New distributed grid support in LAMMPS

Steve Plimpton
Sandia National Labs (retired)
Temple University (adjunct)
sjplimp@gmail.com

8th LAMMPS Workshop and Symposium
Virtual meeting – August 2023

Motivation

- LAMMPS is obviously a particle code
- But grids (or meshes) can be useful for:
  - analysis (grouping particles, data reduction)
  - visualization (color each grid cell)
  - hybrid particle/grid models

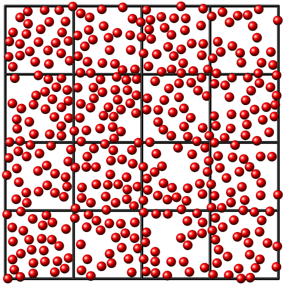
Motivation

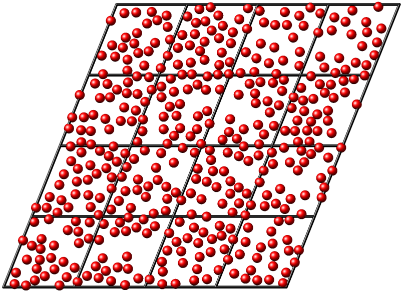
- LAMMPS is obviously a particle code
- But grids (or meshes) can be useful for:
  - analysis (grouping particles, data reduction)
  - visualization (color each grid cell)
  - hybrid particle/grid models
- Examples of hybrid particle/grid models:
  - Long-range Coulombics - FFTs more efficient than huge cutoff
    - charge is mapped to grid cells
    - Poisson's equation solved on grid via FFTs
    - electric field on grid mapped back to particles
  - Two-temperature model
    - atoms + electron temperature, latter on a grid
    - heat diffuses on grid, electron heat couples to atomic motion
  - CG models like material point method (MPM)
    - meshfree continuum-based material model
    - grid used to compute deformation gradient and motion $\vec{a}, \vec{v}$

Grids in LAMMPS

A regular grid overlays entire simulation domain

- 2d or 3d systems
- orthogonal or triclinic, periodic or non-periodic
- any size in each dimension:
    - 4x4 (2d) or 1000x1000x1000 or 100x500x3000
    - 10x10x1 or 1x1x10 or even 1x1x1



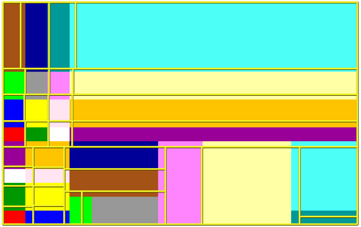


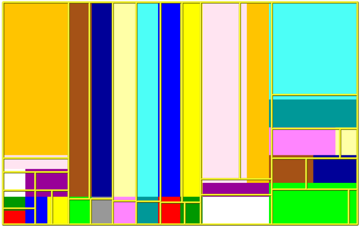
What distributed grid means

- Each proc owns grid cells whose center points
  are inside its sub-domain
- This is always a sub-block of the full grid
- Can also store nearby ghost grid cells its particles interact with
- Works with brick or tiled spatial decompositions

What distributed grid means

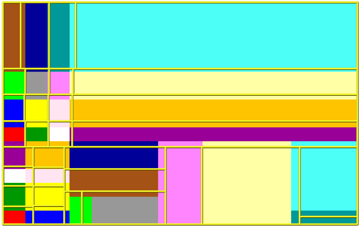
- Each proc owns grid cells whose center points are inside its sub-domain
- This is always a sub-block of the full grid
- Can also store nearby ghost grid cells its particles interact with
- Works with brick or tiled spatial decompositions

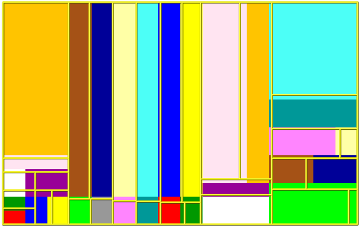




What distributed grid means

- Each proc owns grid cells whose center points are inside its sub-domain
- This is always a sub-block of the full grid
- Can also store nearby ghost grid cells its particles interact with
- Works with brick or tiled spatial decompositions





- Grid cells are typically smaller than proc sub-domains but do not have to be $\Longrightarrow$ a 100 x 100 x 1 grid

Coding details

New Grid2d and Grid3d classes

- can be instantiated by a Pair, Fix, Compute, KSpace style
- stores the partitioning of grid across procs, but NOT data

Coding details

New Grid2d and Grid3d classes

- can be instantiated by a Pair, Fix, Compute, KSpace style
- stores the partitioning of grid across procs, but NOT data

Calling style can:

