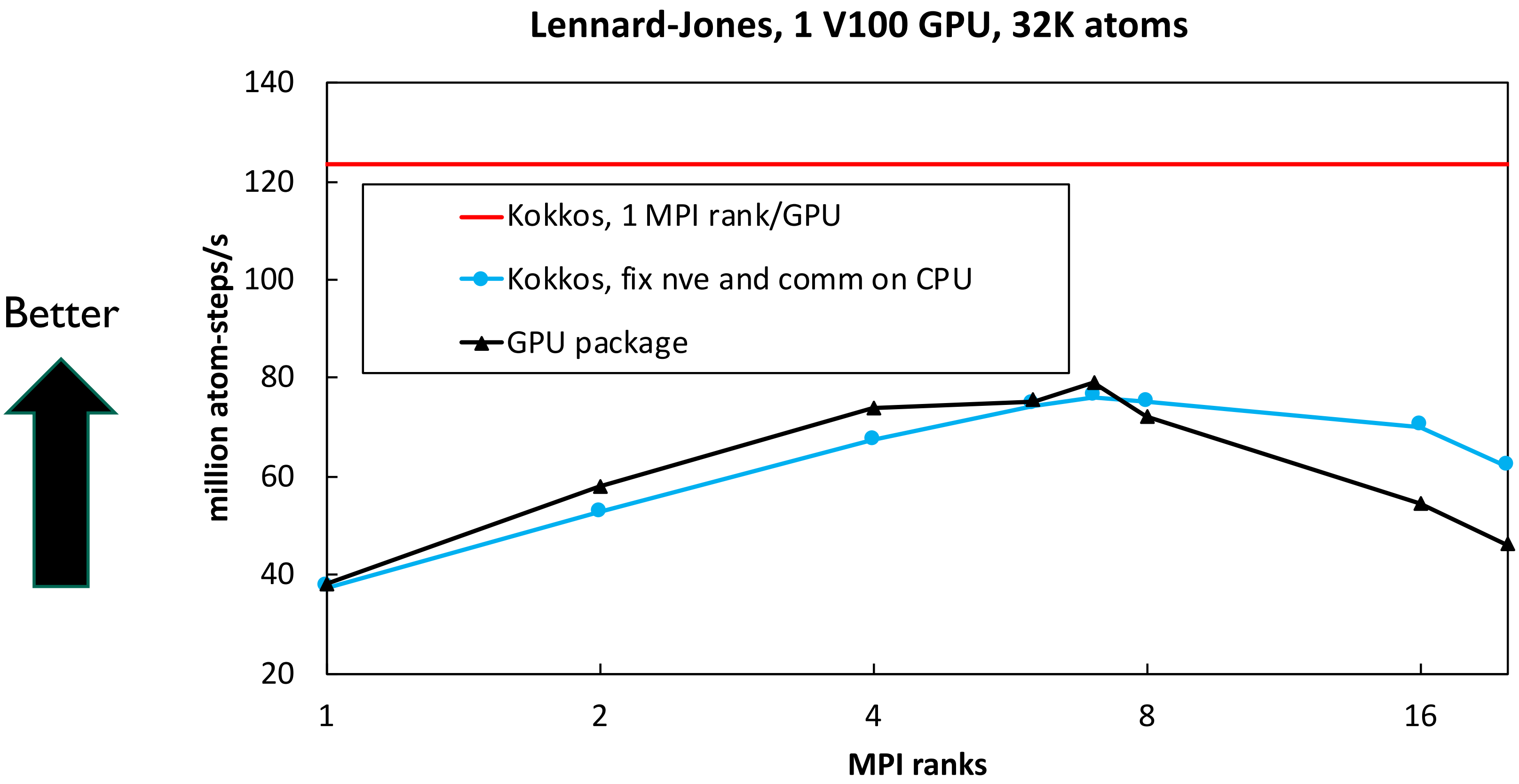
Performance penalty for moving atom data between GPU and CPU

