A Quick Tour of LAMMPS

Steve Plimpton
Sandia National Labs
sjplimp@sandia.gov

5th LAMMPS Workshop Beginner Tutorial
August 2017 - Albuquerque, NM

Follow along with my slides at:
http://lammps.sandia.gov/workshops/Aug17/workshop.html

Sandia National Laboratories is a multi-mission 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-NA0003525. Presentation: SAND2017-7963C

Resources for learning LAMMPS

All on web site, most in distro tarball:

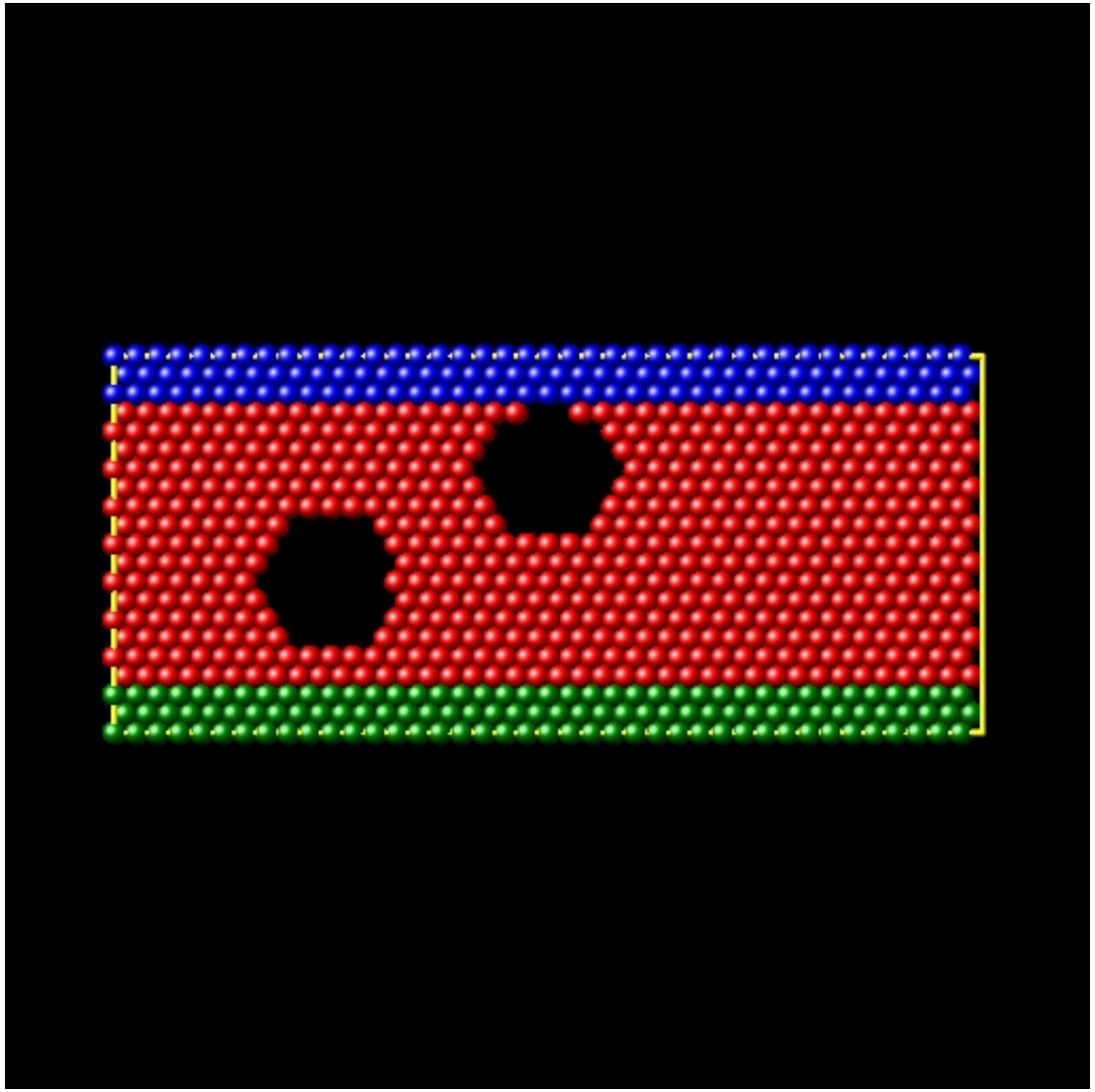
- Workshops (beginner tutorials) & Tutorials
- Features: list of features with links
- Glossary: MD terms ⇒ LAMMPS
- Manual:
  - Intro, Commands, Packages, Accelerating, Howto, Modifying, Errors sections
  - search bubble at top left of every page
- Commands
  - alphabetized command & style tables: one page per command
- Papers: find a paper similar to what you want to model
- Mail list: search archives (lammps-users, post Qs to it
  - http://lammps.sandia.gov/mail.html
- Examples: 50 sub-dirs under examples in distro
  - lower-case = simple, upper-case = more complex
  - many produce movies: http://lammps.sandia.gov/movies.html

Structure of typical input scripts

1. Units and atom style
2. Create simulation box and atoms
   - region, create_box, create_atoms, region commands
     - lattice command vs box units
   - read_data command
     - data file is a text file
     - look at examples/micelle/data.micelle
     - see read_data doc page for full syntax
3. Define groups
4. Set attributes of atoms: mass, velocity
5. Pair style for atom interactions
6. Fixes for time integration and constraints
7. Computes for diagnostics
8. Output: thermo, dump, restart
9. Run or minimize
10. Rinse and repeat (script executed one command at a time)

Obstacle example

input script = examples/obstacle/in.obstacle



Obstacle input script

1st section = setup box and create atoms

```
# 2d LJ obstacle flow

dimension 2
boundary p s p
atom_style atomic
neighbor 0.3 bin
neigh_modify delay 5

# create geometry

lattice hex 0.7
region box block 0 40 0 10 -0.25 0.25
create_box 3 box
create_atoms 1 box
```

Obstacle input script

2nd section = define potential and groups of atoms

```
# LJ potentials

pair_style lj/cut 1.12246
pair_coeff * * 1.0 1.0 1.12246

# define groups

region 1 block INF INF INF 1.25 INF INF
group lower region 1
region 2 block INF INF 8.75 INF INF INF
group upper region 2
group boundary union lower upper
group flow subtract all boundary

set group lower type 2
set group upper type 3
```

Obstacle input script

3rd section = set velocities and fixes

```
# initial velocities

mass * 1.0
compute mobile flow temp
velocity flow create 1.0 482748 temp mobile
fix 1 all nve
fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
fix_modify 2 temp mobile

# Poiseuille flow

velocity boundary set 0.0 0.0 0.0
fix 3 lower setforce 0.0 0.0 0.0
fix 4 upper setforce 0.0 NULL 0.0
fix 5 upper aveforce 0.0 -0.5 0.0
fix 6 flow addforce 1.0 0.0 0.0
```

Obstacle input script

4th section = create 2 obstacles to flow

```
# 2 obstacles

region void1 sphere 10 4 0 3
delete_atoms region void1
region void2 sphere 20 7 0 3
delete_atoms region void2

fix 7 flow indent 100 sphere 10 4 0 4
fix 8 flow indent 100 sphere 20 7 0 4
fix 9 all enforce2d
```

Obstacle input script

5th section: define output and run simulation (JPG, PNG, PPM)

```
# run

timestep 0.003
thermo 1000
thermo_modify temp mobile

#dump 1 all atom 100 dump.obstacle
dump 2 all image 500 image.*.ppm type type &
    zoom 1.6 adiam 1.5
dump_modify 2 pad 5

run 25000
```

