GPUs for Molecular Dynamics (MD) Simulations: a user's perspective (morning lab)

Eduardo M. Bringa (ebringa@yahoo.com) & Emmanuel Millán

CONICET / Instituto de Ciencias Básicas, Universidad Nacional de Cuyo, Mendoza

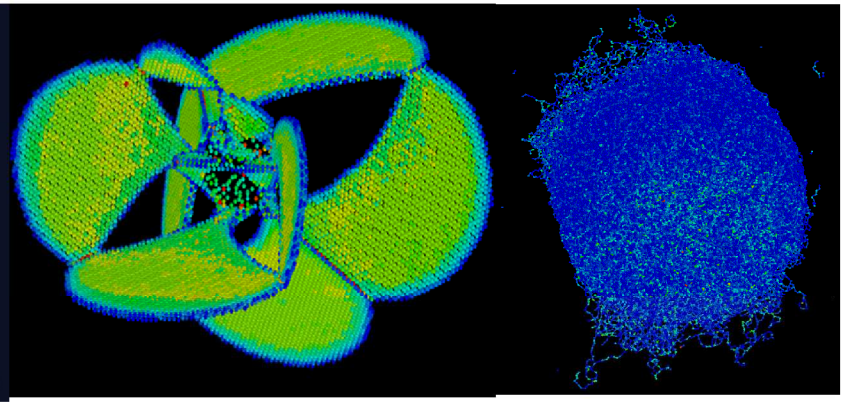
WHPC

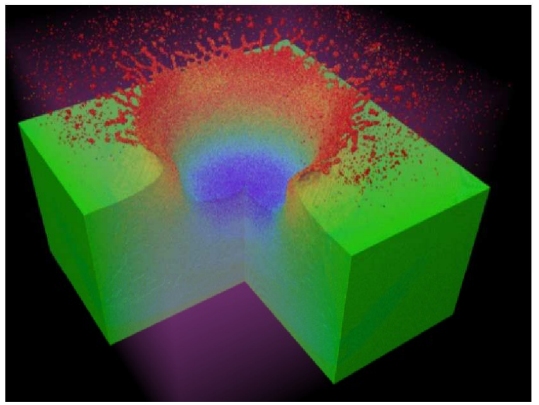
UNC, May 2013

Collaborators:

D. Tramontina, C. Ruestes, F. Fioretti, C. Garcia Garino (UN Cuyo), L. Forconesi (ITU), R. Isoardi (FUESMEN), S. Manzi, E. Perino, M.F. Piccoli, M. Printista (UNSL), D. Schwen (LANL), A. Higginbotham (Oxford)

SiMAF
Simulaciones en Materiales Astrofísica y Física





Una herramienta muy útil para estudiar materiales: Dinámica Molecular clásica =Molecular Dynamics=MD

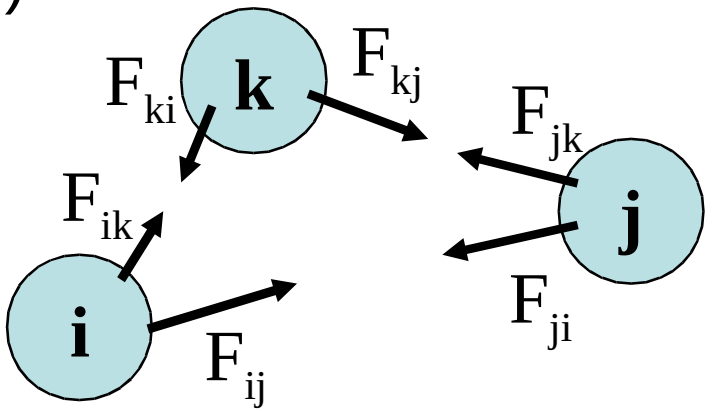
- N partículas clásicas. Partícula *i* con posición $\boldsymbol{r}_i$, tiene velocidad $\mathbf{v}_i$ y aceleración $\boldsymbol{a}_i$.

- Partículas interactúan a través de un potencial empírico, $V(\boldsymbol{r}_1,.., \boldsymbol{r}_i,.., \boldsymbol{r}_N)$, que generalmente incluye interacciones de muchos cuerpos.

- Partículas obedecen las ecuaciones de movimiento de Newton. Partícula *i*, masa m_i:

$$\boldsymbol{F}_i = -\nabla_i V(\boldsymbol{r}_1,.., \boldsymbol{r}_i,.., \boldsymbol{r}_N) = m_i \boldsymbol{a}_i = m_i (d^2\boldsymbol{r}_i / dt^2)$$

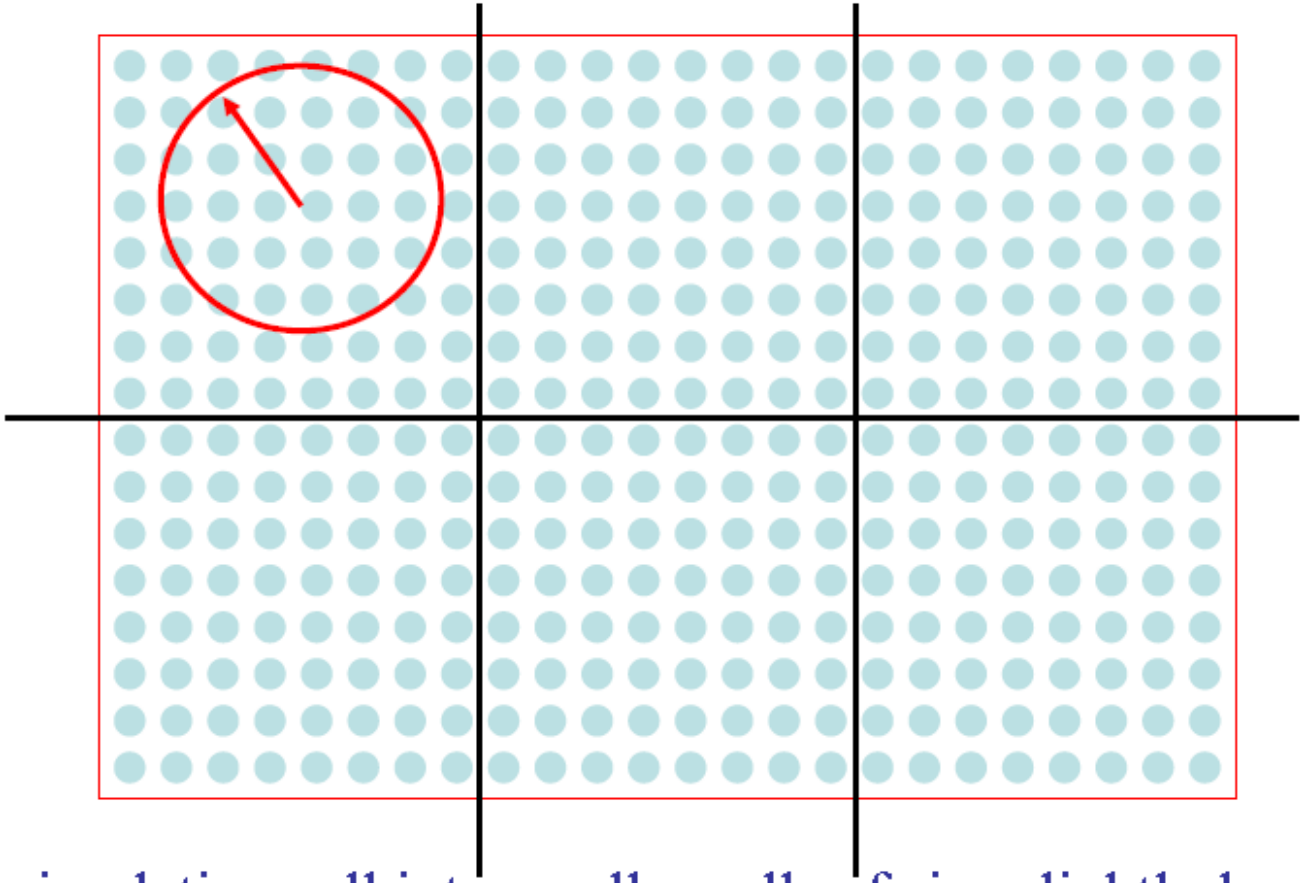
- Volumen<0.5 μm^3~10^9 átomos)
- Tiempos t<1 ns, Δt~1 fs)
- Varios integradores disponibles
- Pueden incorporarse efectos electrónicos (Koci *et al*, PRB 2006).



How do we simulate a large number of atoms?

- Integrating the two body problem is one thing …. But integrating the motion of N particles, with N=(several million-billions) is a whole different ball game.
- **Short-range potentials** (not 1/r): use an appropriate cut-off and do spatial decomposition of the domain. This will ensure nearly perfect parallel scaling [O(N)]. Sometimes a VERY long cut-off is used for (1/r) potentials, with varying results.
- **Long-range potentials** (1/r): old method uses *Ewald* summation. New methods (PME,PPPM=P3M, etc.) are typically O(NlogN). Even newer methods (variations of multipole expansion) can be O(N), at the price of a large computational overhead. This is the same as the problem of **N-body simulations** used in astrophysics.
- Have to be careful with **boundary conditions** (free, periodic, expanding, damping, etc.) and check for system size effects.

Spatial decomposition & cell lists



- Divide simulation cell into smaller cells of size slightly larger than r_c
- Atoms only interact with other atoms in the same of nearest neighbor cell

- Allocating atoms to cells can be done in order N
- Calculating interactions is also order N

Alejandro Strachan, http://nanohub.org/resources/5838#series

How to run LAMMPS?
Add package command to regular LAMMPS input

Package cuda (version added to stable LAMMPS version on 11/03/2013, for package USER-CUDA)
Syntax:
package cuda options
•extension, command extension of the accelerator package to be used (i.e. cuda)
•options = optional arguments which are interpreted by the accelerator package
Examples:
package cuda
package cuda gpu/node 4 pinned 0 dotiming
Description:
This command is intended as an easy to use interface to accelerator packages such as USER-CUDA which provide alternate versions of existing LAMMPS classes. If specified LAMMPS will try to replace the style-names of fixes, pair forces and atom styles with "stylename/extension". When this extended style-name does not exist the original version is used. As a consequence one can use the same input scripts for the accelerated and the standard version of LAMMPS, since without the accelerator package being installed the command is effectively ignored.
Supported packages:
Every accelerator package can scan the list of options for arguments it might interpret. Here the valid options for each package are listed:
USER-CUDA
•**gpu/node *m*** : specify the number of GPUs per node to be used. The default value is *m*=2. GPUs are sorted according to their number of multi processors before assignment. This option is ignored if all of the GPUs are in compute mode *exclusive* or *forbidden*.
•**gpu/node special *m g1 ... gm*** : specify the IDs of GPUs to be used on each node. *m* is the number of GPUs to be used per node, and *g1* to *gm* are their IDs. Attention: GPU IDs in CUDA environments are currently different from IDs reported by i.e. nvidia-smi. The deviceQuery tool of the NVIDIA SDK reports the numbers which are seen by USER-CUDA.
•**override/bpa** values = flag
flag = 0 for TpA algorithm, 1 for BpA algorithm
•**timing** : output some more detailed timing information at the end of the run.
•**test *m*** : output information (positions etc.) of particle *m* several times per timestep. This is only usefull for debug purposes.
Restrictions:
This command must be used before any of the classes that should be replaced are set. This typically means that the command should be used before the system or the atom style is defined.
Only one accelerator package can be used at a time.

USER-CUDA package compilation

USER-CUDA package

Provides (more details about the features of LAMMPS CUDA):

- 26 pair forces
- long range coulomb with pppm/cuda
- nve/cuda, nvt/cuda, npt/cuda, nve/sphere/cuda
- several more important fixes and computes

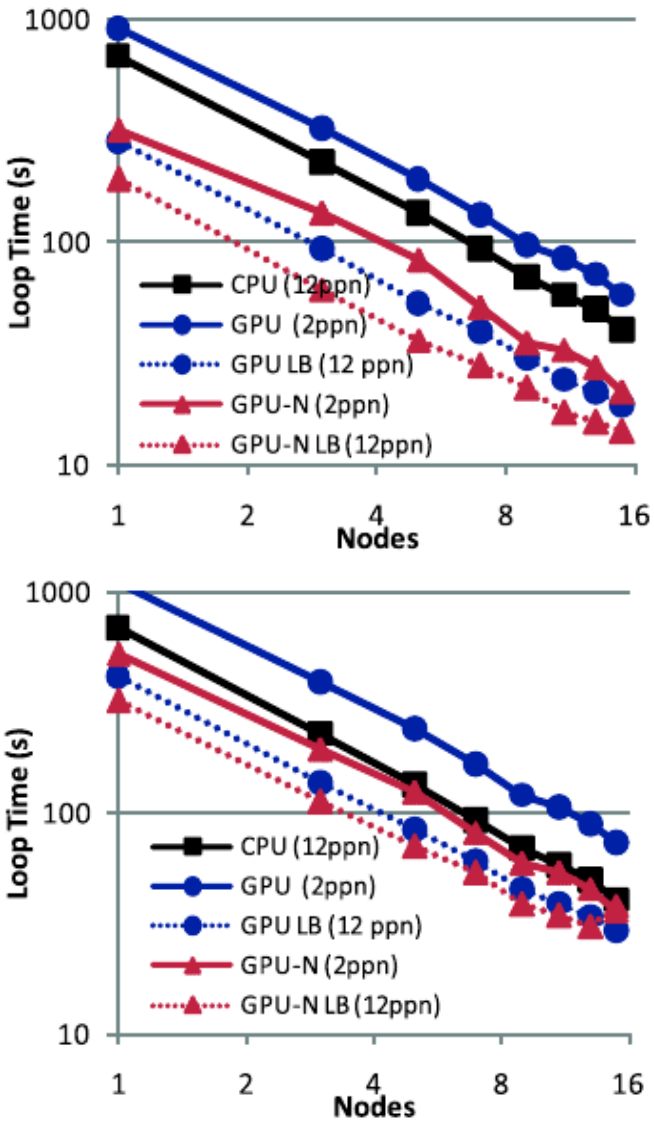
Installation (more details about the installaton of LAMMPS CUDA)

- Make sure you can compile LAMMPS without packages with 'make YOUR-Machinefile'
- # cd lib/cuda
- # emacs Makefile.defaults (precision, debug, compute capability)
- # make
- # cd ../../src
- Install the standard packages with 'make yes-standard'
- Install USER-CUDA with 'make yes-USER-CUDA'
- # make YOUR-MACHINE

LAMMPS (lammps.sandia.gov)GPU vs CPU performance

Strong-scaling for the Lennard-Jones test case with and without acceleration.
2 GPUs Tesla C2050/node.