Integrator is running serially on CPU



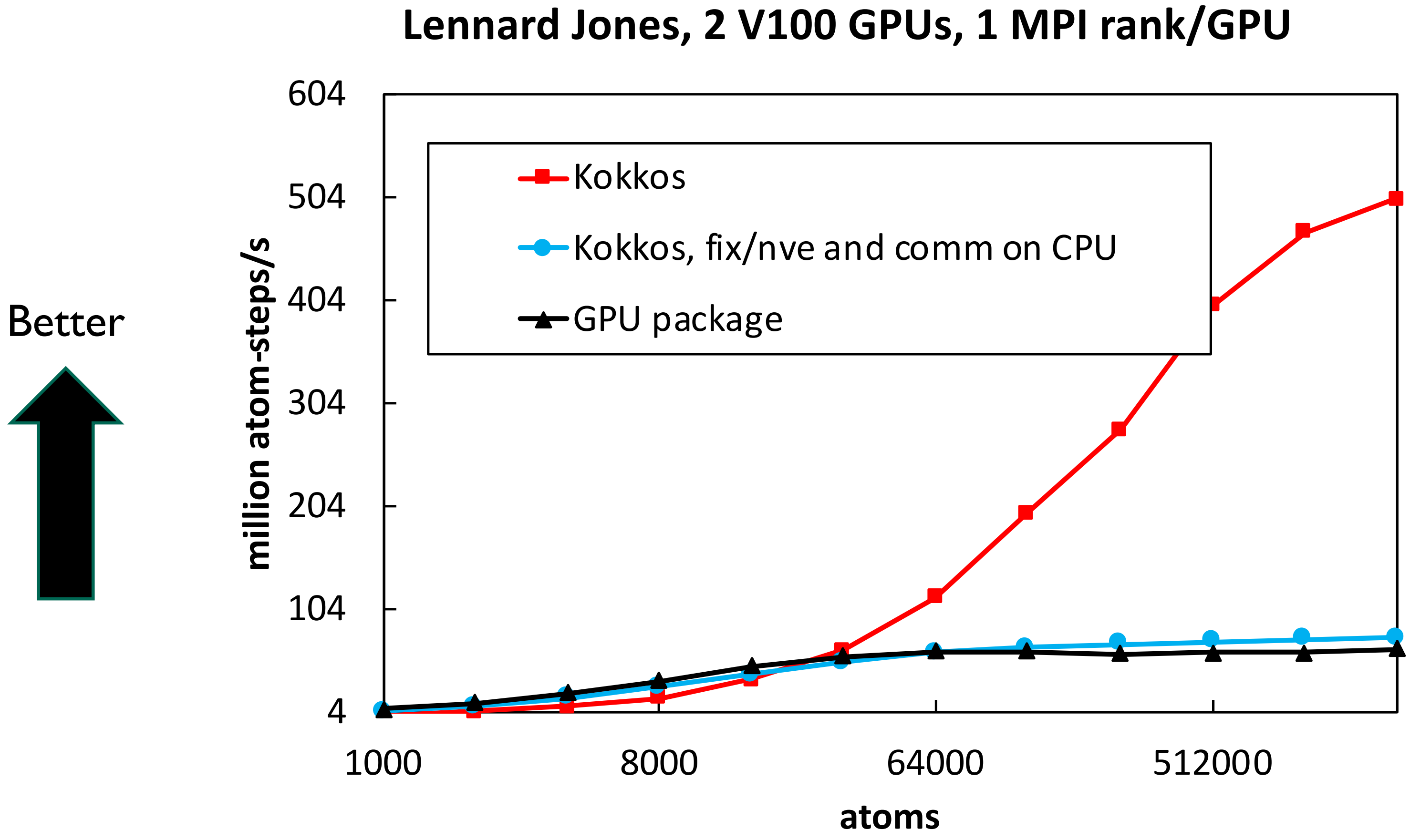
Multiple MPI ranks per GPU

MUST use CUDA MPS with multiple MPI ranks per GPU to get good performance



Two MPI ranks

Higher overhead for Kokkos due to latency of launching multiple kernels to pack communication buffers