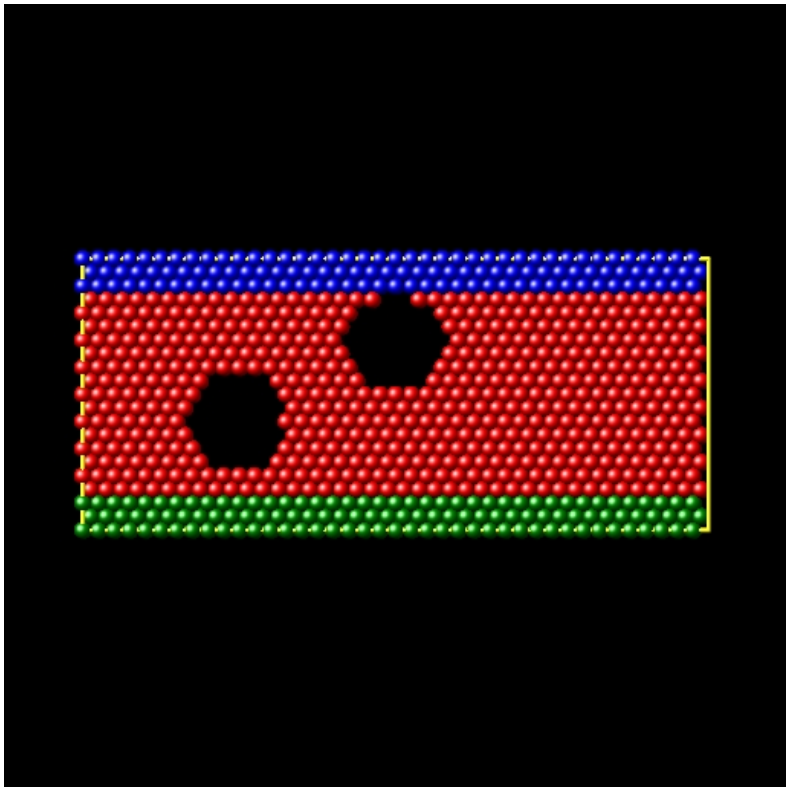
Make a movie

50 JPG or PNG or PPM files

- image.16500.ppm
- ImageMagick display
- Mac Preview

Make/view a movie

- ImageMagick
  convert *.ppm image.gif
- open in browser
  open -a Safari image.gif
- Mac QuickTime
  open image sequence
- Windows Media Player
- Ovito, VMD, AtomEye, ...



Obstacle example

Questions on input script?

Exercise:

run examples/obstacle/in.obstacle on your laptop
examine and visualize output

Examine screen output

```
LAMMPS (5 Oct 2016)
Lattice spacing in x,y,z = 1.28436 2.22457 1.28436
Created orthogonal box = (0 0 -0.321089)
            to (51.3743 22.2457 0.321089)
  4 by 1 by 1 MPI processor grid
Created 840 atoms
120 atoms in group lower
120 atoms in group upper
240 atoms in group boundary
600 atoms in group flow
Setting atom values ...
  120 settings made for type
Setting atom values ...
  120 settings made for type
Deleted 36 atoms, new total = 804
Deleted 35 atoms, new total = 769
```

Neighbor list and memory info

```
Neighbor list info ...
  update every 1 steps, delay 5 steps, check yes
  master list distance cutoff = 1.42246
  binsize = 0.71123, bins = 73 32 1
  1 neighbor lists, perpetual/occasional/extra = 1 0 0
  (1) pair lj/cut, perpetual
    attributes:  half, newton on
    pair build:  half/bin/atomonly/newton
    stencil:  half/bin/2d/newton
    bin:  standard
Setting up Verlet run ...
  Unit style :  lj
  Current step :  0
  Time step :  0.003
Per MPI rank memory allocation (avg) = 3.043 Mbytes
```

Thermo output

```
Step Temp E_pair E_mol TotEng Press Volume
0 1.0004177 0 0 0.68689281 0.46210058 1143.0857
1000 1 -0.32494012 0 0.36166587 1.2240503 1282.5239
2000 1 -0.37815616 0 0.30844982 1.0642877 1312.5691
...
...
...
24000 1 -0.40040333 0 0.28620265 0.94983886 1459.4461
25000 1 -0.37645924 0 0.31014674 1.0526044 1458.7191
Loop time of 0.815388 on 4 procs for 25000 steps
  with 769 atoms
Performance:  7947137.172 tau/day,
  30660.251 timesteps/s
98.7% CPU use with 4 MPI tasks x no OpenMP threads
```

Timing info

```
Loop time of 0.815388 on 4 procs for 25000 steps
  with 769 atoms

MPI task timing breakdown:
Section | min | avg | max | %varavg | %total
---------------------------------------------
Pair | 0.063 | 0.123 | 0.202 | 15.7 | 15.10
Neigh | 0.031 | 0.041 | 0.055 | 4.4 | 5.06
Comm | 0.149 | 0.231 | 0.288 | 11.0 | 28.42
Output | 0.001 | 0.001 | 0.001 | 0.0 | 0.01
Modify | 0.275 | 0.305 | 0.341 | 4.3 | 37.43
Other | ----- | 0.113 | ----- | --- | 13.91
```

Run statistics

Per-processor values at end of run

```
Nlocal:  192.25 ave 243 max 151 min
Histogram:  1 1 0 0 0 0 1 0 0 1
Nghost:  41.75 ave 43 max 39 min
Histogram:  1 0 0 0 0 0 0 1 0 2
Neighs:  408.5 ave 575 max 266 min
Histogram:  1 1 0 0 0 0 0 1 0 1

Total # of neighbors = 1634
Ave neighs/atom = 2.12484
Neighbor list builds = 1631
Dangerous builds = 1
```

Run statistics

Per-processor values at end of run

```
Nlocal:  192.25 ave 243 max 151 min
Histogram:  1 1 0 0 0 0 1 0 0 1
Nghost:  41.75 ave 43 max 39 min
Histogram:  1 0 0 0 0 0 0 1 0 2
Neighs:  408.5 ave 575 max 266 min
Histogram:  1 1 0 0 0 0 0 1 0 1

Total # of neighbors = 1634
Ave neighs/atom = 2.12484
Neighbor list builds = 1631
Dangerous builds = 1
```

Questions on output?

Common input script errors and debugging

- Command not recognized
  - ERROR: Unknown pair style gran/hooke (../force.cpp:246)
  - LAMMPS not built with style being used
  - many styles are in packages
  - lmp_mpi -h will list all styles included in build
- Syntax error in command
  - ERROR: Incorrect args for pair coefficients (file/line #)
  - Last command: pair_coeff * * 1.0 1.0 1.12246 9.0

Common input script errors and debugging

- Command not recognized
  - ERROR: Unknown pair style gran/hooke (../force.cpp:246)
  - LAMMPS not built with style being used
  - many styles are in packages
  - lmp_mpi -h will list all styles included in build
- Syntax error in command
  - ERROR: Incorrect args for pair coefficients (file/line #)
  - Last command: pair_coeff * * 1.0 1.0 1.12246 9.0
- Thermo output blows up immediately
  - usually due to bad atom coords and/or bad force field
  - often leads to lost or out-of-range atoms
- Don't start with too complex a script
  - debug an input script just like a computer program
  - start simple, add complexity one command at a time
  - can use print command to examine variables
  - use thermo/dump output and viz to verify correctness

Additional topics

- More complex input scripts
- Pre-processing to build a complex system
- Force fields: pair, bond, and kspace styles
- Fixex and computes
- Output
- Parallelization and performance
- Quick tour of more advanced topics
- Extending LAMMPS

Defining variables in input scripts

- Styles: index, loop, equal, atom, python, file, ...
  - variable x index run1 run2 run3 run4
  - variable x loop 100
  - variable x trap(f_JJ[3])*${scale}
  - variable x atom -(c_p[1]+c_p[2]+c_p[3])/(3*vol)

Defining variables in input scripts

- Styles: index, loop, equal, atom, python, file, ...
  - variable x index run1 run2 run3 run4
  - variable x loop 100
  - variable x trap(f_JJ[3])*${scale}
  - variable x atom -(c_p[1]+c_p[2]+c_p[3])/(3*vol)
- Formulas can be complex
  - see doc/variable.html
  - thermo keywords (temp, press, ...)
  - math operators & functions (sqrt, log, cos, ...)
  - group and region functions (count, xcm, fcm, ...)
  - various special functions (min, ave, trap, stride, stagger, ...)
  - per-atom vectors (x, vx, fx, ...)
  - output from computes, fixes, other variables
- Formulas can be time- and/or spatially-dependent