- **Top: Comparison of loop time without acceleration (CPU), acceleration with 1 process per GPU (2 ppn), and load balancing (LB) with 6 processes per GPU (12 ppn) for single precision. Neighbor calculations are performed on the GPU for the GPU-N cases.**
- **Bottom: double precision.**
- **Loop times are the wall time required to complete the entire simulation loop.**

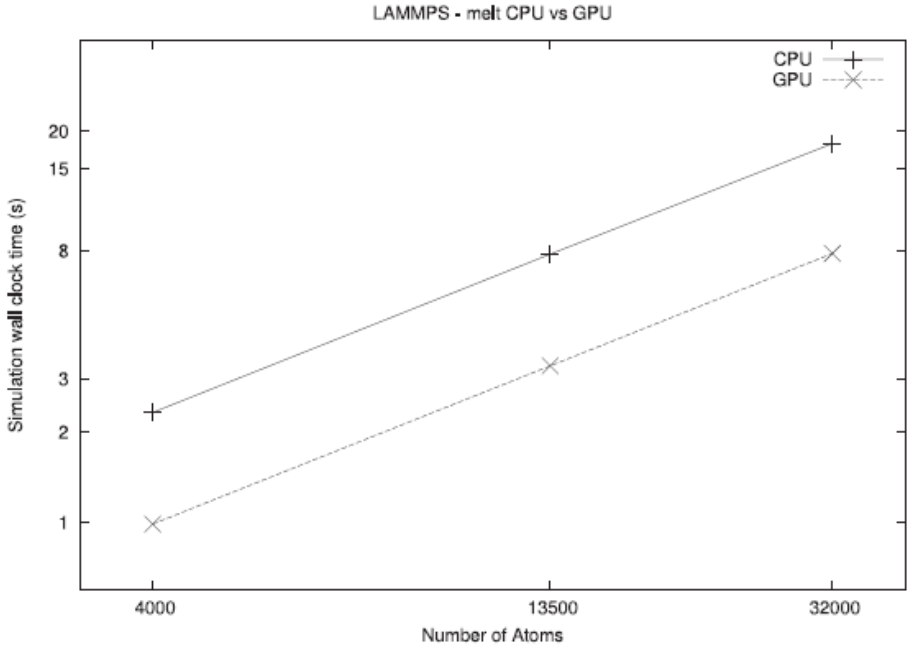


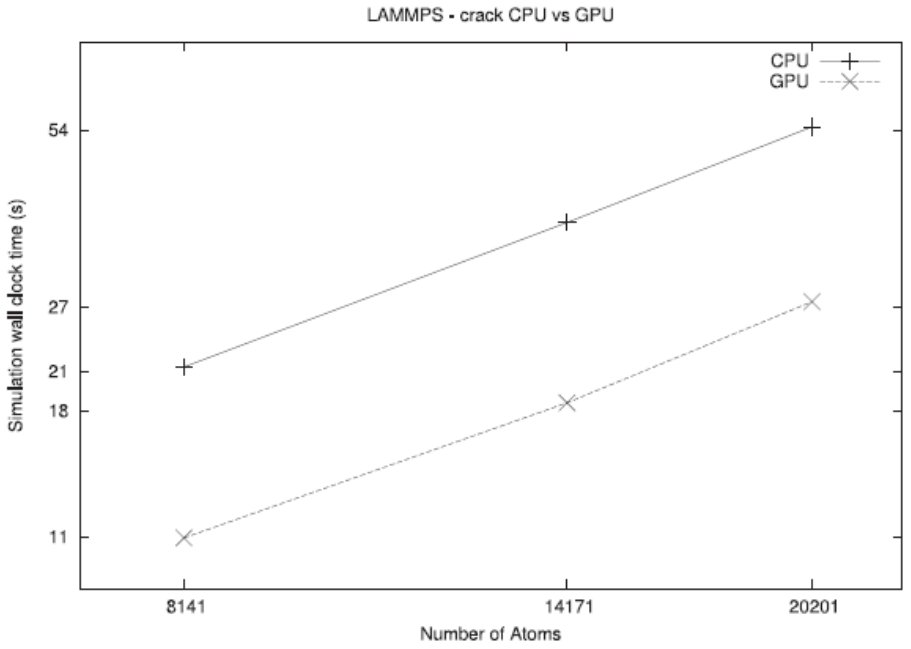
Implementing molecular dynamics on hybrid high performance computers – short range forces. W. Michael Brown, Peng Wang, Steven J. Plimpton, Arnold N. Tharrington. Computer Physics Communications 182 (2011) 898–911.

Scaling with “old” GPU

“GP-GPU Processing of Molecular Dynamics Simulations”. E. Millán Kujtiuk, E.M. Bringa, A. Higginbotham, C. García Garino. Proceedings of HPC 2010. High-Performance Computing Symposium. pp. 3234-3248.

•**GPU:** NVIDIA GeForce 9500 GT 1GB DDR-2 with 32 processing units (cost ~US$ 100 dollars), running with CUDA 3.0.
•**Single CPU:** AMD Athlon 64 X2 Dual Core Processor 4600+ 2.4 GHz each core, Slackware Linux 13.0.
• **Storm:** master node + 12 slave stations. Intel P4 processor at 3.0 Ghz, 1 GiB of DDR-1 RAM, Gigabit Ethernet Card and Rocks Linux 5.3.
•**Twister:** 16 nodes with a Intel Core 2 Duo at 3.0 Ghz processor, 4GiB of DDR-2 RAM, Gigabit Ethernet card and Rocks Linux 5.0.

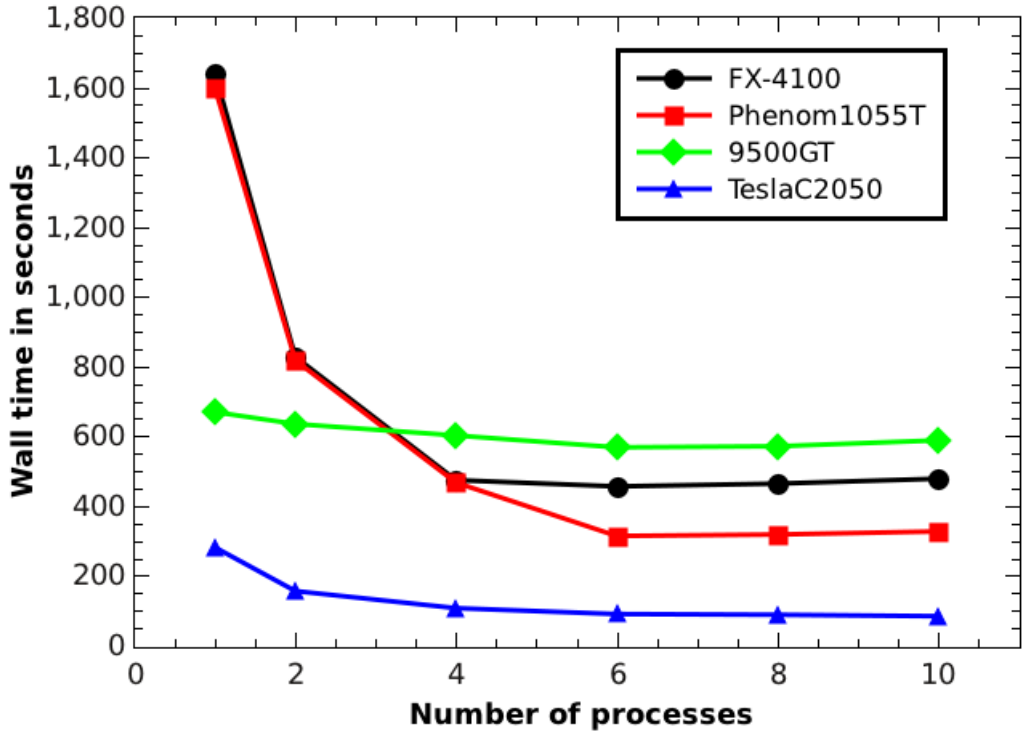




Melt example from LAMMPS, for two GPUs (9500GT & Tesla C2050)

Wallclock time in seconds for 100 simulations varying initial velocity distribution, for 32000 atoms and 500 steps. Single precision for GPU (GPU 9500GT does not support double precision) and double precision for CPU. The difference in final system temperature for single precision was 1.6426 ∓ 0.0004, for double precision in the Tesla GPU the final system temperature was 1.6425 ∓ 0.0004.

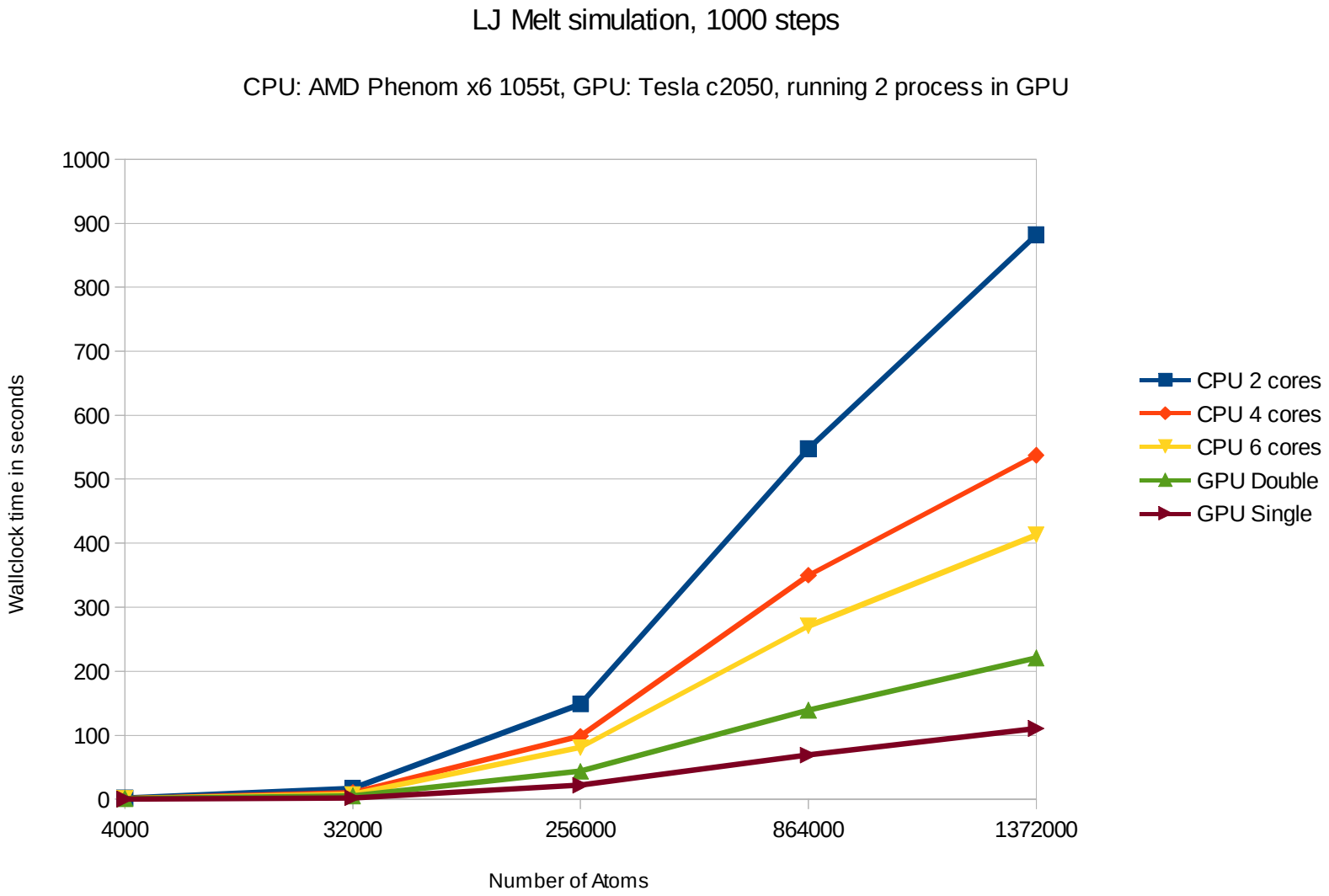
Hardware configuration:
Workstation 1: CPU AMD FX-4100 with GPU NVIDIA 9500GT.
Workstation 2: CPU AMD Phenom x6 1055t with GPU NVIDIA Tesla c2050