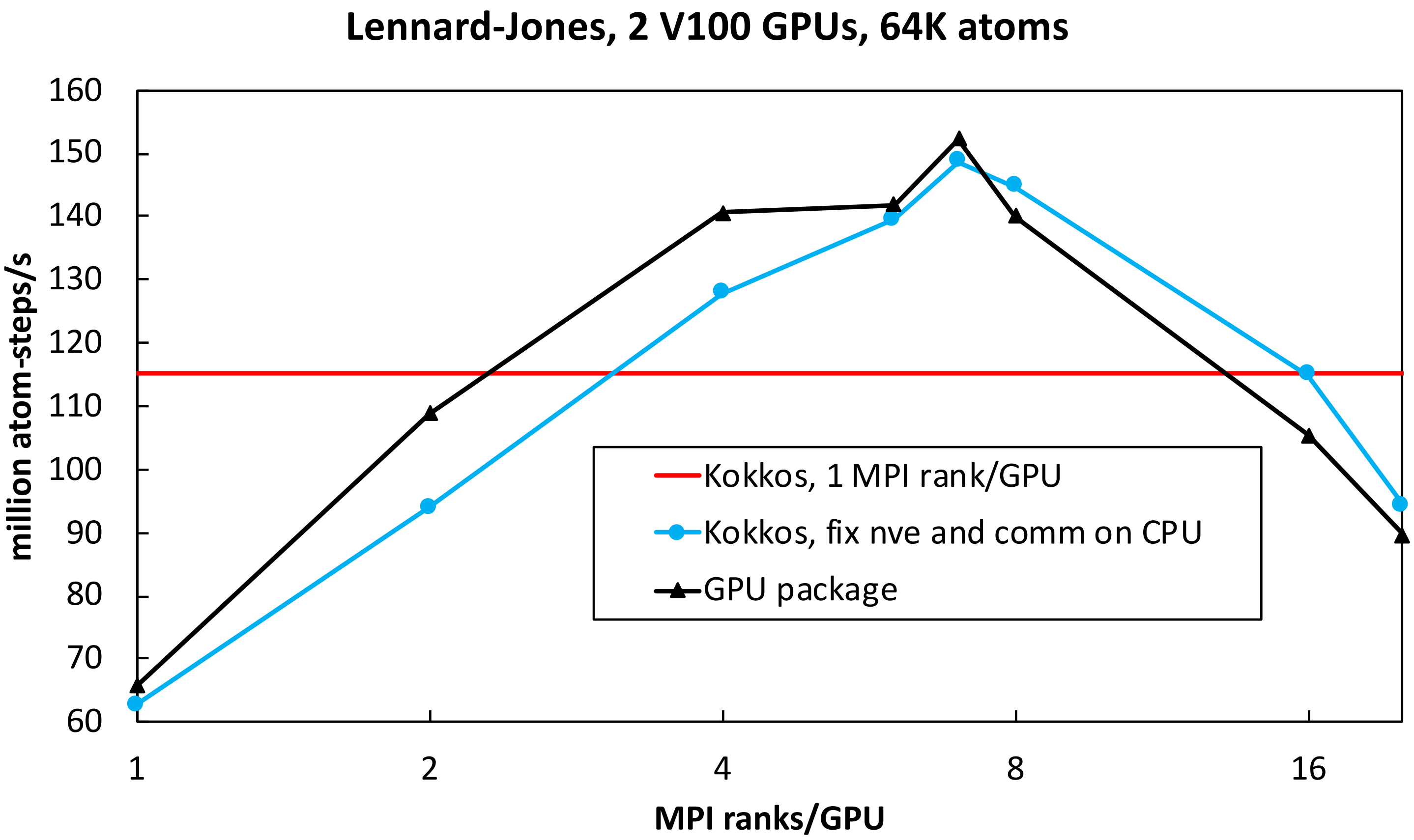


Performance comparison with GPU package

Double-precision only

Using more atoms/GPU probably will probably have different behavior

Better

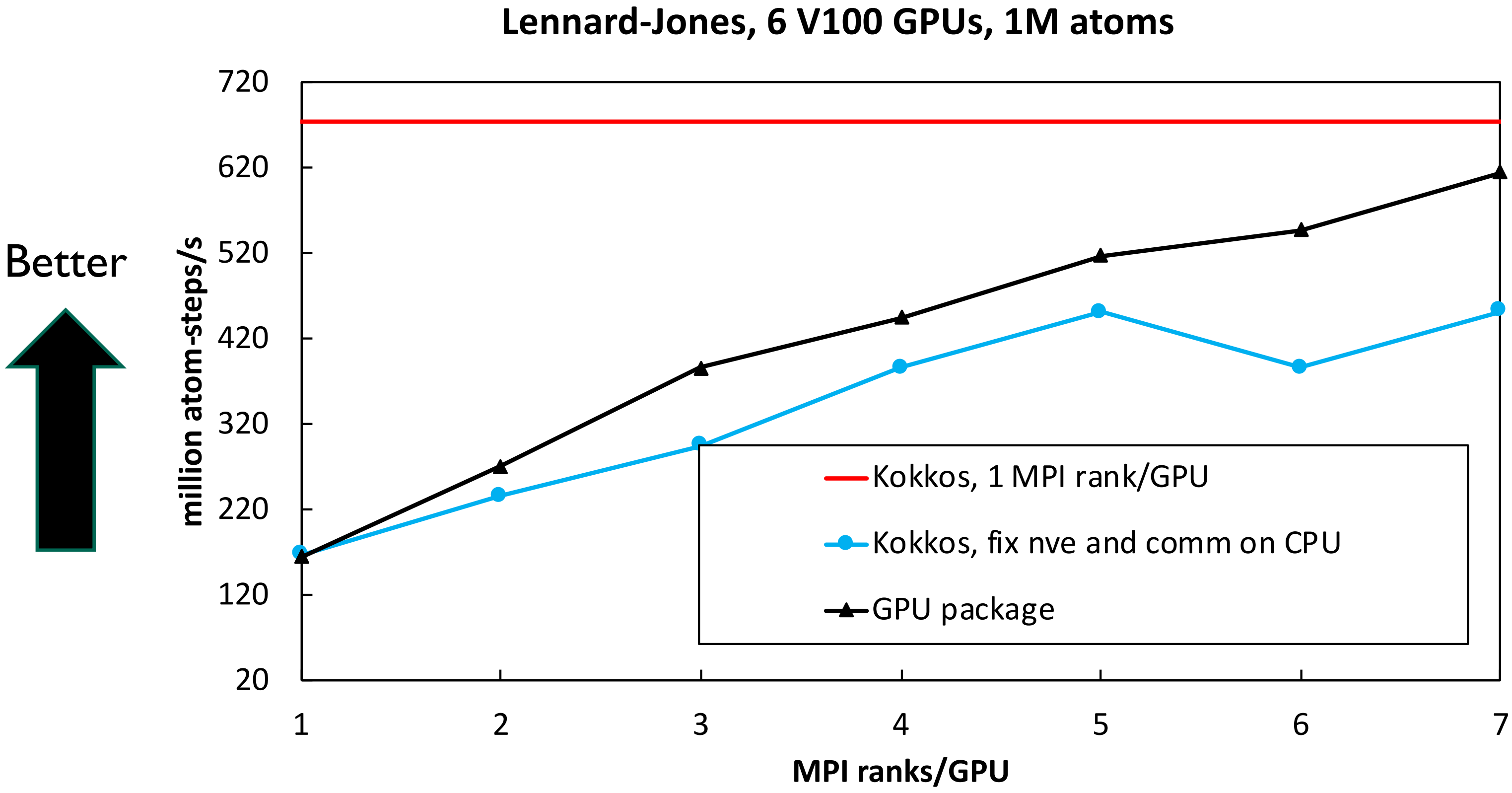


Performance comparison with GPU package

Double-precision only

Full Summit node

Using pinned memory may help Kokkos with integrator and comm on host CPU



ReaxFF

3 versions in LAMMPS:

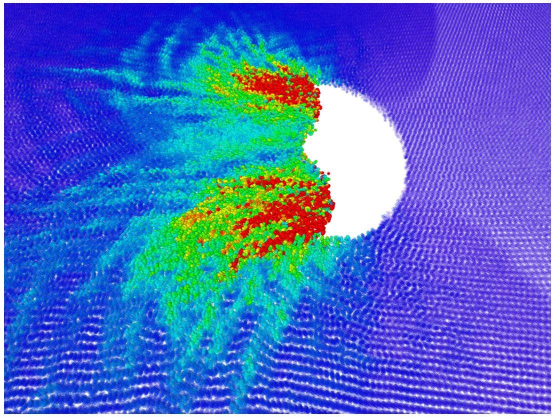
- USER-REAXC
- KOKKOS
- USER-OMP

KOKKOS CUDA version can run on NVIDIA GPUs

KOKKOS version more memory robust, should be used if getting memory errors, or with fix GCMC

KOKKOS MPI-only version faster than USER-REAXC package, at least in some cases

USER-OMP version probably a little better for OpenMP on CPUs (need to benchmark performance)



Limitations of the Kokkos package

If a style isn't in the KOKKOS package, it won't be accelerated. Also may need to transfer atom data back and forth between CPU and GPU every timestep, which reduces performance

USER-INTEL, USER-OMP, and OPT packages can give better vectorization on Intel hardware leading to better performance

GPU and USER-INTEL packages support single and mixed precision, KOKKOS package only supports double precision (but working on fixing this soon)

Conclusions

Computer hardware is becoming more complicated, requiring special code to run well

Kokkos library: goal is performance portability for current and future hardware

LAMMPS KOKKOS package allows LAMMPS to run on NVIDIA GPUs and Intel many-core CPUs, and will also support future supercomputers

Give KOKKOS package a try, post questions or issues to the LAMMPS mail list