Using variables in input scripts

- Substitute in any command via $x or ${myVar}
- Immediate formula evaluation via $() syntax:
  - avoids need to define variable separately
  - variable xmid equal (xlo+xhi)/2
  - region 1 block $xmid EDGE INF INF EDGE EDGE
  - region 1 block $((xlo+xhi)/2) EDGE INF INF EDGE EDGE
- Next command increments a variable to next value
- Many commands allow variables as arguments
  - fix addforce 0.0 v_fy 1.0
  - dump_modify every v_foo
  - region sphere 0.0 0.0 0.0 v_radius
  - only if command doc page says so

Power tools for input scripts

- Filename options:
  - dump.*.% for per-snapshot or per-processor output
  - read_data data.protein.gz
  - read_restart old.restart.*
- If/then/else via if command
- Insert another script via include command
  - useful for long list of params

Power tools for input scripts

- Filename options:
  - dump.*.% for per-snapshot or per-processor output
  - read_data data.protein.gz
  - read_restart old.restart.*
- If/then/else via if command
- Insert another script via include command
  - useful for long list of params
- Looping via next and jump commands
- Invoke a shell command or external program
  - shell cd subdir1
  - shell my_analyze out.file $n ${param}
- Invoke Python from your script: doc/Section_python.html
  - pass LAMMPS data to Python, return values in variables
  - Python function can callback to LAMMPS
- Various ways to run multiple simulations from one script
  - see doc/Section_howto 6.4

Example script for multiple runs

Run 8 successive simulations on P processors:

```
variable r equal random(1,1000000000,58798)
variable a loop 8
variable t index 0.8 0.85 0.9 0.95 1.0 1.05 1.1 1.15
log log.$a
read data.polymer
velocity all create $t $r
fix 1 all nvt $t $t 1.0
dump 1 all atom 1000 dump.$a.*
run 100000
next t
next a
jump in.polymer
```

Example script for multiple runs

Run 8 successive simulations on P processors:

```
variable r equal random(1,1000000000,58798)
variable a loop 8
variable t index 0.8 0.85 0.9 0.95 1.0 1.05 1.1 1.15
log log.$a
read data.polymer
velocity all create $t $r
fix 1 all nvt $t $t 1.0
dump 1 all atom 1000 dump.$a.*
run 100000
next t
next a
jump in.polymer
```

Instead, run 8 simulations on 3 partitions until finished:

- change a & t to universe-style variables
- mpirun -np 12 lmp_linux -p 3x4 -in in.polymer

Pre-processing tools to build complex systems

LAMMPS does not build molecular systems or auto-magically assign force field parameters for you

Pre-processing tools to build complex systems

LAMMPS does not build molecular systems or auto-magically assign force field parameters for you

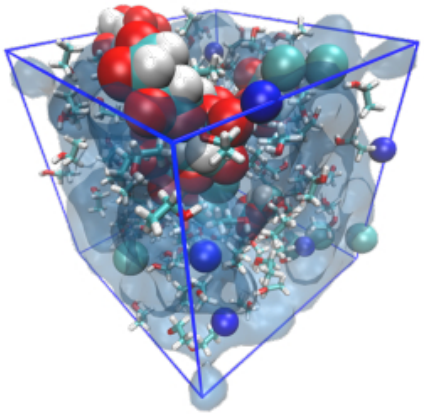
- Data file must include list of bonds, angles, etc
- Data file can include force field assignments
- Tools directory has converters for both steps
  - ch2lmp = CHARMM converter
  - amber2lmp = AMBER converter
  - msi2lmp = Accelrys converter

Pre-processing tools to build complex systems

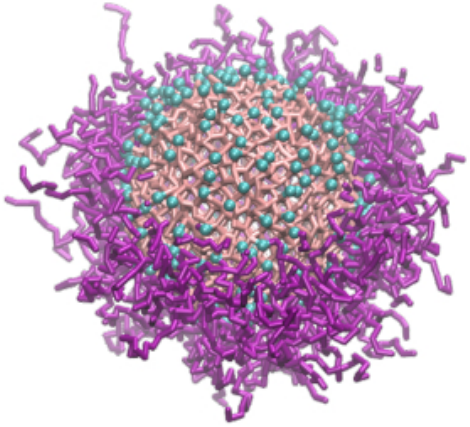
LAMMPS does not build molecular systems or auto-magically assign force field parameters for you

- Data file must include list of bonds, angles, etc
- Data file can include force field assignments
- Tools directory has converters for both steps
  - ch2lmp = CHARMM converter
  - amber2lmp = AMBER converter
  - msi2lmp = Accelrys converter
- Provided builders
  - Moltemplate (Andrew Jewett)
  - Pizza.py = chain and patch tools (Python)
- Builders that can create LAMMPS input
  - see website Pre/Post processing page for list
  - VMD TopoTools (Axel Kohlmeyer)
  - Avogadro, Packmol, ATB (Auto Topo Builder), EMC

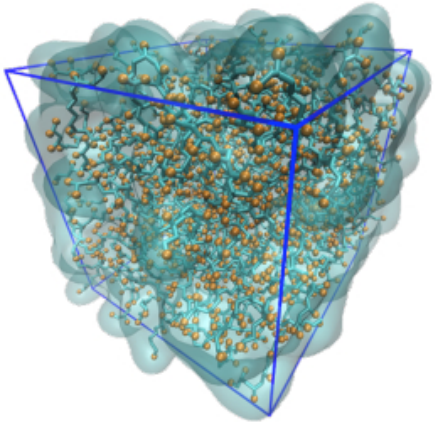
EMC builder tool - Wed PM breakout session



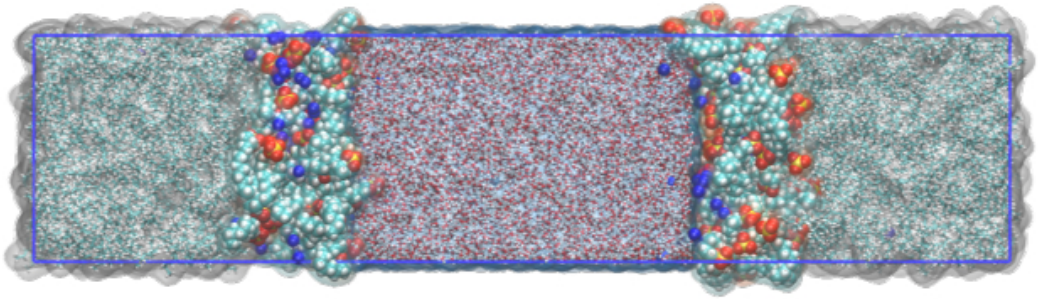
bulk



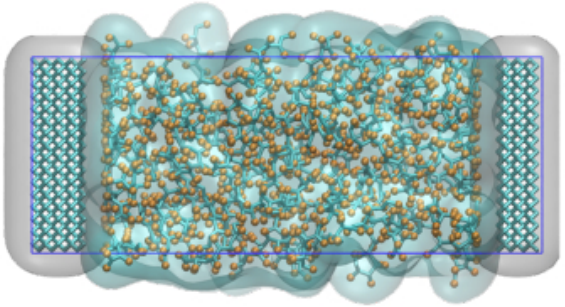
graft



copolymer



multiphase



surface

Pair styles

LAMMPS lingo for interaction potentials

Pair styles

LAMMPS lingo for interaction potentials

- A pair style can be true pair-wise or many-body
  - LJ, Coulombic, Buckingham, Morse, Yukawa, ...
  - EAM, Tersoff, REBO, ReaxFF, ...
- Bond/angle/dihedral/improper styles = permanent bonds