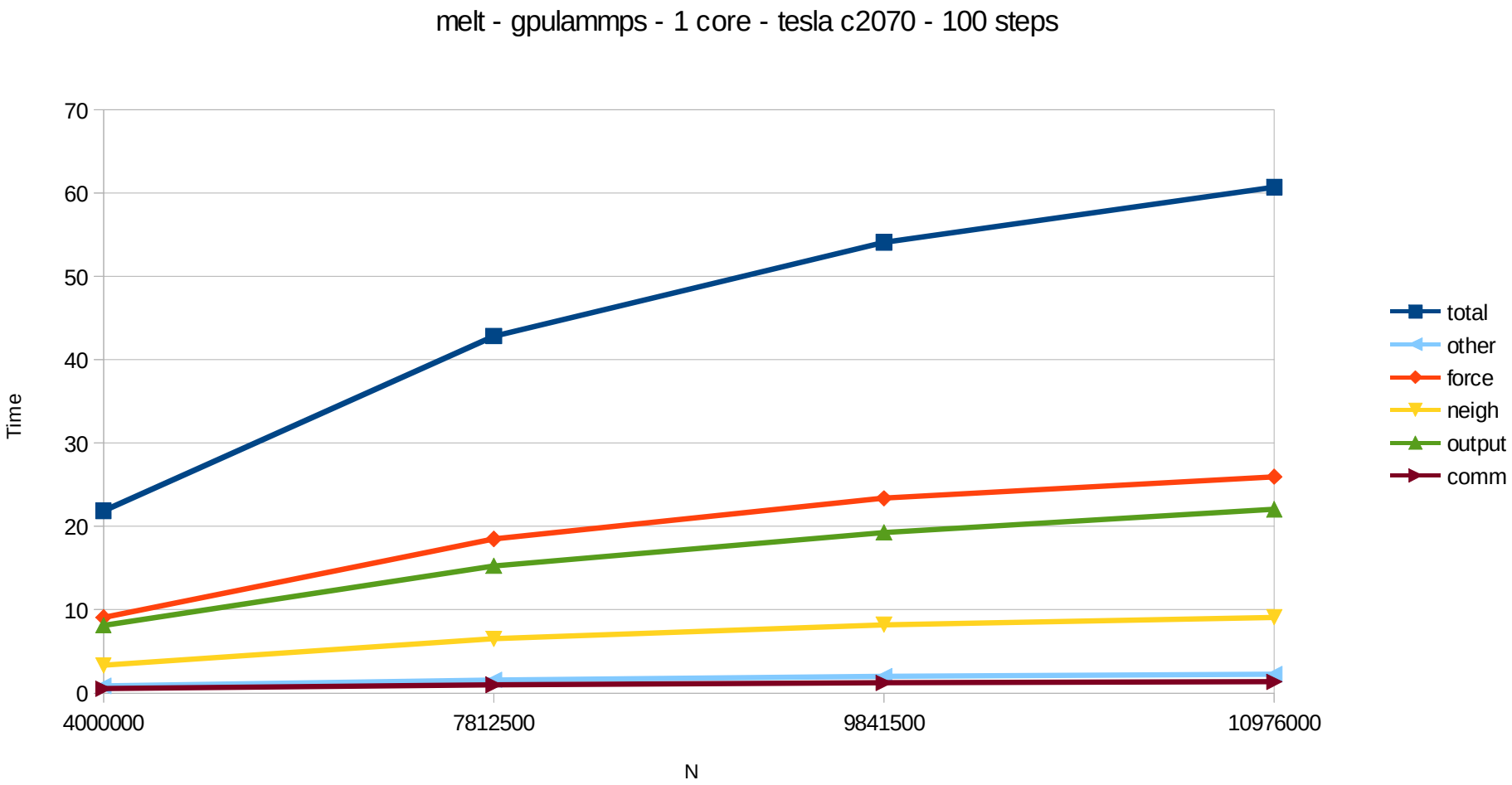
Emmanuel Nicolás Millán, Carlos García Garino y Eduardo M. Bringa. Parallel execution of a parameter sweep for molecular dynamics simulations in a hybrid GPU/CPU environment. XVIII Congreso Argentino de Ciencias de la Computación 2012 (CACIC), Bahia Blanca, Buenos Aires. ISBN-978-987-1648-34-4. 2012. http://sedici.unlp.edu.ar/handle/10915/23634

Scaling in CPU/GPU

GPU: NVIDIA Tesla c2050, 3 GB RAM, 448 processing units
CPU: AMD Phenom x6 1055t 2.8GHz. 12 GB of RAM.



Time spent in different portions of the code



Can the timing for each of the different portions be reduced?
How would you do it?

GPUs for Molecular Dynamics (MD) Simulations: a user's perspective (afternoon lecture)

Eduardo M. Bringa (ebringa@yahoo.com) & Emmanuel Millán

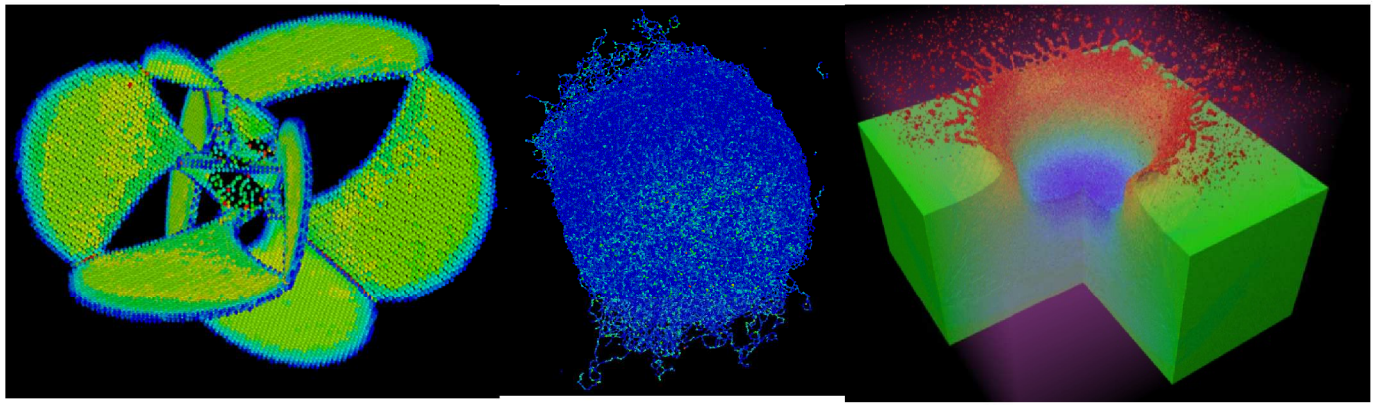
CONICET / Instituto de Ciencias Básicas, Universidad Nacional de Cuyo, Mendoza

WHPC

UNC, May 2013

Collaborators:

D. Tramontina, C. Ruestes, F. Fioretti, C. Garcia Garino (UN Cuyo), L. Forconesi (ITU), R. Isoardi (FUESMEN), S. Manzi, E. Perino, M.F. Piccoli, M. Printista (UNSL), D. Schwen (LANL), A. Higginbotham (Oxford)



Outline

- **Introduction**
- **LAMMPS**
- **GPU & USER-CUDA packages**
- **Performance:**

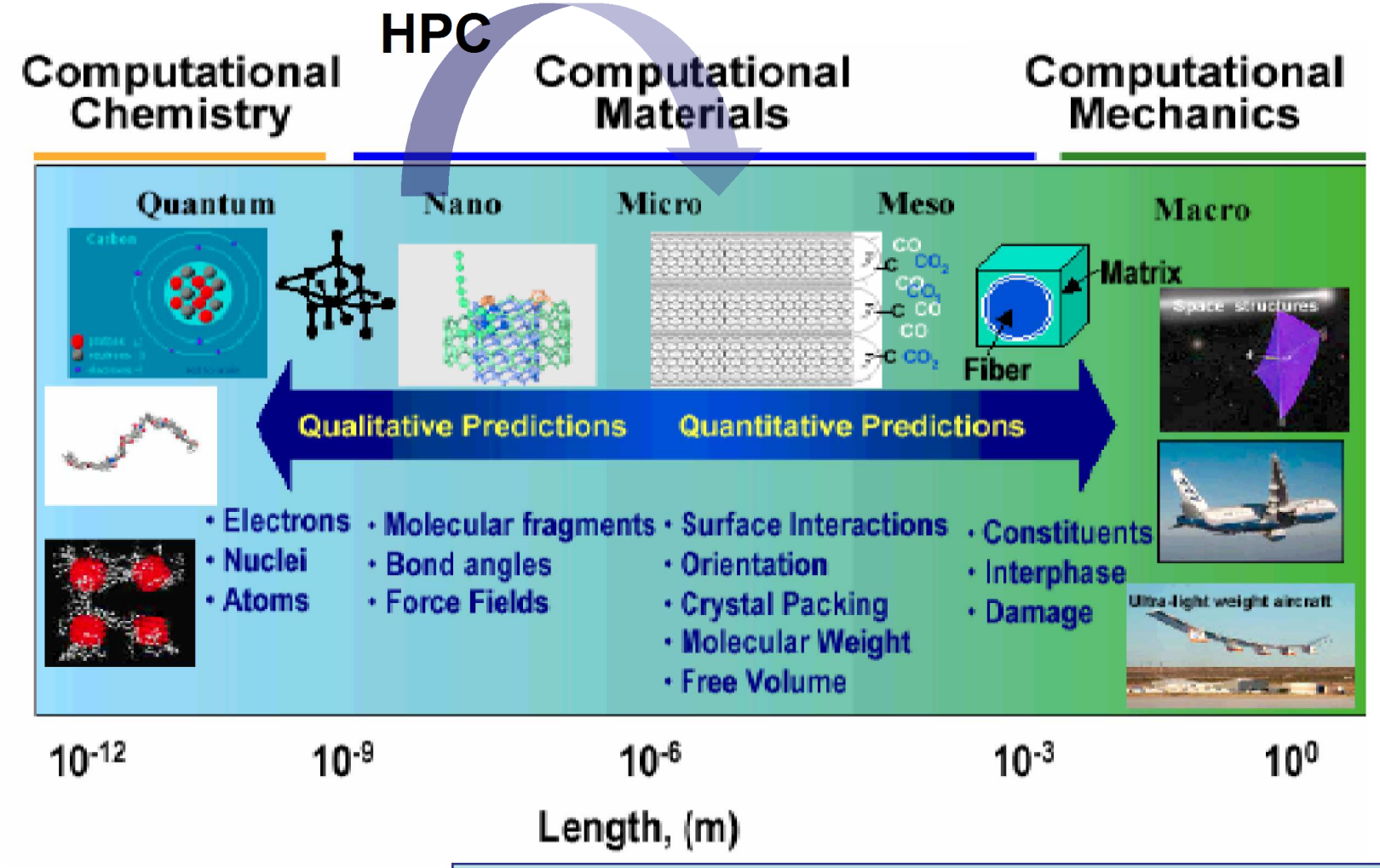
a)single/mixed/double precision

b)CPU/GPU neighbor lists

c)Load balancing: static & dynamic

- **MD Examples**
- **Other GPU projects**

Simulations at different scales: HPC allows going from nano to micro



By Greg Odesgard, NASA Langley Research Center

University of Virginia, MSE 492/627: Introduction to Atomistic Simulations, Leonid Zhigilei, http://www.people.virginia.edu/~lz2n/mse627

Una herramienta muy útil para estudiar materiales: Dinámica Molecular clásica =Molecular Dynamics=MD

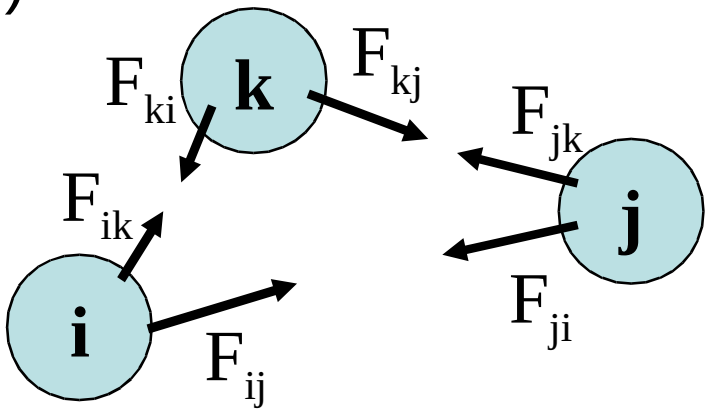
- N partículas clásicas. Partícula *i* con posición $\boldsymbol{r}_i$, tiene velocidad $\mathbf{v}_i$ y aceleración $\boldsymbol{a}_i$.

- Partículas interactúan a través de un potencial empírico, $V(\boldsymbol{r}_1,.., \boldsymbol{r}_i,.., \boldsymbol{r}_N)$, que generalmente incluye interacciones de muchos cuerpos.

- Partículas obedecen las ecuaciones de movimiento de Newton. Partícula *i*, masa m_i:

$$\boldsymbol{F}_i = -\nabla_i V(\boldsymbol{r}_1,.., \boldsymbol{r}_i,.., \boldsymbol{r}_N) = m_i \boldsymbol{a}_i = m_i (d^2\boldsymbol{r}_i / dt^2)$$

- Volumen<0.5 μm^3~10^9 átomos)
- Tiempos t<1 ns, Δt~1 fs)
- Varios integradores disponibles
- Pueden incorporarse efectos electrónicos (Koci *et al*, PRB 2006).



