# LAMMPS Developer **Discussion**

Steve Plimpton & Axel Kohlmeyer

LAMMPS Workshop
24 February 2010

1. GPU, multi-core, other optimizations
2. More efficient long-range Coulombics
3. Support for users and developers
4. Software engineering ideas
5. Other topics ??

# GPU Enhancements

- One effort by Paul C, Mike B, Axel K, ORNL folks
  - src/GPU package
  - pair_style lj/cut, pair_style gayberne, more to follow
  - neighbor listing now on GPU
  - 5-10x speed-up

- One effort by Christian Muller group at Technische U, Germany
  - verlet/cuda, fix/cuda, pair/cuda, compute/cuda, pppm/cuda, etc
  - goal: keep data on GPU as much as possible
  - convert pairs, fixes, etc, one at a time
  - script with non-converted commands will move data back to CPU
  - will become src/USER-CUDA or possibly their own WWW site?

- Could we adapt other more expensive potentials for GPUs (a la GayBerne)
- Further talks/discussion on Friday, feedback?

# Multi-core Support

- Can run LAMMPS now in all-MPI mode (one MPI process per core)

- Typically as good or better performance than Open-MP or threads
  on modest core count

- What happens as cores → 16, 32, 128, etc ?
  MPI at coarse-grain node level
  shared-memory versions of kernels (pair, neighbor) to use cores
  could possibly exploit CPU cores and GPU as co-procs in hybrid manner

- Axel's ideas:
  hi-level threading library
  have alternate versions of pair styles built on top of library

# Speed Issues

- LAMMPS is slower than NAMD, Gromacs, Desmond for bio problems
  - 2x for NAMD (version 2.6), same speed on LJ ?
  - 5-6x for Gromacs (?)
  - not sure for Desmond

- Possible reasons for serial performance difference
  - need to quantify (breakdown):
    - short-cuts for water
    - assembly versions of pair potentials, sqrt() trick
    - short-cuts for long-range Coulombics
- Possible reasons for parallel scalability difference:
  - more efficient PPPM (SPME) in parallel
    - 2 FFTs per step instead of 4
    - run on different processors than short-range
  - more clever load-balancing on high processor counts
    - NAMD via CHARM++, Gromacs/Desmond via DE Shaw methods

# More efficient Long-range Coulombics

- Axel's systems with relatively few charges
- Alternates to PPPM:
  fast multiipole, SPME, multi-grid, real-space formulations
- Look at doing PPPM in tandem with pairwise on separate set of processors
- Long-range van der Waals
  kspace_style ewald/n
  no equivalent for PPPM
- Reduced communication in FFTs → better scalabiity

# Support for Users

- Continuous patch/release strategy for bug fixes and new features:
  - do you like it?
  - do you update your version frequently?
  - do we break scripts too often?
  - separate releases for bug-fixes and new features?
  - code & tarballs now print/contain most current date
  - old tarballs now available for download on WWW site

- Could we make LAMMPS easier to build?

- Repository of input scripts, how-to discussions, FAQ, Wiki
  - separate from mail list
  - Sandia rule:  no external posting of content to Sandia WWW pages
  - should we have a WWW form for posting scripts, your how-to helps ?
  - should we have an off-site Wiki ?

# Getting your Code added to main LAMMPS

- Why?

  good for you: users/collaborators, bug-finding and feature-additions, your name on top of the “authors” page

  good for LAMMPS

- Lo energy barrier mode:

  conform to the Style model

  just add class(es) that meet the interface

  include doc pages

- Hi energy barrier mode:

  change existing code in other Styles

  worst is to require changes to internal core of LAMMPS

  I am a barrier, both in time and resistance to change (speed, fragility)

- Never hurts to ask for advice up-front, may save you effort & rewriting

# 3 Modes for releasing your code in LAMMPS

(1) Code we understand and support:

becomes part of main LAMMPS or standard package

must code to the LAMMPS look and feel:

no STL, no C++ style I/O, operate like other LAMMPS styles do,

error checking, doc pages in LAMMPS style, works well in parallel

(2) User package:

just meet the Style interface, we never look at your code, easy upgrade

need one HTML doc page (per Style) that can link to your own doc files

user questions and bugs are referred to you

src/USER-ATC is a good example

(3) Empty user package:

same as (2), but users get files/doc/help from your WWW site

you are responsible for everything, LAMMPS just enables easy build

some users have requested this (CUDA GPU, granular/fluids)

# Support for Developers

- SVN repository is internal
  should we have an external, more open repository ?

- Other suggestions for simplifying/speeding the development process,
  and release of your add-on code ?

# Software Engineering

- Ideas for hooking to other tools/libraries ?
  - real-time or pre/post processing
- Extending LAMMPS library interface ?
- Are there C++ ideas we should be exploiting ?
  - need to be vanilla to be portable to high-end platforms
- Builders for molecular systems or heterogenous solids
  - Axel's experience with building in VMD
  - PACKMOL – put molecules in a volume

# Other Ideas

- Features you want to see?
- New problems you'd like to do with LAMMPS?
- Out-of-the-box ideas for improving LAMMPS?