Pair styles

LAMMPS lingo for interaction potentials

- A pair style can be true pair-wise or many-body
  - LJ, Coulombic, Buckingham, Morse, Yukawa, ...
  - EAM, Tersoff, REBO, ReaxFF, ...
- Bond/angle/dihedral/improper styles = permanent bonds
- Variants optimized for CPU (many-core), KNL, GPU
  - OPT, GPU, USER-OMP, USER-INTEL, KOKKOS packages
  - lj/cut, lj/cut/gpu, lj/cut/kk, lj/cut/omp
  - see doc/Section_accelerate.html
- Coulomb interactions included in pair style
  - lj/cut, lj/cut/coul/cut, lj/cut/coul/wolf, lj/cut/coul/long
  - done to optimize inner loop

Categories of pair styles

- Solids
  - eam, eim, meam, adp, etc
- Bio and polymers
  - charmm, class2, gromacs, dreiding, etc
- Reactive
  - tersoff, bop, airebo, comb, reax/c, etc
- Coarse-grained
  - dpd, granular, sph, peri, colloid, lubricate, brownian, FLD
- Aspherical
  - gayberne, resquared, line, tri, etc

Categories of pair styles

- Solids
  - eam, eim, meam, adp, etc
- Bio and polymers
  - charmm, class2, gromacs, dreiding, etc
- Reactive
  - tersoff, bop, airebo, comb, reax/c, etc
- Coarse-grained
  - dpd, granular, sph, peri, colloid, lubricate, brownian, FLD
- Aspherical
  - gayberne, resquared, line, tri, etc

- Pair table for tabulation of any pair-wise interaction
- Pair hybrid style enables for hybrid models
  - polymers on metal
  - CNTs in water
  - solid-solid interface between 2 materials

Pair styles

See doc/Section_commands.html for full list

Annotated with (gikot) for 5 accelerated variants

none	hybrid	hybrid/overlay	adp (o)
airebo (o)	beck (go)	body	bop
born (go)	born/coul/long (cgo)	born/coul/long/cs	born/coul/msm (o)
born/coul/wolf (go)	brownian (o)	brownian/poly (o)	buck (cgko)
buck/coul/cut (cgko)	buck/coul/long (cgko)	buck/coul/long/cs	buck/coul/msm (o)
buck/long/coul/long (o)	colloid (go)	comb (o)	comb3
coul/cut (gko)	coul/debye (gko)	coul/dsf (gko)	coul/long (gko)
coul/long/cs	coul/msm	coul/streitz	coul/wolf (ko)
dpd (o)	dpd/tstat (o)	dsmc	eam (cgkot)
eam/alloy (cgkot)	eam/fs (cgkot)	eim (o)	gauss (go)
gayberne (gio)	gran/hertz/history (o)	gran/hooke (co)	gran/hooke/history (o)
hbond/dreiding/lj (o)	hbond/dreiding/morse (o)	kim	lcbop
line/lj (o)	lj/charmm/coul/charmm (cko)	lj/charmm/coul/charmm/implicit (cko)	lj/charmm/coul/long (cgiko)
lj/charmm/coul/msm	lj/class2 (cgko)	lj/class2/coul/cut (cko)	lj/class2/coul/long (cgko)
lj/cut (cgikot)	lj/cut/coul/cut (cgko)	lj/cut/coul/debye (cgko)	lj/cut/coul/dsf (gko)
lj/cut/coul/long (cgikot)	lj/cut/coul/msm (go)	lj/cut/dipole/cut (go)	lj/cut/dipole/long
lj/cut/tip4p/cut (o)	lj/cut/tip4p/long (ot)	lj/expand (cgko)	lj/gromacs (cgko)
lj/gromacs/coul/gromacs (cko)	lj/long/coul/long (o)	lj/long/dipole/long	lj/long/tip4p/long
lj/smooth (co)	lj/smooth/linear (o)	lj96/cut (cgo)	lubricate (o)
lubricate/poly (o)	lubricateU	lubricateU/poly	meam (o)
mie/cut (o)	morse (cgot)	nb3b/harmonic (o)	nm/cut (o)
nm/cut/coul/cut (o)	nm/cut/coul/long (o)	peri/eps	peri/lps (o)
peri/pmb (o)	peri/ves	polymorphic	reax
rebo (o)	resquared (go)	snap	soft (go)
sw (cgkio)	table (gko)	tersoff (cko)	tersoff/mod (ko)
tersoff/zbl (ko)	tip4p/cut (o)	tip4p/long (o)	tri/lj (o)
yukawa (go)	yukawa/colloid (go)	zbl (o)	

ıl pair styles in USER packages, which can be used if LAMMPS is built with the appropriate package.

awpmd/cut	coul/cut/soft (o)	coul/diel (o)	coul/long/soft (o)
eam/cd (o)	edip (o)	eff/cut	gauss/cut
list	lj/charmm/coul/long/soft (o)	lj/cut/coul/cut/soft (o)	lj/cut/coul/long/soft (o)
lj/cut/dipole/sf (go)	lj/cut/soft (o)	lj/cut/tip4p/long/soft (o)	lj/sdk (gko)
lj/sdk/coul/long (go)	lj/sdk/coul/msm (o)	lj/sf (o)	meam/spline
meam/sw/spline	quip	reax/c	sph/heatconduction
sph/idealgas	sph/lj	sph/rhosum	sph/taitwater

Pair styles

See doc/pair_style.html for one-line descriptions

- pair_style none - turn off pairwise interactions
- pair_style hybrid - multiple styles of pairwise interactions
- pair_style hybrid/overlay - multiple styles of superposed pairwise interactions

- pair_style adp - angular dependent potential (ADP) of Mishin
- pair_style airebo - AIREBO potential of Stuart
- pair_style beck - Beck potential
- pair_style body - interactions between body particles
- pair_style bop - BOP potential of Pettifor
- pair_style born - Born-Mayer-Huggins potential
- pair_style born/coul/long - Born-Mayer-Huggins with long-range Coulombics
- pair_style born/coul/msm - Born-Mayer-Huggins with long-range MSM Coulombics
- pair_style born/coul/wolf - Born-Mayer-Huggins with Coulombics via Wolf potential
- pair_style brownian - Brownian potential for Fast Lubrication Dynamics
- pair_style brownian/poly - Brownian potential for Fast Lubrication Dynamics with polydispersity
- pair_style buck - Buckingham potential
- pair_style buck/coul/cut - Buckingham with cutoff Coulomb
- pair_style buck/coul/long - Buckingham with long-range Coulombics
- pair_style buck/coul/msm - Buckingham long-range MSM Coulombics
- pair_style buck/long/coul/long - long-range Buckingham with long-range Coulombics
- pair_style colloid - integrated colloidal potential
- pair_style comb - charge-optimized many-body (COMB) potential
- pair_style coul/cut - cutoff Coulombic potential
- pair_style coul/debye - cutoff Coulombic potential with Debye screening
- pair_style coul/dsf - Coulombics via damped shifted forces
- pair_style coul/long - long-range Coulombic potential
- pair_style coul/msm - long-range MSM Coulombics
- pair_style coul/wolf - Coulombics via Wolf potential
- pair_style dipole/cut - point dipoles with cutoff
- pair_style dpd - dissipative particle dynamics (DPD)
- pair_style dpd/tstat - DPD thermostatting
- pair_style dsmc - Direct Simulation Monte Carlo (DSMC)
- pair_style eam - embedded atom method (EAM)
- pair_style eam/alloy - alloy EAM
- pair_style eam/fs - Finnis-Sinclair EAM
- pair_style eim - embedded ion method (EIM)
- pair_style gauss - Gaussian potential
- pair_style gayberne - Gay-Berne ellipsoidal potential
- pair_style gran/hertz/history - granular potential with Hertzian interactions
- pair_style gran/hooke - granular potential with history effects
- pair_style gran/hooke/history - granular potential without history effects

Relative CPU cost of potentials

See http://lammps.sandia.gov/bench.html#potentials for details
Can estimate how long your simulation will run