Interatomic potential (Phys/Eng) or Force Field (Chem/Bio)

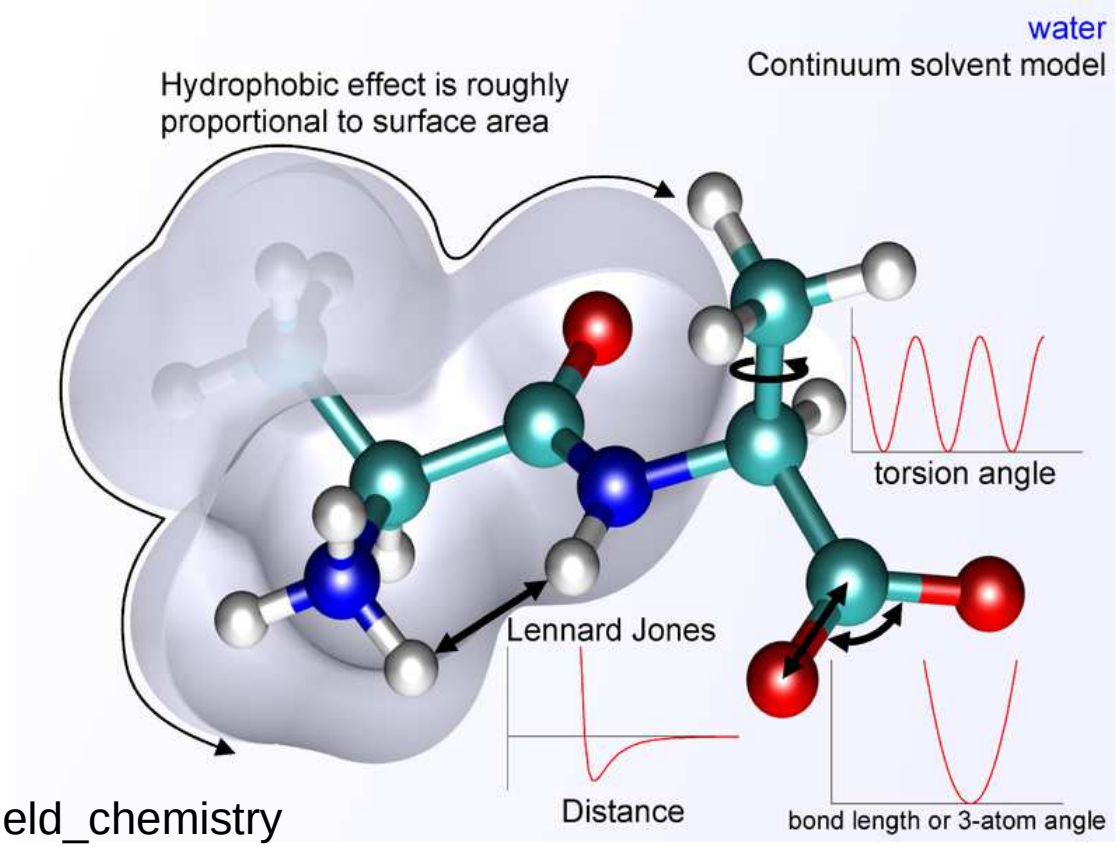
Example

$$E_{total} = E_{bonded} + E_{nonbonded}$$

$$E_{bonded} = E_{bond} + E_{angle} + E_{dihedral}$$

$$E_{nonbonded} = E_{electrostatic} + E_{vanderWaals}$$

Golden rule: "garbage in, garbage out"

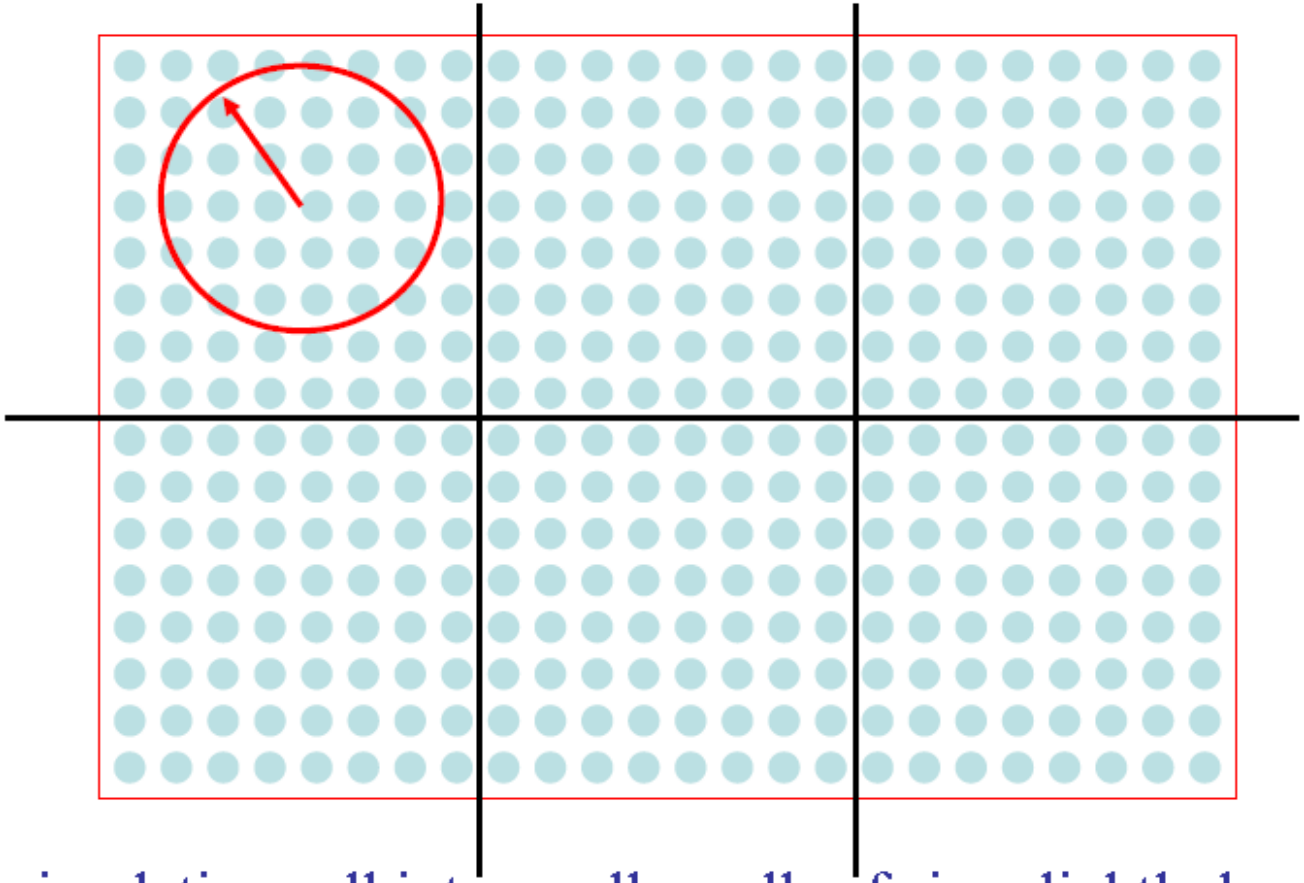


http://en.wikipedia.org/wiki/Force_field_chemistry

How do we simulate a large number of atoms?

- Integrating the two body problem is one thing …. But integrating the motion of N particles, with N=(several million-billions) is a whole different ball game.
- **Short-range potentials** (not 1/r): use an appropriate cut-off and do spatial decomposition of the domain. This will ensure nearly perfect parallel scaling [O(N)]. Sometimes a VERY long cut-off is used for (1/r) potentials, with varying results.
- **Long-range potentials** (1/r): old method uses *Ewald* summation. New methods (PME,PPPM=P3M, etc.) are typically O(NlogN). Even newer methods (variations of multipole expansion) can be O(N), at the price of a large computational overhead. This is the same as the problem of **N-body simulations** used in astrophysics.
- Have to be careful with **boundary conditions** (free, periodic, expanding, damping, etc.) and check for system size effects.

Spatial decomposition & cell lists



- Divide simulation cell into smaller cells of size slightly larger than r_c
- Atoms only interact with other atoms in the same of nearest neighbor cell

- Allocating atoms to cells can be done in order N
- Calculating interactions is also order N

Alejandro Strachan, http://nanohub.org/resources/5838#series

Atomistic simulations are extremely helpful but … still have multiple limitations

With MD you can obtain….

"Real" time evolution of your system.

Thermodynamic properties, including T(r,t) temperature profiles that can be used in rate equations.

Mechanical properties, including elastic and plastic behavior.

Surface/bulk/cluster growth and modification.

X-ray and "IR" spectra

Etcetera …

Limitations of MD

- Can simulate only small samples ($L<1$ μm, up to ~10^9 atoms).
- Can simulate only short times ($t<1$ μs, because $\Delta t \sim 1$ fs).
- Computationally expensive (weeks).
- **Potential's golden rule: trash in → trash out.**
- Interaction potentials for alloys, molecular solids, and excited species not well know.
- Despite its limitations **MD** is a very **powerful tool** to study nanosystems.

Summary: there are many opportunities for MD

- **Petascale➔Exascale! (USA & EU initiatives).** Science **335**, 394 (2012).
- **New software: novel algorithms and preferably open source** (Nature 482, 485 (2012)**]. Still need significant advances in visualization (**Visual Strategies: A Practical Guide to Graphics for Scientists and Engineers, F. Frankel & A. Depace, Yale University Press, 2012**), dataset analysis [**Science **334**, 1518 (2011)**], self-recovery & fault tolerance, etc.**
- **New hardware :** better (faster/greener/cheaper) processors, connectivity, memory and disk access; MD-tailored machines (MD-GRAPE-4, Anton, etc.); GPUs, MICs, hybrid architectures (GPU/CPU); cloud computing, etc.
- **Experiments going micro-nano/ns-ps ➔ same as MD**
- **Can go micron-size, but still have to connect to mm-m scale ➔** novel approaches needed, including smart sampling, concurrent coupling, dynamic/ adaptive **load balancing/refining** for heterogeneous systems, asynchronous simulations, etc.
- **No news beyond ns (1e8 atoms), microseconds (1e4 atoms), s (1e2 atoms) :(**
- **Need human resources with mix of hardware, soft & science expertise.**

Many MD codes can now use GPU acceleration
Often used as black-boxes without understanding limitations

LAMMPS (*Large-scale Atomic/Molecular Massively Parallel Simulator*):
http://lammps.sandia.gov/ . MPI ofr several GPUs/cores (LJ: 1.2 ~10^7 atoms max Tesla C2070)
GPULAMMPS: http://code.google.com/p/gpulammps/ CUDA + OpenCL

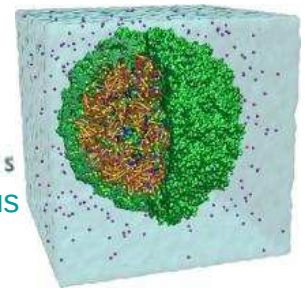
HOOMD-Blue (*Highly Optimized Object-oriented Many-particle Dynamics***)**:
http://codeblue.umich.edu/hoomd-blue/index.html OMP for several GPUs in single board.

DL_POLY:
http://www.cse.scitech.ac.uk/ccg/software/DL_POLY/ F90+MPI, CUDA+OpenMP port.

GROMACS : http://www.gromacs.org/Downloads/Installation_Instructions/Gromacs_on_GPUs
Uses OpenMM libs (https://simtk.org/home/openmm). No paralelization. ~10^6 atoms max.

AMBER (*Assisted Model Building with Energy Refinement*): http://ambermd.org/gpus/
Ross Walker (keynote). MPI for several GPUs/cores. TIP3P, PME, ~10^6 atoms max Tesla C2070)

NAMD (*"Not another" MD*): http://www.ks.uiuc.edu/Research/namd/
GPU/CPU clusters.
VMD (Visual MD): http://www.ks.uiuc.edu/Research/vmd/



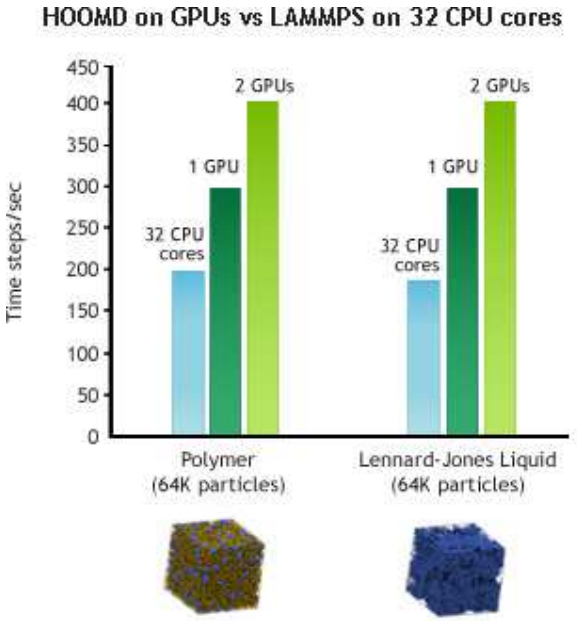
1,000,000+ atom Satellite Tobacco Mosaic Virus
Freddolino *et al*., *Structure*, 14:437-449, 2006.

Many more!!!!

GTC 2010 Archive: videos and pdf's: http://www.nvidia.com/object/gtc2010-presentation-archive.html#md

Highly Optimized Object-oriented Many-particle Dynamics -HOOMD-Blue

http://codeblue.umich.edu/hoomd-blue/index.html



(64K particles)

- LAMMS on 32 x86 CPU cores
- HOOMD-BLUE on 1 Tesla 10-series GPU
- HOOMD-BLUE on 2 Tesla 10-series GPUs

http://www.nvidia.com/object/molecular_dynamics.html

- *General purpose molecular dynamics simulations fully implemented on graphics processing units* Joshua A. Anderson, Chris D. Lorenz, and Alex Travesset, Journal of Computational Physics **227** (2008) 5342-5359.
- *Molecular Dynamics on Graphic Processing Units: HOOMD to the Rescue*. Joshua A. Anderson and Alex Travesset, Computing in Science & Engineering 10(6) (2008).