- define multiple grids (different sizes)
- define/store one or more scalar/vector data sets on each grid
- each grid and data field is named by the caller,
  so that other commands can access the data
- grid data reference: f_ID:gridname:dataname[3]

Coding details

New Grid2d and Grid3d classes

- can be instantiated by a Pair, Fix, Compute, KSpace style
- stores the partitioning of grid across procs, but NOT data

Calling style can:

- define multiple grids (different sizes)
- define/store one or more scalar/vector data sets on each grid
- each grid and data field is named by the caller,
  so that other commands can access the data
- grid data reference: f_ID:gridname:dataname[3]

Support for forward and reverse communication

- forward: comm of owned cell data to ghost cells
- reverse: comm/summation of ghost cell data to owned cells
- caller provides pack & unpack methods for its grid data

Support for load balancing

Code snippets for caller using Grid2d class

1. Define global grid of size Nx by Ny:
   grid = Grid2d(LAMMPS *lmp, MPI_Comm world, Nx, Ny);
2. Handful of methods to choose if/how ghost cells are defined

Code snippets for caller using Grid2d class

1. Define global grid of size Nx by Ny:
   grid = Grid2d(LAMMPS *lmp, MPI_Comm world, Nx, Ny);
2. Handful of methods to choose if/how ghost cells are defined
3. Partition the grid - return extents:
   grid->setup_grid(ixlo,ixhi,iylo,iyhi,oxlo,oxhi,oylo,oyhi);

Code snippets for caller using Grid2d class

1. Define global grid of size Nx by Ny:
   ```
   grid = Grid2d(LAMMPS *lmp, MPI_Comm world, Nx, Ny);
   ```
2. Handful of methods to choose if/how ghost cells are defined
3. Partition the grid - return extents:
   ```
   grid->setup_grid(ixlo,ixhi,iylo,iyhi,oxlo,oxhi,oylo,oyhi);
   ```
4. Setup and perform forward, reverse communication:
   ```
   grid->setup_comm(nbuf1,nbuf2);
   grid->forward_comm(nper,nbyte,buf1,buf2,MPI_DOUBLE);
   grid->reverse_comm(nper,nbyte,buf1,buf2,MPI_DOUBLE);
   ```
   caller provides pack/unpack callback methods

Code snippets for caller using Grid2d class

1. Define global grid of size Nx by Ny:
   grid = Grid2d(LAMMPS *lmp, MPI_Comm world, Nx, Ny);
2. Handful of methods to choose if/how ghost cells are defined
3. Partition the grid - return extents:
   grid->setup_grid(ixlo,ixhi,iylo,iyhi,oxlo,oxhi,oylo,oyhi);
4. Setup and perform forward, reverse communication:
   grid->setup_comm(nbuf1,nbuf2);
   grid->forward_comm(nper,nbyte,buf1,buf2,MPI_DOUBLE);
   grid->reverse_comm(nper,nbyte,buf1,buf2,MPI_DOUBLE);
   caller provides pack/unpack callback methods
5. Read_file() and write_file() methods
   for reading/writing grid data from/to files
6. Remap() methods to invoke when load balancing occurs

Code snippets for caller using Grid2d class

1. Define global grid of size Nx by Ny:
   grid = Grid2d(LAMMPS *lmp, MPI_Comm world, Nx, Ny);
2. Handful of methods to choose if/how ghost cells are defined
3. Partition the grid - return extents:
   grid->setup_grid(ixlo,ixhi,iylo,iyhi,oxlo,oxhi,oylo,oyhi);
4. Setup and perform forward, reverse communication:
   grid->setup_comm(nbuf1,nbuf2);
   grid->forward_comm(nper,nbyte,buf1,buf2,MPI_DOUBLE);
   grid->reverse_comm(nper,nbyte,buf1,buf2,MPI_DOUBLE);
   caller provides pack/unpack callback methods
5. Read_file() and write_file() methods
   for reading/writing grid data from/to files
6. Remap() methods to invoke when load balancing occurs
7. Caller provides grid data access methods for other classes

Current use of distributed grids in LAMMPS

- KSpace solvers:
  - PPPM: gathering charge, FFTs, scattering forces
  - MSM: multilevel cascade of grid resolutions
- Pair styles:
  - AMOEBA and HIPPO force fields
  - multiple terms with FFTs (similar to PPPM)