Potential	System	Atoms	Timestep	CPU	LJ Ratio
Granular	chute flow	32000	0.0001 tau	5.08e-7	0.34x
FENE bead/spring	polymer melt	32000	0.012 tau	5.32e-7	0.36x
Lennard-Jones	LJ liquid	32000	0.005 tau	1.48e-6	1.0x
DPD	pure solvent	32000	0.04 tau	2.16e-6	1.46x
EAM	bulk Cu	32000	5 fmsec	3.59e-6	2.4x
Tersoff	bulk Si	32000	1 fmsec	6.01e-6	4.1x
Stillinger-Weber	bulk Si	32000	1 fmsec	6.10e-6	4.1x
EIM	crystalline NaCl	32000	0.5 fmsec	9.69e-6	6.5x
SPC/E	liquid water	36000	2 fmsec	1.43e-5	9.7x
CHARMM + PPPM	solvated protein	32000	2 fmsec	2.01e-5	13.6x
MEAM	bulk Ni	32000	5 fmsec	2.31e-5	15.6x
Peridynamics	glass fracture	32000	22.2 nsec	2.42e-5	16.4x
Gay-Berne	ellipsoid mixture	32768	0.002 tau	4.09e-5	28.3x
AIREBO	polyethylene	32640	0.5 fmsec	8.09e-5	54.7x
COMB	crystalline SiO2	32400	0.2 fmsec	4.19e-4	284x
eFF	H plasma	32000	0.001 fmsec	4.52e-4	306x
ReaxFF	PETN crystal	16240	0.1 fmsec	4.99e-4	337x
ReaxFF/C	PETN crystal	32480	0.1 fmsec	2.73e-4	185x
VASP/small	water	192/512	0.3 fmsec	26.2	17.7e6
VASP/medium	CO2	192/1024	0.8 fmsec	252	170e6
VASP/large	Xe	432/3456	2.0 fmsec	1344	908e6

Bond styles (also angle, dihedral, improper)

- Used for molecules with fixed bonds
  - Fix bond/break and bond_style quartic can break them
- To learn what bond styles LAMMPS has ...
  where would you look?

Bond styles (also angle, dihedral, improper)

- Used for molecules with fixed bonds
  - Fix bond/break and bond_style quartic can break them
- To learn what bond styles LAMMPS has ...
  where would you look?
- doc/Section_commands.html or doc/bond_style.html

none	*zero*	*hybrid*	*class2 (ko)*
fene (iko)	*fene/expand (o)*	*harmonic (ko)*	*morse (o)*
nonlinear (o)	*quartic (o)*	*table (o)*	

These are additional bond styles in USER packages, which can be used if LAMMPS is built with the appropriate package.

harmonic/shift (o)	*harmonic/shift/cut (o)*	*oxdna/fene*	*oxdna2/fene*

- bond_style none - turn off bonded interactions
- bond_style hybrid - define multiple styles of bond interactions

- bond_style class2 - COMPASS (class 2) bond
- bond_style fene - FENE (finite-extensible non-linear elastic) bond
- bond_style fene/expand - FENE bonds with variable size particles
- bond_style harmonic - harmonic bond
- bond_style morse - Morse bond
- bond_style nonlinear - nonlinear bond
- bond_style quartic - breakable quartic bond
- bond_style table - tabulated by bond length

Long-range Coulombics

KSpace style in LAMMPS lingo, see doc/kspace_style.html

- Options:
  - traditional Ewald, scales as $O(N^{3/2})$
  - PPPM (like PME), scales as $O(N \log(N))$
  - MSM, scales as $O(N)$, lj/cut/coul/msm
- Additional options:
  - non-periodic, PPPM (z) vs MSM (xyz)
  - long-range dispersion (LJ)

Long-range Coulombics

KSpace style in LAMMPS lingo, see doc/kspace_style.html

- Options:
  - traditional Ewald, scales as $O(N^{3/2})$
  - PPPM (like PME), scales as $O(N \log(N))$
  - MSM, scales as $O(N)$, lj/cut/coul/msm
- Additional options:
  - non-periodic, PPPM (z) vs MSM (xyz)
  - long-range dispersion (LJ)
- PPPM is fastest choice for most systems
  - FFTs can scale poorly for large processor counts
- MSM can be faster for low-accuracy or large proc counts
- Pay attention to cutoff & accuracy settings
  - can affect performance dramatically (see timing output)
  - adjust Real vs KSpace work

Fixes

Most flexible feature in LAMMPS
Allow control of "what" happens "when" within each timestep

Loop over timesteps:

communicate ghost atoms

build neighbor list (once in a while)
compute forces
communicate ghost forces

output to screen and files

Fixes

Most flexible feature in LAMMPS
Allow control of "what" happens "when" within each timestep

Loop over timesteps:

fix initial	NVE, NVT, NPT, rigid-body integration
communicate ghost atoms	
fix neighbor	insert particles
build neighbor list (once in a while)	
compute forces	
communicate ghost forces	
fix force	SHAKE, langevin drag, wall, spring, gravity
fix final	NVE, NVT, NPT, rigid-body integration
fix end	volume & T rescaling, diagnostics
output to screen and files	

Fixes

- ∼200 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0

Fixes

- ∼200 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0
- To learn what fix styles LAMMPS has ...
  where would you look?

Fixes

- ∼200 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0
- To learn what fix styles LAMMPS has ...
  where would you look?
- doc/Section_commands.html or doc/fix.html

Fixes

- ~200 fixes in LAMMPS
- You choose what group of atoms to apply fix to
- Already saw some in obstacle example:
  - fix 1 all nve
  - fix 2 flow temp/rescale 200 1.0 1.0 0.02 1.0
  - fix 3 lower setforce 0.0 0.0 0.0
  - fix 5 upper aveforce 0.0 -0.5 0.0
  - fix 6 flow addforce 1.0 0.0 0.0
- To learn what fix styles LAMMPS has ...
  where would you look?
- doc/Section_commands.html or doc/fix.html
- If you familiarize yourself with fixes,
  you'll know many things LAMMPS can do
- Many fixes store output accessible by other commands
  - thermostat energy
  - force on wall
  - rigid body COM

Computes

- $\sim$120 computes in LAMMPS
- Calculate some property of system, in parallel
- Always for the current timestep
- To learn what compute styles LAMMPS has ...

Computes

- ∼120 computes in LAMMPS
- Calculate some property of system, in parallel
- Always for the current timestep
- To learn what compute styles LAMMPS has ...
  doc/Section_commands.html or doc/compute.html

angle	angle/local	angmom/chunk	body/local	bond	bond/local
centro/atom	chunk/atom	cluster/atom	cna/atom	com	com/chunk
contact/atom	coord/atom	damage/atom	dihedral	dihedral/local	dilatation/atom
dipole/chunk	displace/atom	erotate/asphere	erotate/rigid	erotate/sphere	erotate/sphere/atom
event/displace	global/atom	group/group	gyration	gyration/chunk	heat/flux
hexorder/atom	improper	improper/local	inertia/chunk	ke	ke/atom
ke/rigid	msd	msd/chunk	msd/nongauss	omega/chunk	orientorder/atom
pair	pair/local	pe	pe/atom	plasticity/atom	pressure
property/atom	property/local	property/chunk	rdf	reduce	reduce/region
rigid/local	slice	sna/atom	snad/atom	snav/atom	stress/atom
temp (k)	temp/asphere	temp/body	temp/chunk	temp/com	temp/deform
temp/partial	temp/profile	temp/ramp	temp/region	temp/sphere	ti
torque/chunk	vacf	vcm/chunk	voronoi/atom		

These are additional compute styles in USER packages, which can be used if LAMMPS is built with the appropriate package.

ackland/atom	basal/atom	cnp/atom	dpd	dpd/atom	fep
force/tally	heat/flux/tally	ke/eff	ke/atom/eff	meso/e/atom	meso/rho/
meso/t/atom	pe/tally	pe/mol/tally	saed	smd/contact/radius	smd/dama
smd/hourglass/error	smd/internal/energy	smd/plastic/strain	smd/plastic/strain/rate	smd/rho	smd/tlsph
smd/tlsph/dt	smd/tlsph/num/neighs	smd/tlsph/shape	smd/tlsph/strain	smd/tlsph/strain/rate	smd/tlsph
smd/triangle/mesh/vertices	smd/ulsph/num/neighs	smd/ulsph/strain	smd/ulsph/strain/rate	smd/ulsph/stress	smd/vol
stress/tally	temp/drude	temp/eff	temp/deform/eff	temp/region/eff	temp/rota