HOOMD-Blue 0.11.2

Host	GPU (ECC off)	Polymer TPS	LJ liquid TPS
Xeon E5-2670 (SandyBridge-E)	Tesla K20X	885	1014
Xeon E5-2667 (SandyBridge-E)	Tesla K20m	785	890
Xeon E5-2670 (SandyBridge-E)	Tesla M2090	597	682
Xeon X5670 (Westmere)	Tesla M2070	486	553
Xeon X5670 (Westmere)	Tesla M2050	490	560

Highly Optimized Object-oriented Many-particle Dynamics -HOOMD-Blue

http://codeblue.umich.edu/hoomd-blue/doc/page_quick_start.html

Python script for LJ run: test.hoomd

$$V(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$

```
from hoomd_script import *
# create 100 random particles of name A
init.create_random(N=100, phi_p=0.01,
name='A')
# specify Lennard-Jones interactions
    # between particle pairs
lj = pair.lj(r_cut=3.0)
lj.pair_coeff.set('A', 'A', epsilon=1.0,
sigma=1.0)
# integrate at constant temperature
all = group.all();
integrate.mode_standard(dt=0.005)
integrate.nvt(group=all, T=1.2, tau=0.5)
# run 10,000 time steps
run(10e3)
```

Run

```
$ hoomd test.hoomd
```

Output:

```
HOOMD-blue 0.9.0
Compiled: Wed Oct 28 06:58:46 EDT 2009
Copyright 2008, 2009 Ames Laboratory Iowa State University and
the Regents of the University of Michigan -----
http://codeblue.umich.edu/hoomd-blue/
 This code is the implementation of the algorithms discussed in:
Joshua A. Anderson, Chris D. Lorenz, and Alex Travesset - 'General
Purpose Molecular Dynamics Fully Implemented on Graphics
Processing Units', Journal of Computational Physics 227 (2008)
5342-5359 -----
test.hoomd:004 | init.create_random(N=100, phi_p=0.01, name='A')
test.hoomd:007 | lj = pair.lj(r_cut=3.0)
test.hoomd:008 | lj.pair_coeff.set('A', 'A', epsilon=1.0, sigma=1.0)
test.hoomd:011 | all = group.all(); Group "all" created containing
100 particles test.hoomd:012 | integrate.mode_standard(dt=0.005)
test.hoomd:013 | integrate.nvt(group=all, T=1.2, tau=0.5)
test.hoomd:016 | run(10e3)
starting run ** Time 00:00:00 | Step 10000 / 10000 | TPS 35417.9 |
ETA 00:00:00 Average TPS: 35405 --------- --
Neighborlist stats:
370 normal updates / 100 forced updates / 0 dangerous updates
n_neigh_min: 0 / n_neigh_max: 10 / n_neigh_avg: 2.41
bins_min: 0 / bins_max: 6 / bins_avg: 1.5625
run complete **
```

Outline

- Introduction
- LAMMPS
- GPU & USER-CUDA packages
- Performance:

a)single/mixed/double precision

b)CPU/GPU neighbor lists

c)Load balancing: static & dynamic

- MD Examples.
- Other GPU projects

LAMMPS (http://lammps.sandia.gov/)

Some of my personal reasons to use LAMMPS:

1) Free, open source (GNU license).

2) Easy to learn and use:

(a) mailing list in sourceforge.

(b) responsive developers and user community.

(c) extensive docs:http://lammps.sandia.gov/doc/Section_commands.html#3_5

3) It runs efficiently in my laptop (2 cores) and in BlueGeneL (100 K cores), including parallel I/O, with the same input script. Also efficient for GPUs.

4) Very efficient parallel energy minimization, including cg & FIRE.

5) Includes many-body, bond order, & reactive potentials. Can simulate inorganic & bio systems, granular and CG systems.

6) Can do extras like DSMC, TAD, NEB, TTM, semi-classical methods, etc.

7) Extensive set of analysis routines: coordination, centro, cna, etc.

8) Easy to write analysis inside input, using something similar to pseudo-code.

LAMMPS (http://lammps.sandia.gov/) on GPUs

Two possibilities: Package GPU (former GPU-LAMMPS) or Package USER-CUDA

http://lammps.sandia.gov/doc/package.html (about gpu, cuda and omp packages)

•Need **CUDA GPU driver** and **CUDA toolkit**, but CUDA SDK is not needed.

•**Sample compilation (**compile gpu library, add files to main src dir, then compile whole code**)**:

cd ~/lammps/lib/gpu

emacs Makefile.linux (compute capability 1.3/2.0, single/mix/double precision)

make -f Makefile.linux (to obtain libgpu.a)

cd ../../src

emacs ./MAKE/Makefile.linux (here need to change options and paths)

make yes-asphere (for granular materials)

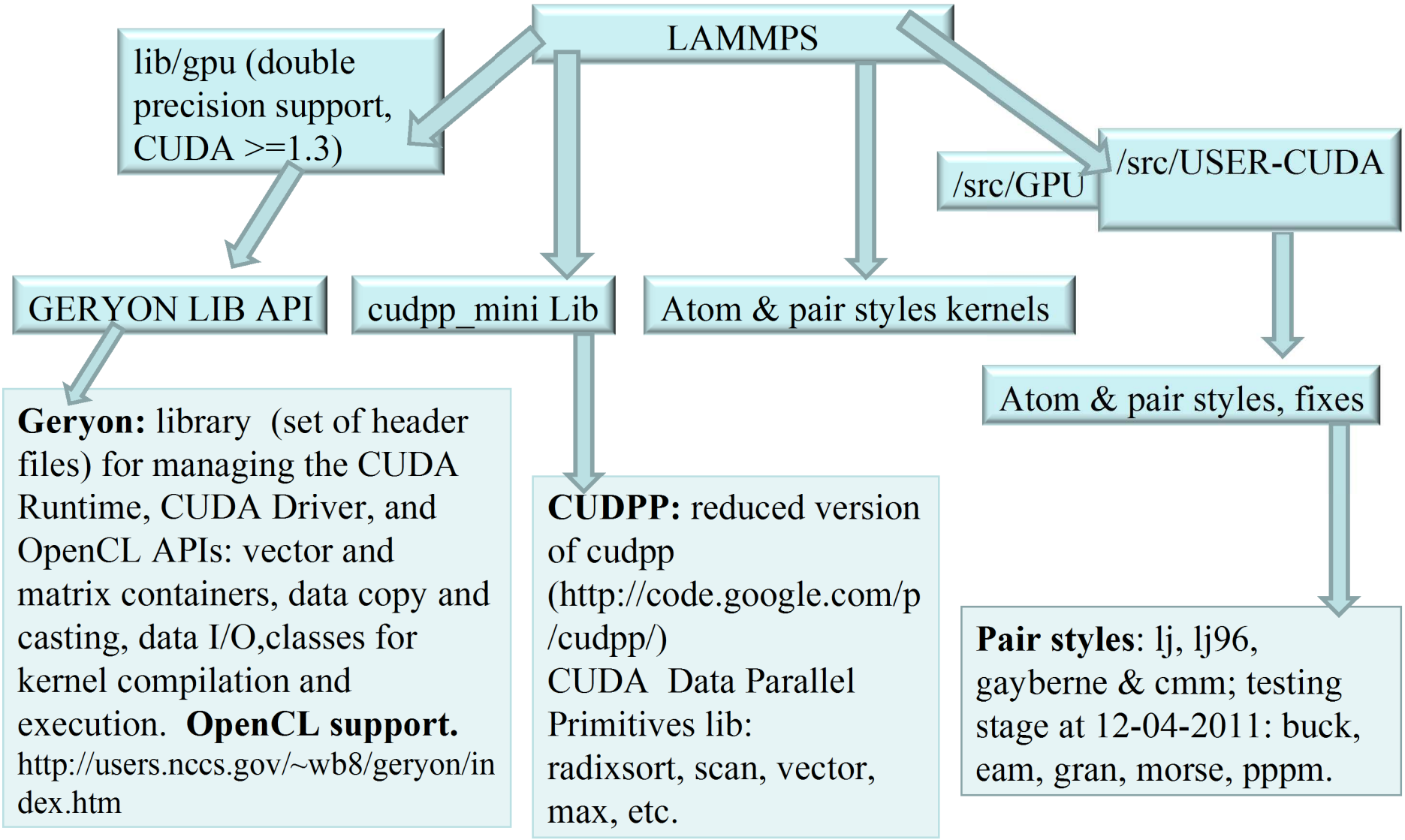
Make yes-manybody (for EAM in Package GPU)

make yes-kspace (for electrostatics, needs fftw.2.x)

make yes-gpu (to copy /src/gpu files to /src)

make linux (to obtain executable)

Schematic code structure



GPU Package

Novel, most up to date GPU code. GNU GPL v2 license.

- **Main developers:** Paul Crozier (Sandia), Mike Brown, Arnold Tharrington, Scott Hampton (Oak Ridge), Axel Kohlmeyer (Temple), Christian Trott, Lars Winterfeld (Ilmenau, Germany), Duncan Poole, Peng Wang (Nvidia), etc.
- Immense number of features (potentials, fixes, etc.) available
- Supports OpenCL
- Alternative: Package USER-CUDA.

Features in USER-CUDA (more extensive set than in GPU package)

Run styles:

verlet/cuda

Forces:

born/coul/long/cuda, buck/coul/cut/cuda,buck/coul/long/cuda, buck/cuda, cg/cmm/coul/cut/cuda, cg/cmm/coul/debye/cuda,, cg/cmm/coul/long/cuda, cg/cmm/cuda, eam/cuda, eam/alloy/cuda, eam/fs/cuda, gran/hooke/cuda, lj/charmm/coul/charmm/implicit/cuda, lj/charmm/coul/charmm/cuda, lj/charmm/coul/long/cuda, lj/cut/coul/cut/cuda, lj/cut/coul/debye/cuda, lj/cut/coul/long/cuda, lj/cut/cuda, lj/expand/cuda, lj/gromacs/coul/gromacs/cuda, lj/gromacs/cuda, lj/smooth/cuda, lj96/cut/cuda, morse/cuda, morse/coul/long/cuda, pppm/cuda

Fixes:

npt/cuda, nve/cuda, nvt/cuda, nve/sphere/cuda, enforce2d/cuda, temp/berendsen/cuda, temp/rescale/cuda, addforce/cuda , setforce/cuda, aveforce/cuda, shake/cuda, gravity/cuda, freeze/cuda.

Computes:

temp/cuda, temp/partial/cuda, pressure/cuda, pe/cuda

Atom styles:

atomic/cuda, charge/cuda, full/cuda, granular/cuda

Comparison of GPU and USER-CUDA packages

http://lammps.sandia.gov/doc/Section_accelerate.html#acc_8

GPU pack:assign multiple CPUs (cores) to single GPU ["hybrid" nodes with multicore CPU(s) and GPU(s)].
USER-CUDA: you can only use one CPU per GPU.

GPU pack: moves per-atom data (coordinates, forces) back-and-forth between CPU and GPU every timestep.
USER-CUDA: only does this on timesteps when a CPU calculation is required (e.g. to invoke a fix or compute that is non-GPU-ized) → data transfer cost of USER-CUDA can be low → faster than GPU.

GPU pack often faster than USER-CUDA pack, if atoms/GPU is "small". Crossover depends strongly on pair style. LJ (single precision) ~ 50K-100K atoms/GPU. Double precision → crossover can be smaller.

Both packs compute bonded interactions (bonds, angles, etc) on the CPU → model with bonds will force the USER-CUDA package to transfer per-atom data back-and-forth between the CPU and GPU every timestep. If GPU is running with several MPI processes assigned to one GPU, cost of computing the bonded interactions is spread across more CPUs and hence the GPU package can run faster.

GPU pack. with multiple CPUs assigned to one GPU → performance depends on bandwidth between the CPUs and GPU → **performance affected if full 16 PCIe lanes are not available for each GPU**. In HPC environments two devices might share one PCIe 2.0 16x slot. Also **many multi-GPU mainboards do not provide full 16 lanes to each of the PCIe 2.0 16x slots.**

Differences between the two packages

http://lammps.sandia.gov/doc/Section_accelerate.html#acc_8

GPU package accelerates only pair force, neighbor list, and PPPM calculations.

USER-CUDA package currently supports a wider range of pair styles and can also accelerate many fix styles and some compute styles, as well as neighbor list and PPPM calculations.

The USER-CUDA package does not support acceleration for minimization.

The USER-CUDA package does not support hybrid pair styles.

The USER-CUDA package can order atoms in the neighbor list differently from run to run resulting in a different order for force accumulation.

The USER-CUDA package has a limit on the number of atom types that can be used in a simulation.

The GPU package requires neighbor lists to be built on the CPU when using exclusion lists or a triclinic simulation box.

The GPU package uses more GPU memory than the USER-CUDA package. This is generally not a problem since typical runs are computation-limited rather than memory-limited.

Outline

- Introduction
- LAMMPS
- GPU & USER-CUDA packages

- Performance:

a)single/mixed/double precision

b)CPU/GPU neighbor lists

c)Load balancing: static & dynamic

- MD Examples.
- Other GPU projects

Review on LAMMPS GPU implementation

Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington. Comp. Phys. Comm. **182** (2011) 898–911

- **Discussion of several important issues in porting a large molecular dynamics code for use on parallel hybrid machines.**
- **Objectives:**

  a) Minimize the amount of code that must be ported for efficient acceleration.

  b) Utilize the available processing power from both multi-core CPUs and accelerators.
- **Presents results on a parallel test cluster containing 32 Fermi GPUs and 180 CPU cores.**

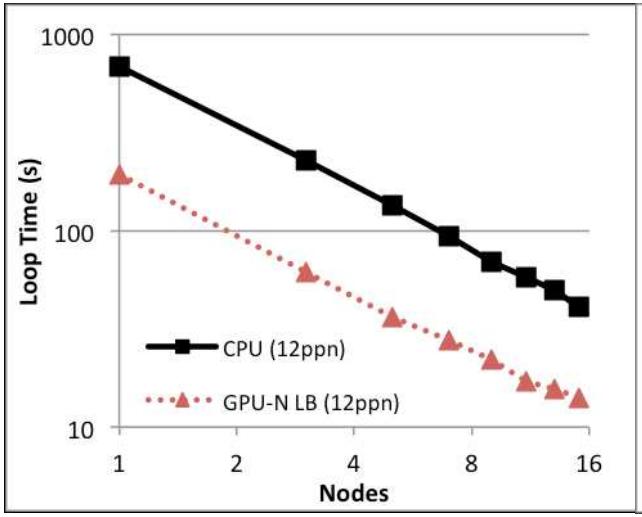
Parallel and CPU/GPU Decomposition

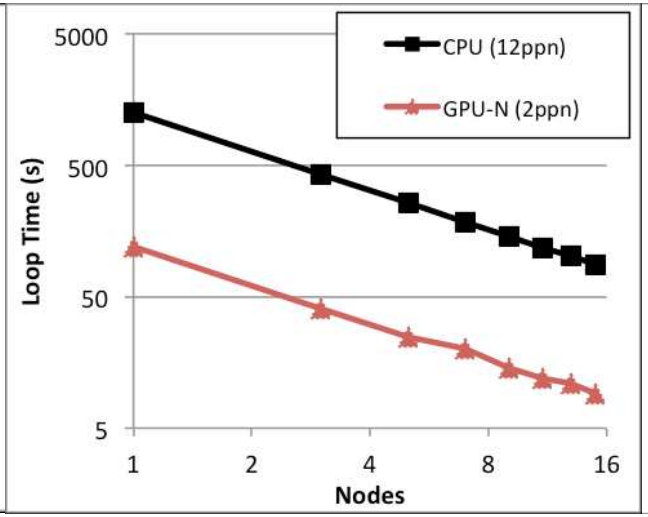
- **Multiple MPI processes (CPU cores) can share single accelerator (GPU) .**
- **User can choose fixed load balance between CPU & GPU for the calculation of short range forces.**
- **Dynamic load balancing can also be chosen.**
- **GPU force calculation.**
- **Neighbor list can be carried out in GPU or CPU.**
- **Time integration represents only small computational cost and it is carried out in CPU.**