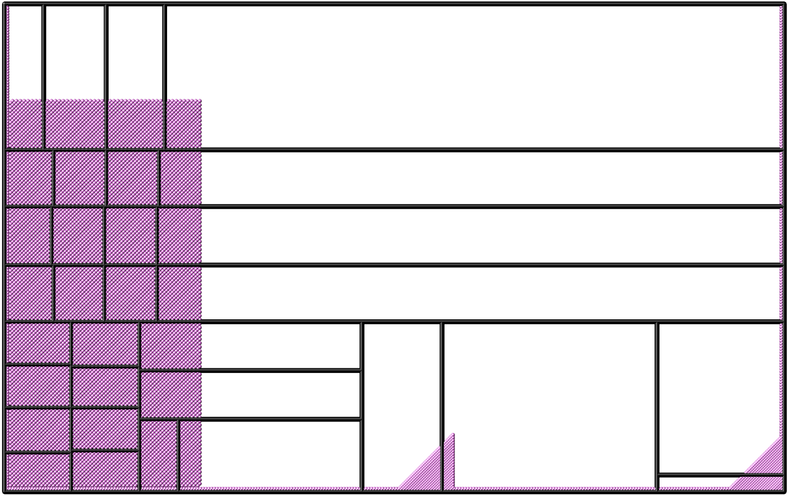
Current use of distributed grids in LAMMPS

- KSpace solvers:
  - PPPM: gathering charge, FFTs, scattering forces
  - MSM: multilevel cascade of grid resolutions
- Pair styles:
  - AMOEBA and HIPPO force fields
  - multiple terms with FFTs (similar to PPPM)
- Fix styles:
  - fix ttm/grid = two-temperature model (fix ttm is global grid)
  - fix ave/grid for particles or grid cells
    - fix ave/chunk allows chunks which are regular grid cells
    - but it's a global grid, not distributed
    - thus inefficient in CPU and memory for large grids
    - fix ave/grid can use arbitrarily large grids
- Compute styles: compute property/grid

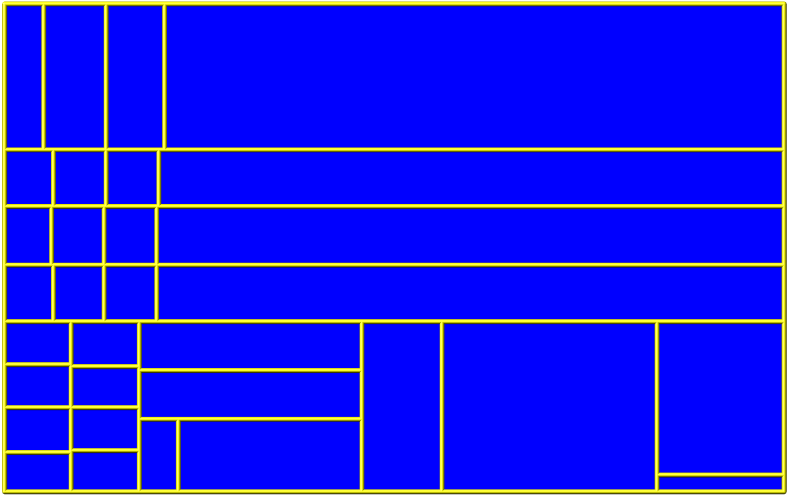
Current use of distributed grids in LAMMPS

- KSpace solvers:
  - PPPM: gathering charge, FFTs, scattering forces
  - MSM: multilevel cascade of grid resolutions
- Pair styles:
  - AMOEBA and HIPPO force fields
  - multiple terms with FFTs (similar to PPPM)
- Fix styles:
  - fix ttm/grid = two-temperature model (fix ttm is global grid)
  - fix ave/grid for particles or grid cells
    - fix ave/chunk allows chunks which are regular grid cells
    - but it's a global grid, not distributed
    - thus inefficient in CPU and memory for large grids
    - fix ave/grid can use arbitrarily large grids
- Compute styles: compute property/grid
- Dump styles:
  - dump grid, dump image, dump movie
  - OVITO can read/viz LAMMPS dump grid files

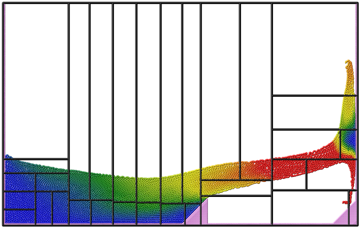
SPH movie of water flow - particles colored by KE

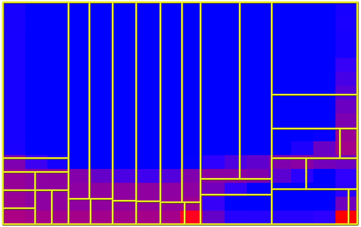


SPH movie of water flow - grid colored by particle count



Last snapshot comparison





More details

User guide: https://docs.lammps.org/Howto_grid.html

- Overview from user perspective
- Current commands that use distributed grids
- How to access grid data in input script commands

Programmer Guide: https://docs.lammps.org/Developer_grid.html

- How to write a new style which uses a distributed grid
- Description of all methods in Grid2d/Grid3d classes