Computes

- Key point:
  - computes store results of their calculation
  - other commands invoke them and use the results
  - e.g. thermo output, dumps, fixes
- Output of computes:
  - global vs per-atom vs local
  - scalar vs vector vs array
  - extensive vs intensive values

Computes

- Key point:
  - computes store results of their calculation
  - other commands invoke them and use the results
  - e.g. thermo output, dumps, fixes
- Output of computes:
  - global vs per-atom vs local
  - scalar vs vector vs array
  - extensive vs intensive values
- Examples:
  - temp & pressure = global scalar or vector
  - pe/atom = potential energy per atom (vector)
  - displace/atom = displacement per atom (array)
  - pair/local & bond/local = per-neighbor or per-bond info
- Many computes are useful with averaging fixes:
  - fix ave/time, ave/chunk (spatial), ave/atom
  - fix ave/histo, ave/correlate

Thermo output

One line of output every N timesteps to screen and log file

- See doc/thermo_style.html

Thermo output

One line of output every N timesteps to screen and log file

- See doc/thermo_style.html
- Any scalar can be output:
  - dozens of keywords: temp, pyy, eangle, lz, cpu
  - any output of a compute or fix: c_ID, f_ID[N], c_ID[N][M]
    - fix ave/time stores time-averaged quantities
  - equal-style variable: v_MyVar
  - one value from atom-style variable: v_xx[N]
  - any property for one atom: q, fx, quat, etc

Thermo output

One line of output every N timesteps to screen and log file

- See doc/thermo_style.html
- Any scalar can be output:
  - dozens of keywords: temp, pyy, eangle, lz, cpu
  - any output of a compute or fix: c_ID, f_ID[N], c_ID[N][M]
    - fix ave/time stores time-averaged quantities
  - equal-style variable: v_MyVar
  - one value from atom-style variable: v_xx[N]
  - any property for one atom: q, fx, quat, etc
- Post-process via:
  - tools/python/logplot.py log.lammps X Y (via GnuPlot)
  - tools/python/log2txt.py log.lammps data.txt X Y ...
  - know how to read thermo output across multiple runs

Dump output

Snapshot of per-atom values every N timesteps

- See doc/dump.html

Dump output

Snapshot of per-atom values every N timesteps

- See doc/dump.html
- Styles
  - atom, custom (both native LAMMPS)
    - VMD will auto-read if file named *.lammpstraj
  - xyz for coords only
  - cfg for AtomEye
  - DCD, XTC for CHARMM, NAMD, GROMACS
    - good for back-and-forth runs and analysis
  - HDF5 and NetCDF and VTK (Paraview)
- Two additional styles
  - local: per-neighbor, per-bond, etc info
  - image: instant picture, rendered in parallel

Dump output

- Any per-atom quantity can be output
  - dozens of keywords: id, type, x, xs, xu, mux, omegax, ...
  - any output of a compute or fix: f_ID, c_ID[M]
  - atom-style variable: v_foo

Dump output

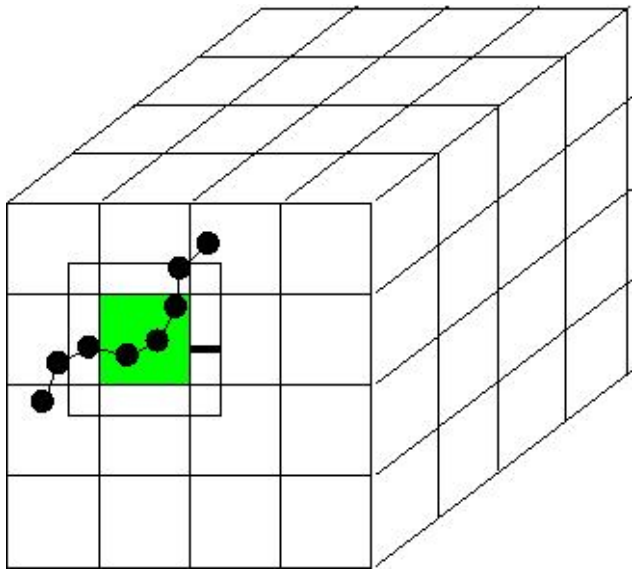
- Any per-atom quantity can be output
  - dozens of keywords: id, type, x, xs, xu, mux, omegax, ...
  - any output of a compute or fix: f_ID, c_ID[M]
  - atom-style variable: v_foo
- Additional options:
  - control which atoms by group or region
  - control which atoms by threshold
    - dump_modify thresh c_pe $>$ 3.0
  - text or binary or gzipped
  - one big file or per snapshot or per proc
  - see dump_modify fileper or nfile
  - MPIIO package for parallel dump output

Dump output

- Any per-atom quantity can be output
  - dozens of keywords: id, type, x, xs, xu, mux, omegax, ...
  - any output of a compute or fix: f_ID, c_ID[M]
  - atom-style variable: v_foo
- Additional options:
  - control which atoms by group or region
  - control which atoms by threshold
    - dump_modify thresh c_pe $>$ 3.0
  - text or binary or gzipped
  - one big file or per snapshot or per proc
  - see dump_modify fileper or nfile
  - MPIIO package for parallel dump output
- Post-run conversion
  - tools/python/dump2cfg.py, dump2pdb.py, dump2xyz.py
  - Pizza.py dump, cfg, ensight, pdb, svg, vtk, xyz

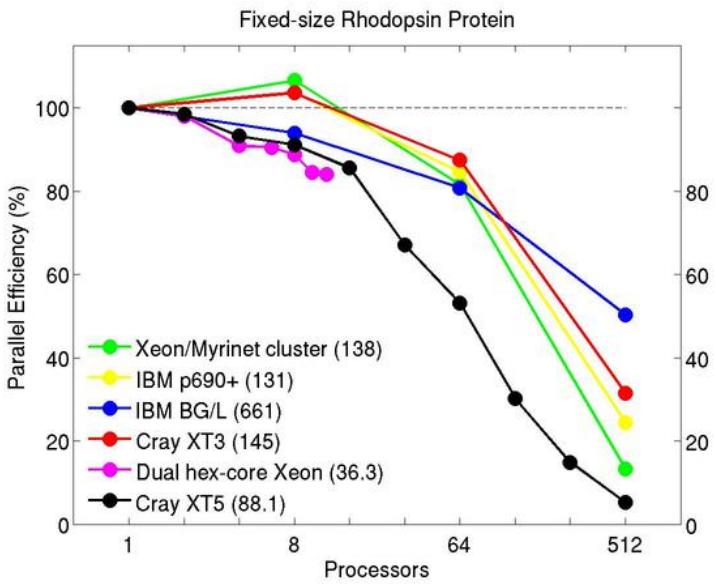
Parallelization in LAMMPS

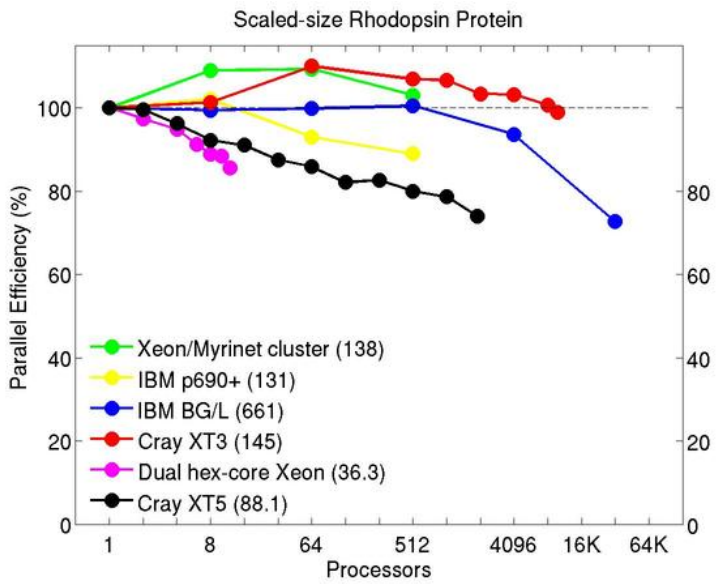
- Physical domain divided into 3d bricks
- One brick per processor
- Atoms carry properties & topology as they migrate
- Comm of ghost atoms within cutoff
  - 6-way local stencil
- Short-range forces $\Rightarrow$ CPU cost scales as $O(N/P)$