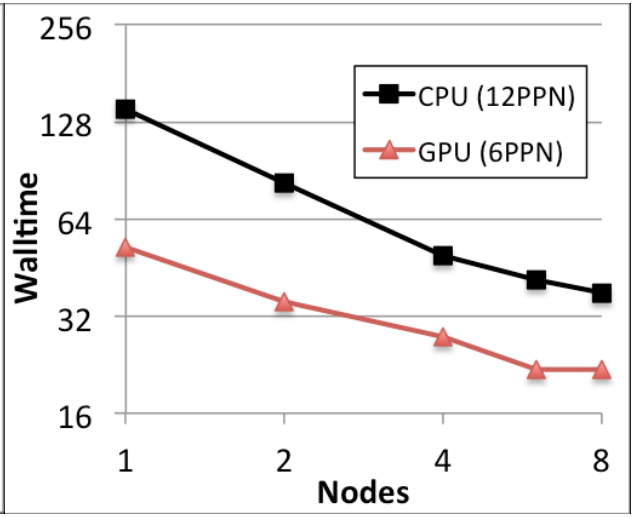
GPU LAMMPS Parallel Performance

Benchmarks: http://users.nccs.gov/~wb8/gpu/bench.htm

- **Also: http://sites.google.com/site/akohlmey/software/lammps-benchmarks**







864K atoms LJ liquid, reduced density=0.8442. NVE, r_{cut}= 2.5σ, 5000 steps.
Speedup ~ 3-4.

Gay-Berne ellipsoids
125 K atoms, NVE, r_{cut}= 7σ, 1000 steps.
Speedup ~ 9-11.

Rhodopsin protein in solvated lipid bilayer. CHARMM force field, long-range Coulombics via PPPM,SHAKE constraints. Counter-ions and reduced amount of water.
32K atoms, 1000 timesteps, LJ r_{cut}= 1 nm, neighbor skin of 1.0 σ, NPT.
Speedup ~ 1.7-3.

Yona cluster:
15 Nodes : 2x6-core AMD Opteron 2435 (2.6GHz) & **2 Tesla C2050** GPUs 3GB GDDR5 memory, 448 cores (1.15GHz) , memory bandwidth: 144GB/s. GPUs are connected on PCIx16 gen 2.0 slots. ECC support enabled. Mellanox MT26428 QDR InfiniBand interconnect.

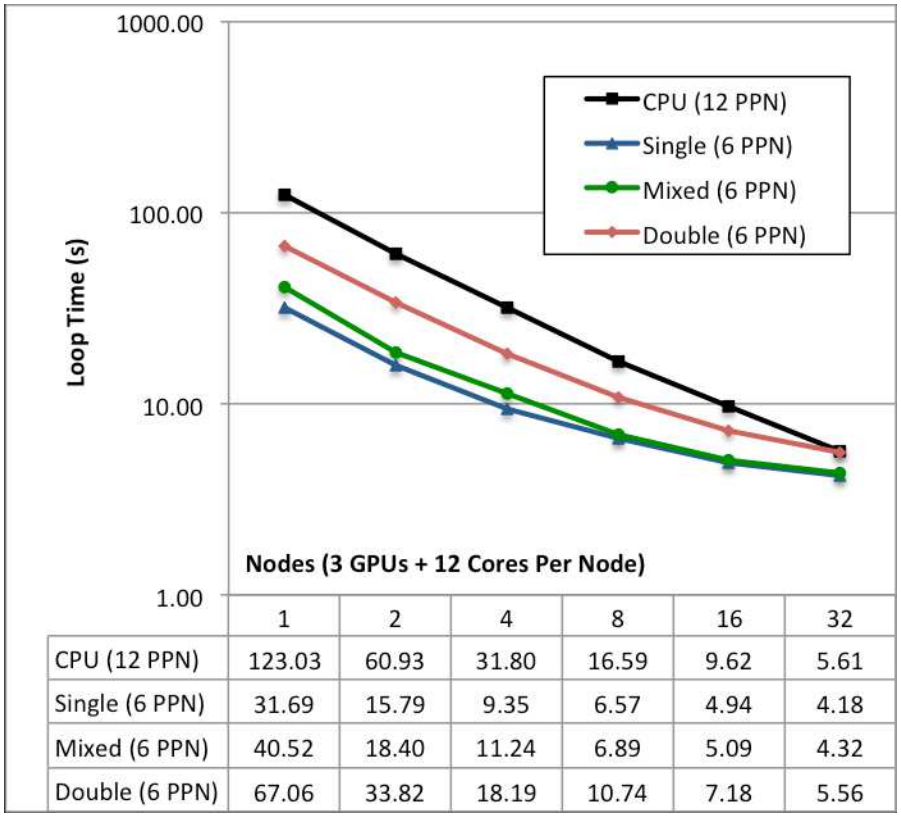
Single vs. Double Precision (lib compile option)

- **Force kernels can be compiled to use single, double, or mixed precision.**
- **The drawback of double precision for memory-bound kernels is that twice as many bytes must be fetched for cutoff evaluation.**
- **A potential solution is to use mixed precision. In this case, the positions are stored in single precision, but accumulation and storage of forces, torques, energies, and virials is performed in double precision.**
- **Because this memory access occurs outside the loop, the performance penalty for mixed precision is very small.**

Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington.Comp. Phys. Comm. **182** (2011) 898–911

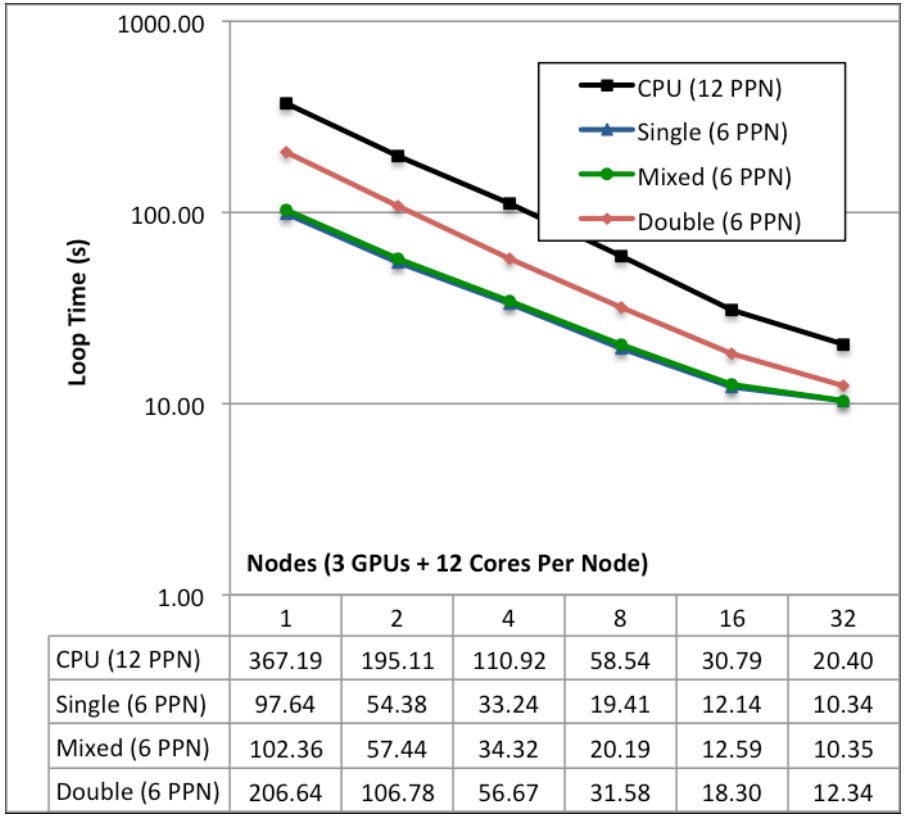
Benchmarks for different precision modes

- Single precision OK for many runs, but use at your peril! Colberg & *Höfling*, Comp. Phys. Comm. **182** (2011) 1120–1129.
- Mixed precision (single for positions and double for forces) nearly as fast as single precision!
- Double precision still cheaper than CPU.



	1	2	4	8	16	32
CPU (12 PPN)	123.03	60.93	31.80	16.59	9.62	5.61
Single (6 PPN)	31.69	15.79	9.35	6.57	4.94	4.18
Mixed (6 PPN)	40.52	18.40	11.24	6.89	5.09	4.32
Double (6 PPN)	67.06	33.82	18.19	10.74	7.18	5.56

256000 atoms LJ liquid, reduced density=0.8442. NVE, r_{cut}= 2.5σ, 5000 steps.



	1	2	4	8	16	32
CPU (12 PPN)	367.19	195.11	110.92	58.54	30.79	20.40
Single (6 PPN)	97.64	54.38	33.24	19.41	12.14	10.34
Mixed (6 PPN)	102.36	57.44	34.32	20.19	12.59	10.35
Double (6 PPN)	206.64	106.78	56.67	31.58	18.30	12.34

Rhodopsin protein in solvated lipid bilayer. CHARMM force field, long-range Coulombics via PPPM,SHAKE constraints. Counter-ions and reduced amount of water to make a 32K atom system, replicated 2x2x2 to create box.
256,000 atoms, 1000 timesteps, LJ r_{cut}= 1 nm, neighbor skin of 1.0 σ, NPT.

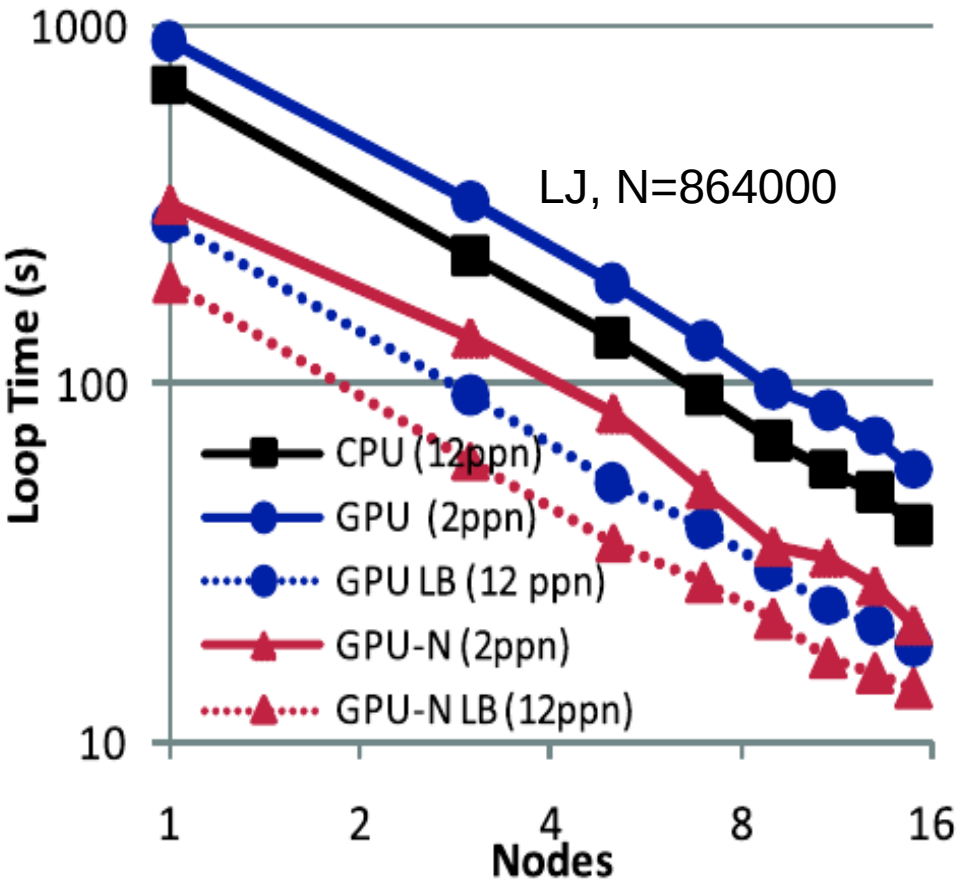
Fixed or Dynamic Load balancing?

Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington. Comp. Phys. Comm. **182** (2011) 898–911

- **Fixed load balancing: setting the CPU core to accelerator ratio and by setting the fraction of particles that will have forces calculated by the accelerator.**
- **Consider a job run with 4 MPI processes on a node with 2 accelerator devices and the fraction set to 0.7. At each timestep, each MPI process will place data transfer of positions, kernel execution of forces, and data transfer of forces into the device (GPU) queue for 70% of the particles.**
- **At the same time data is being transferred and forces are being calculated on the GPU, the MPI process will perform force calculations on the CPU.**
- **<u>Ideal fraction:</u> CPU time = GPU time for data transfer and kernel execution ➔ dynamic balancing with calculation of optimal fraction based on CPU/GPU timing at some timestep interval.**

Benchmarks for load balancing

- **The performance impact resulting from splitting the force calculation between the host and device will depend on the CPU core to device ratio and the relative rates of force calculation on the host and device.**
- **Processes per node (ppn).**
- **Dynamic Load Balancing (LB).**
- **Neighboring performed on the GPU (GPU-N).**

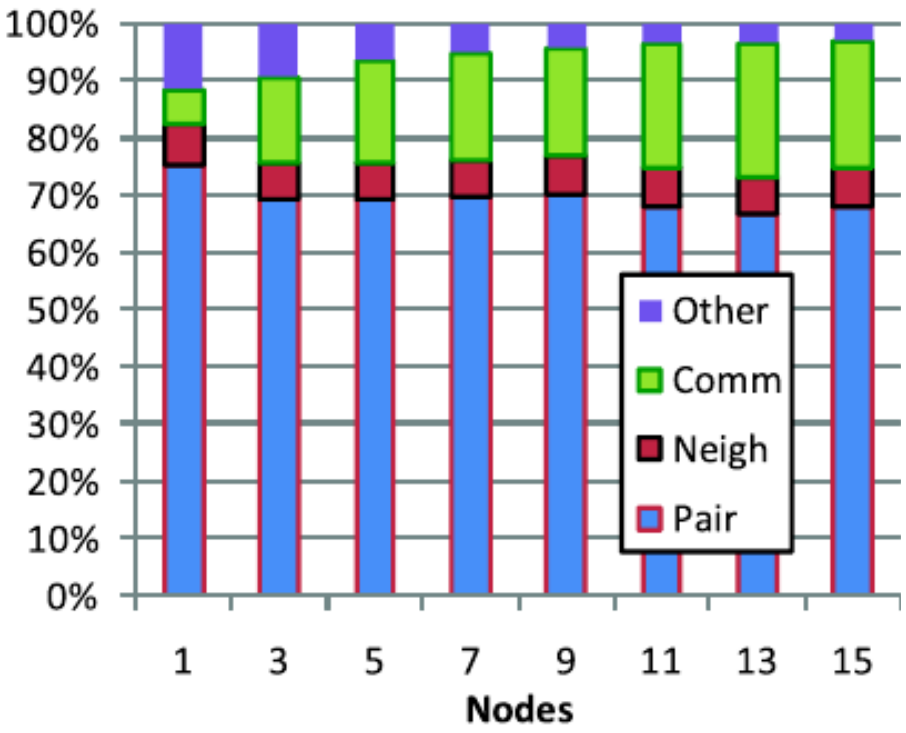


Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington. Comp. Phys. Comm. **182** (2011) 898–911

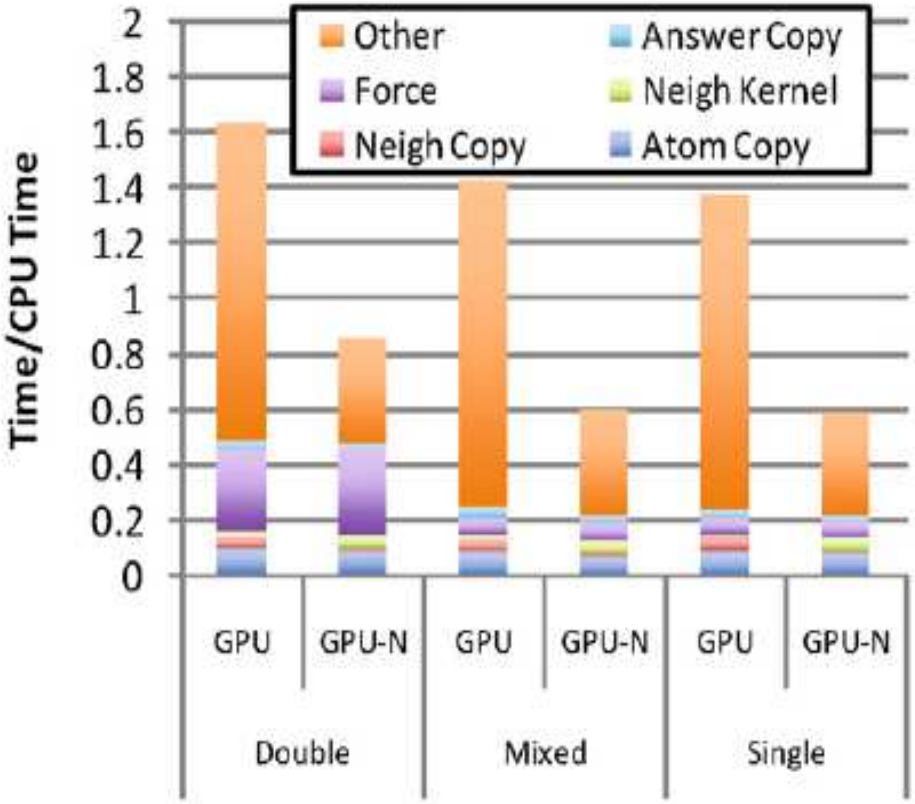
More benchmarks …

Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington. Comp. Phys. Comm. **182** (2011) 898–911

Strong scaling benchmark using LJ. cutoff of 2.5 and N=864 K.

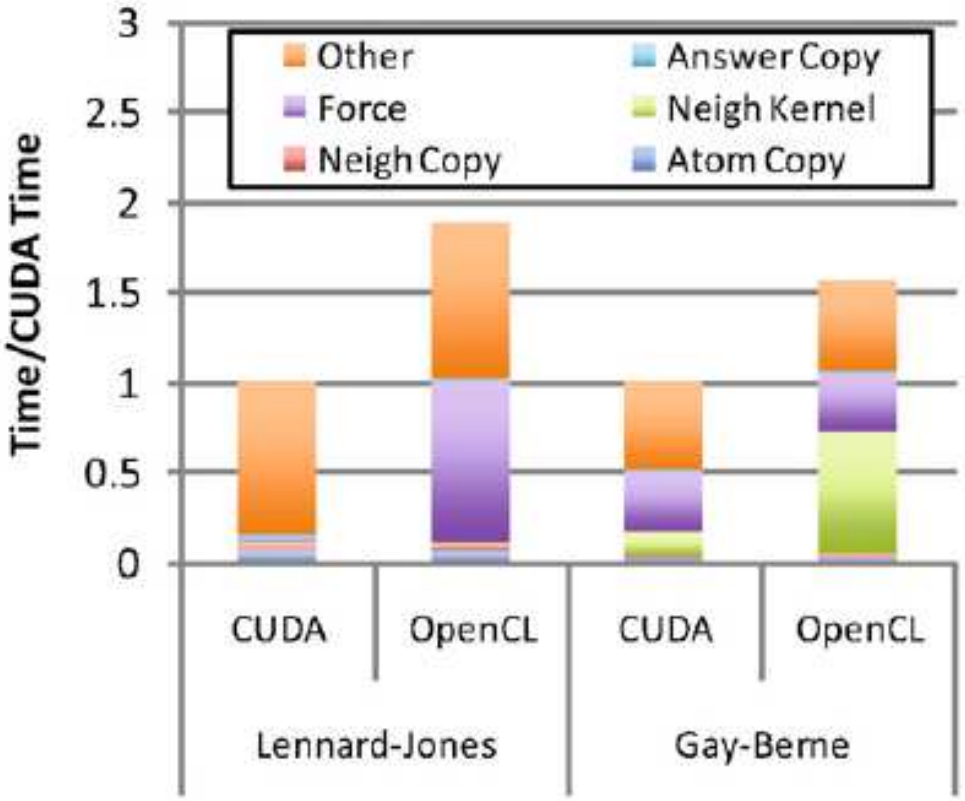


LJ. Single node.



LAMMPS: OpenCL vs CUDA

Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington. Comp. Phys. Comm. **182** (2011) 898–911



Single node. Code compiled with CUDA and OpenCL. N=256K with neighboring performed on the GPU. Time normalized by the time required to complete simulation loop with CUDA.

CPU versus CPU-GPU Speedups

Implementing molecular dynamics on hybrid high performance computers – short range forces
W. Michael Brown, Peng Wang, Steven J. Plimpton and Arnold N. Tharrington. Comp. Phys. Comm. **182** (2011) 898–911

Test case	1 node		15 nodes	
	Cores	Speedup	Cores	Speedup
LJ CPU	12	9.6	180	162.5
LJ GPU single	12	23.4	180	356.4
LJ GPU-N single	12	34.4	180	467.1
LJ GPU double	12	16.0	180	224.1
LJ GPU-N double	12	20.4	180	172.6
GB CPU	12	12.8	180	182.5
GB GPU single	12	146	30	1747.4
GB GPU-N single	12	144.1	30	1541.7
GB GPU double	12	37.2	30	511.4
GB GPU-N double	12	40.9	30	503.7

Outline

- Introduction
- LAMMPS
- GPU & USER-CUDA packages
- Performance:

a)single/mixed/double precision

b)CPU/GPU neighbor lists

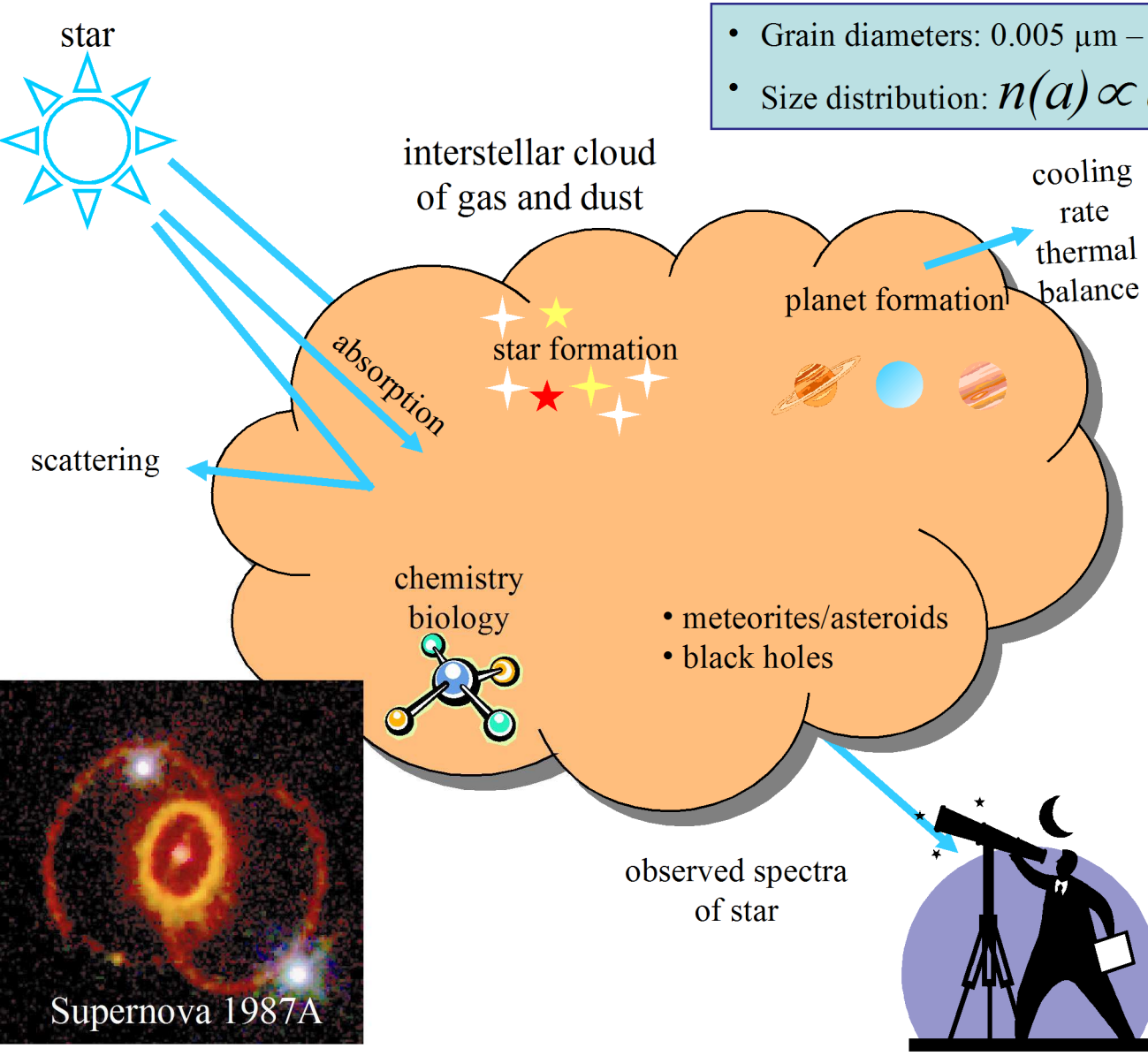
c)Load balancing: static & dynamic

- MD Examples.
- Other GPU projects

Interstellar dust plays an important role in astrophysical processes

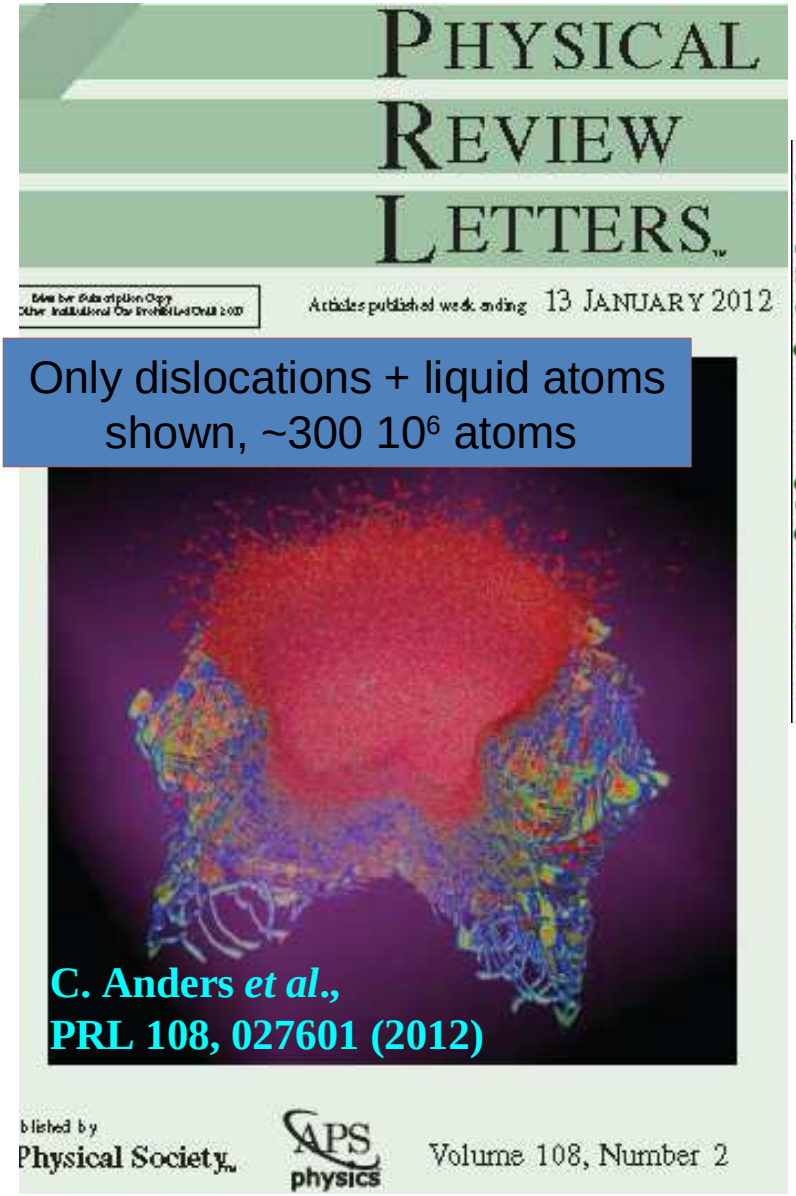
Grain size matters for evolution, astro-chemistry, etc.