Parallel performance

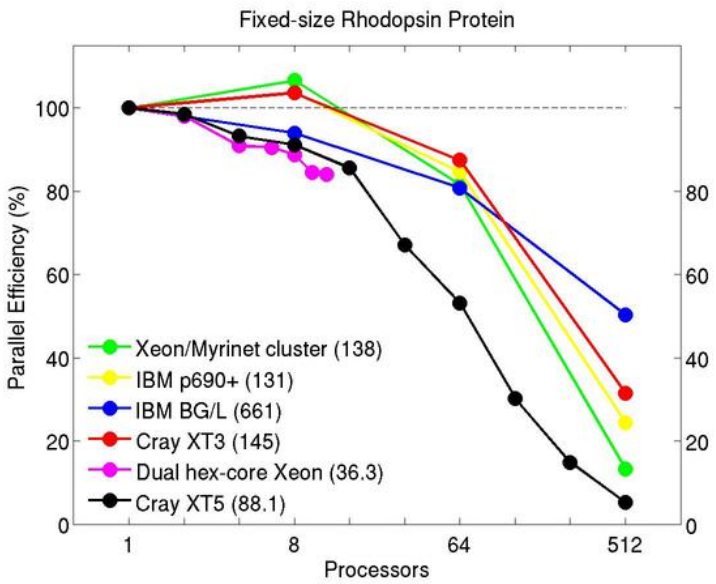
See http://lammps.sandia.gov/bench.html
Strong or weak scaling, O(N/P) until too few atoms/proc

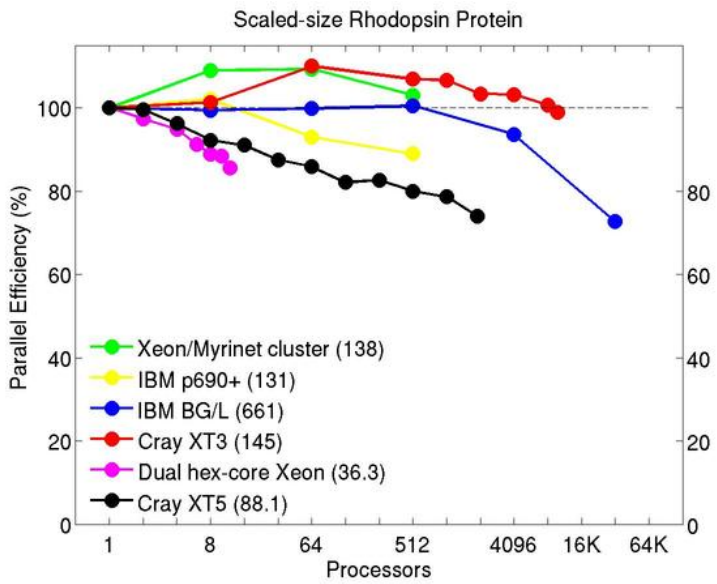




Parallel performance

See http://lammps.sandia.gov/bench.html
Strong or weak scaling, O(N/P) until too few atoms/proc





Exercise: run bench/in.lj, change N and P, is it $O(N/P)$?

- lmp_linux -v x 2 -v y 2 -v z 2 < in.lj
- mpirun -np 2 lmp_linux < in.lj

How to speed-up your simulations

See doc/Section_accelerate.html of manual

1. Many ideas for long-range Coulombics
   - PPPM with 2 vs 4 FFTs (smoothed PPPM)
   - PPPM with staggered grid
   - run_style verlet/split
   - processor layout

How to speed-up your simulations

See doc/Section_accelerate.html of manual

1. Many ideas for long-range Coulombics
   - PPPM with 2 vs 4 FFTs (smoothed PPPM)
   - PPPM with staggered grid
   - run_style verlet/split
   - processor layout
2. OPT, GPU, USER-INTEL, USER-OMP, KOKKOS packages
   - OPT for CPU
   - GPU for NVIDIA GPUs with multiple cores/GPU
   - USER-INTEL for Intel CPU and KNL
   - USER-OMP for OpenMP on multicore CPUs
   - KOKKOS for OpenMP, KNL, GPU
   - Benchmark info at http://lammps.sandia.gov/bench.html
     - new version in next month, focus on accelerator packages
   - Stan Moore, Performance breakout, Wed PM

How to speed-up your simulations

3. Increase time scale via timestep size
   - fix shake for rigid bonds (2 fs)
   - run_style respa for hierarchical timesteps (4 fs)

4. Increase length scale via coarse graining
   - all-atom vs united-atom vs bead-spring
   - mesoscale models:
   - ASPHERE, BODY, COLLOID, FLD packages
   - GRANULAR, PERI, RIGID, SRD packages
   - see doc/Section_packages.html for details
   - Dan Bolintineanu, Coarse-graining breakout, Wed PM

Quick tour of more advanced topics

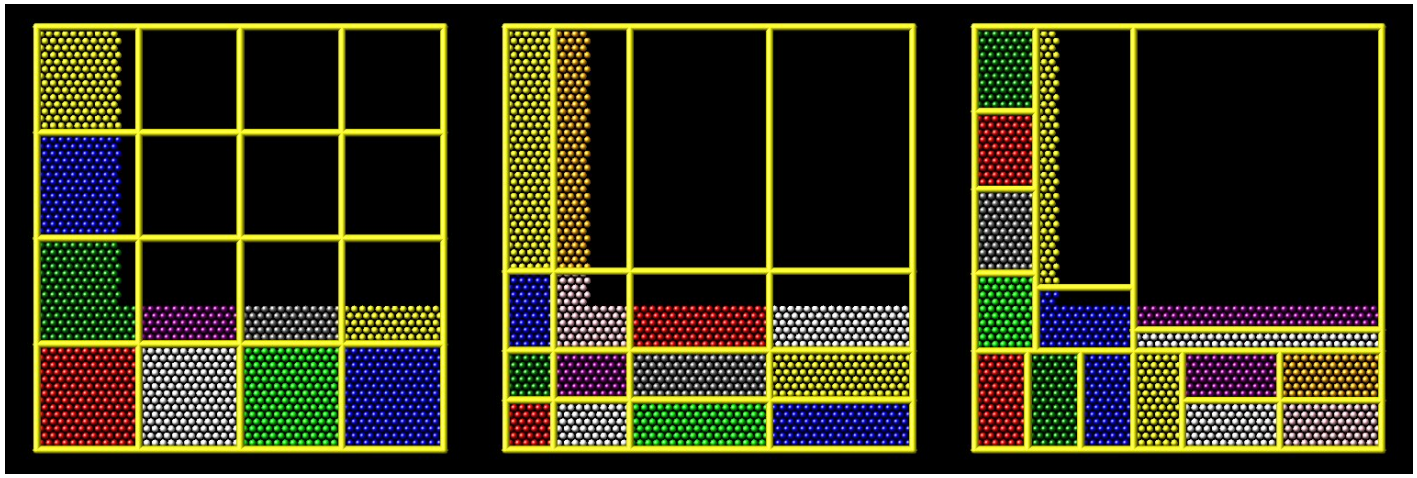
- Units
  - see doc/units.html
  - LJ, real, metal, cgs, si
  - all input/output in one unit system
- Ensembles
  - see doc/Section_howto.html 6.16
  - one or more thermostats (by group)
  - single barostat
  - rigid body dynamics
- Hybrid models
  - pair_style hybrid and hybrid/overlay
  - atom_style hybrid sphere bond ...

Quick tour of more advanced topics

- Aspherical particles
  - see doc/Section_howto.html 6.14
  - ellipsoidal, lines, triangles, rigid bodies
  - ASPHERE package
- Mesoscale and continuum models
  - Corase-grained packages mentioned 2 slides ago
  - PERI package for Peridynamics
  - USER-ATC package for atom-to-continuum (FE)
  - USER-SPH, USER-SMD packages for smoothed particle hydro
  - GRANULAR package for granular media
  - LIGGGHTS simulator for DEM (external code)
    - www.liggghts.com/www.cfdem.com
    - built on top of LAMMPS

Quick tour of more advanced topics

- Multi-replica modeling
  - REPLICA package, see doc/Section_howto.html 6.14
  - parallel tempering, PRD, TAD, NEB
- Load balancing
  - balance command for static LB
  - fix balance command for dynamic LB
  - adjusting proc dividers, or recursive coordinate bisection
  - weighted balancing now an option



Quick tour of more advanced topics

- Energy minimization
- Via usual dynamics
  - pair_style soft
  - fix nve/limit and fix viscous
- Via gradient-based minimization
  - min_style cg, htfn, sd
- Via damped-dynamics minimization
  - min_style quickmin and fire
  - used for nudged-elastic band (NEB)

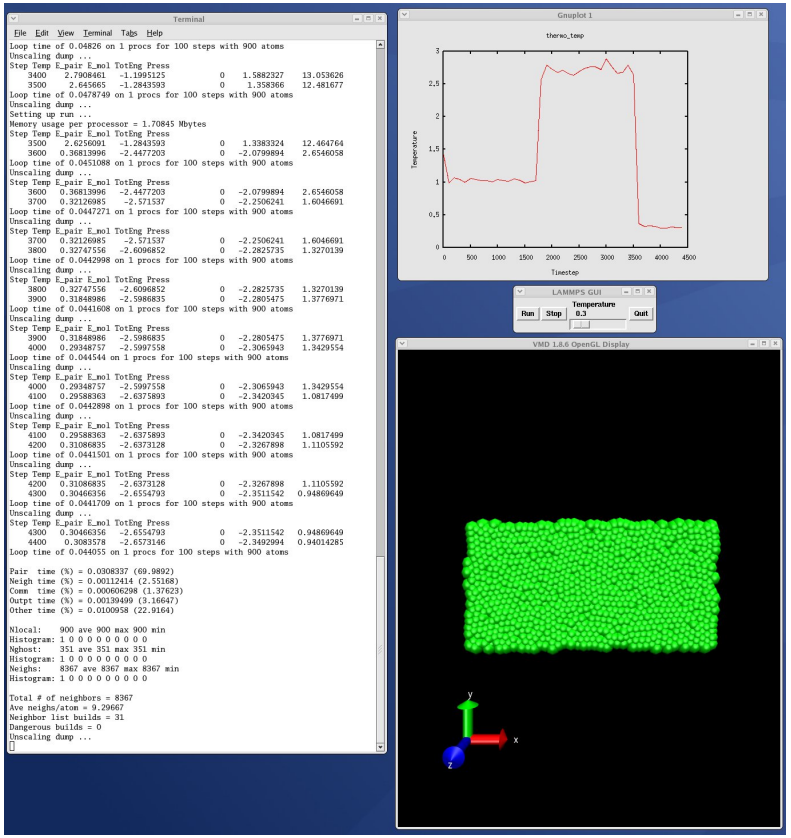
Quick tour of more advanced topics

- Energy minimization
- Via usual dynamics
  - pair_style soft
  - fix nve/limit and fix viscous
- Via gradient-based minimization
  - min_style cg, htfn, sd
- Via damped-dynamics minimization
  - min_style quickmin and fire
  - used for nudged-elastic band (NEB)
- Packages, packages, packages ...
  - package = collection of commands with a theme, or wrapper on an external or provided library
  - ∼60 packages in LAMMPS, wide variety
  - see doc/Section_packages.html for details

Quick tour of more advanced topics

Use LAMMPS as a library

- doc/Section_howto.html 6.10 and 6.19
- C-style interface (C, C++, Fortran, Python)
- examples/COUPLE dir
- python and python/examples directories



Customizing and modifying LAMMPS

- 95% of LAMMPS is customized add-on classes, via styles
- Write a new derived class, drop into src, re-compile

Customizing and modifying LAMMPS

- 95% of LAMMPS is customized add-on classes, via styles
- Write a new derived class, drop into src, re-compile
- Resources:
  - doc/Section_modify.html
  - doc/PDF/Developer.pdf
    - class hierarchy & timestep structure
  - Workshops and Tutorial links on LAMMPS web site:
    - slides for hackers/developers breakout of past workshops
- Come to Developers breakout session, Wed PM
- Please contribute your new code to the LAMMPS distro!
  - doc/Section modify 15:
    Submitting new features for inclusion in LAMMPS
- GitHub site: https://github.com/lammps/lammps

What have people already done with LAMMPS?

- Pictures: http://lammps.sandia.gov/pictures.html
- Movies: http://lammps.sandia.gov/movies.html

	evaporation self-assembly		Compression of nanoparticles
	GCMC model of zeolite occupancy		self-assembling nanofibers from Thiophene-peptide oligomers
	layer-by-layer self assembly		smoothed particle hydrodynamics (SPH) models
	dislocations moving thru grain boundaries		electron force field for non-adiabatic dynamics
	granular particles flowing from hopper		granular Discrete Elememt Method (DEM) models
	fiber dynamics		brazing of two-metal system
	Peridynamics mesoscale modeling of impact fracture		shear faults in a model brittle solid
	stick/slip and polymer flow on rough surfaces		crystallization of polyethylene melt
	melting of polycrystalline metal		deformation and void nucleation under shock loading
	dynamics of an isolated edge dislocation		cavitation in liquid metal
	nanoprecipitates and shock induced plasticity		Brazil nut effect
	ultra-thin Cu nanowire formation		Cu nanowire loading and unloading
	Au nanowire formation and extension		flow of water and ions thru a silica pore
	metal response to He bubble formation		dynamics of rhodopsin protein in lipid membrane
	CO2 escaping from binding pocket of RuBisCO protein		C-terminus of RuBisCO closing over binding pocket
	entropy-driven nano-motor		metal solidification
	liquid crystal conformations		

What have people already done with LAMMPS?

- Pictures: http://lammps.sandia.gov/pictures.html
- Movies: http://lammps.sandia.gov/movies.html

	evaporation self-assembly		Compression of nanoparticles
	GCMC model of zeolite occupancy		self-assembling nanofibers from Thiophene-peptide oligomers
	layer-by-layer self assembly		smoothed particle hydrodynamics (SPH) models
	dislocations moving thru grain boundaries		electron force field for non-adiabatic dynamics
	granular particles flowing from hopper		granular Discrete Elememt Method (DEM) models
	fiber dynamics		brazing of two-metal system
	Peridynamics mesoscale modeling of impact fracture		shear faults in a model brittle solid
	stick/slip and polymer flow on rough surfaces		crystallization of polyethylene melt
	melting of polycrystalline metal		deformation and void nucleation under shock loading
	dynamics of an isolated edge dislocation		cavitation in liquid metal
	nanoprecipitates and shock induced plasticity		Brazil nut effect
	ultra-thin Cu nanowire formation		Cu nanowire loading and unloading
	Au nanowire formation and extension		flow of water and ions thru a silica pore
	metal response to He bubble formation		dynamics of rhodopsin protein in lipid membrane
	CO2 escaping from binding pocket of RuBisCO protein		C-terminus of RuBisCO closing over binding pocket
	entropy-driven nano-motor		metal solidification
	liquid crystal conformations		

- Papers: http://lammps.sandia.gov/papers.html
  - authors, titles, abstracts for 1000s of papers