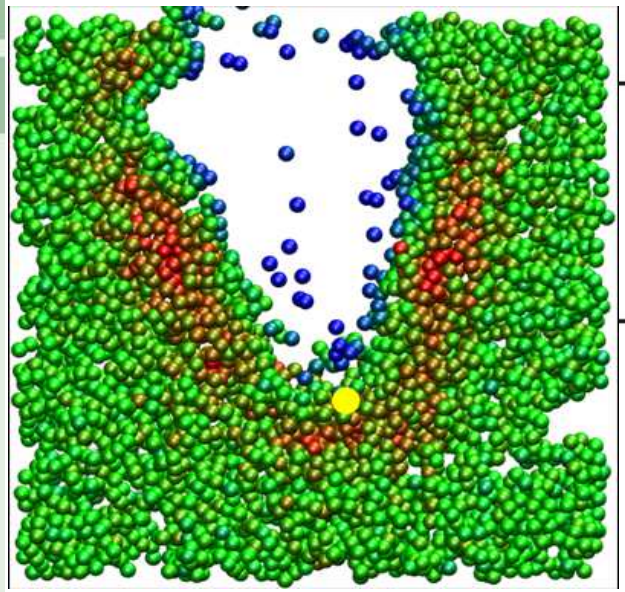
- Grain diameters: 0.005 μm – 5 μm
- Size distribution: $n(a) \propto a^{-3.5}$



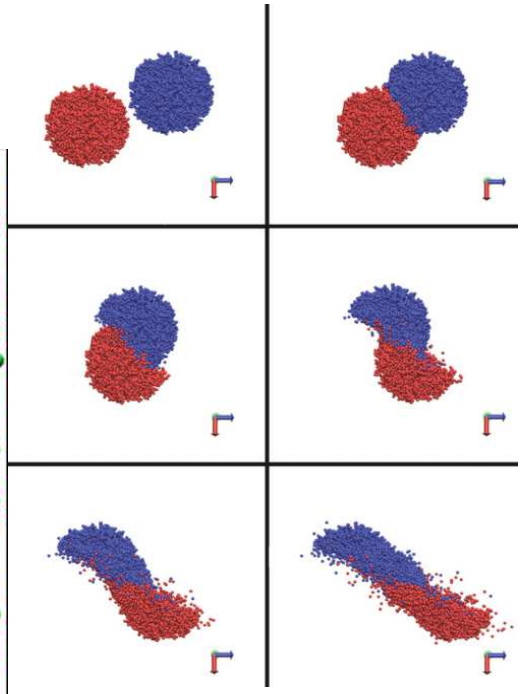
Modification of materials: phase transitions and damage to space hardware, and due to shrapnel (NIF)

Large-scale MD links nano and microscales in damage induced by nanoprojectiles





Granular mechanics of grain-surface collisions
Ringl *et al.*,
PRE 86, 061313 (2012)
PRE KALEIDOSCOPE

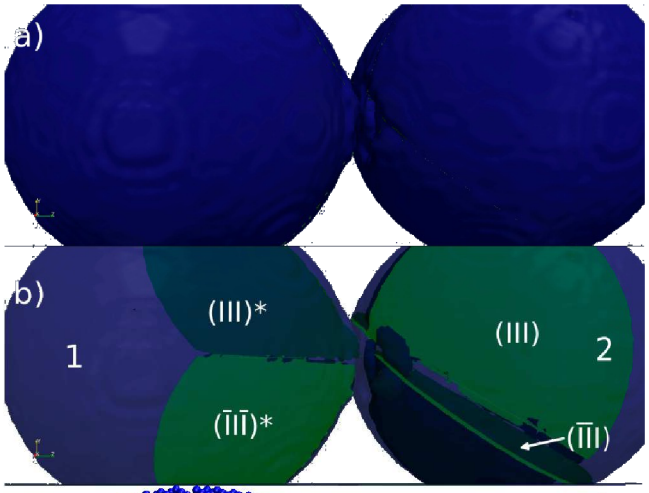


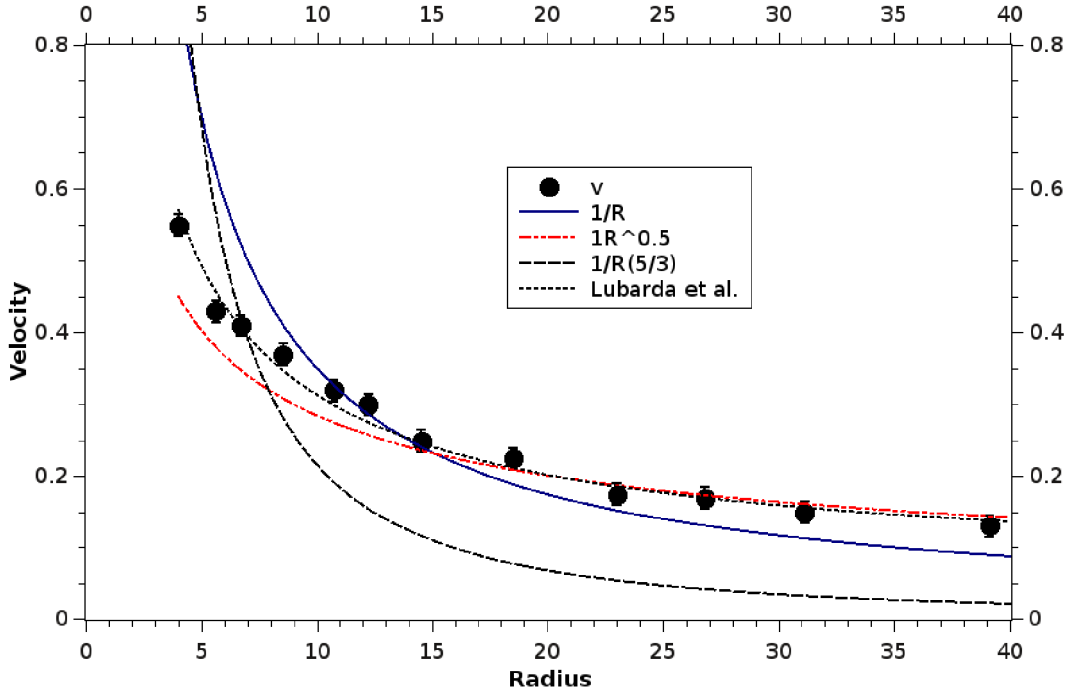
Granular mechanics of nano-grain collisions
Ringl *et al.*, Ap.J. **752** (2012) 151

New granular friction scheme implemented for GPUs by E. Millan

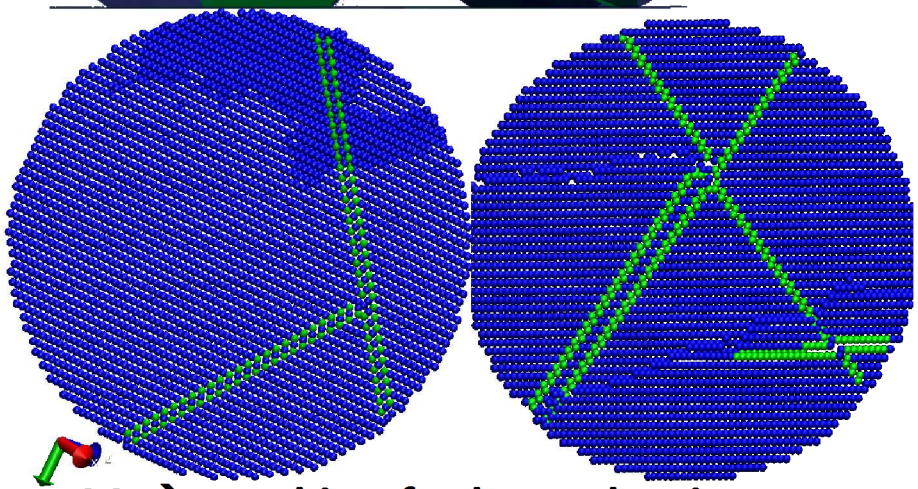
Plasticity threshold in grain-grain impact

Granular models assume lack of plasticity





Dislocation-based model by Lubarda *et al*. agrees with MD. Millan, Tramontina, *et al*., to be submitted (2013)



FCC → stacking faults and twins

Millan, Tramontina, *et al*., Anales MACI (2013)

GPUs + CPUs to run ~10000 independent MD simulations

“Numerical” experiments using LAMMPS: parameter sweep for cluster collisions

Goal: reduce the makespan of multiples jobs executing parallel processes both in the CPU and GPU.

Ad-hoc strategy: split jobs bewteen CPU&GPU. Could be improved with job scheduling tools.

Different parallel modes considered:

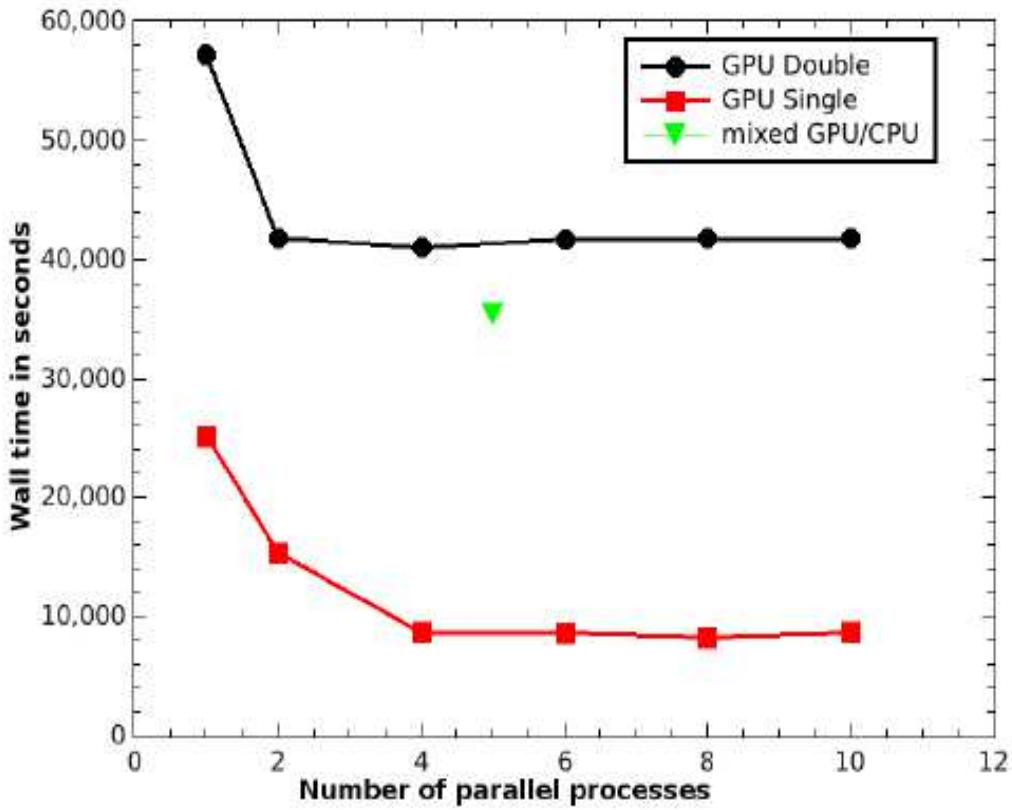
- Process parametric study on multicore CPU workstation using OpenMPI.
- Process parametric study on the GPU.
- Hybrid studies: RUBY script to assign workload both to CPU and GPU according to predefined strategy. MPI is considered for this case as well, and Dynamic or static load balancing can be considered.
- Only up to 10 simultaneous jobs due to memory limitations.

Nanograin collisions – GPU vs CPU performance

	Makespan	Speedup
CPU 1 process	308250 s	
CPU 6 processes	60334 s	5.1x
GPU double 4 processes	41070 s	7.5x
GPU 4 processes / CPU 1 process x 2 cores	35511 s	8.6x

Parallel simul. CPU	CPU processes (-np)	Parallel simul. GPU	GPU Double	Distr.	
				gpu	cpu
6	1	0	60334 s	0	150
0	0	4	41070 s	150	0
1	2	4	40038 s	145	5
			38932 s	140	10
			35511 s	130	20
			38658 s	120	30
			48218 s	110	40

Parallel simulations	GPU double	GPU single
1	57238 s	25233 s
2	41885 s	15409 s
4	41070 s	8755 s
6	41731 s	8659 s
8	41804 s	8304 s
10	41808 s	8760 s

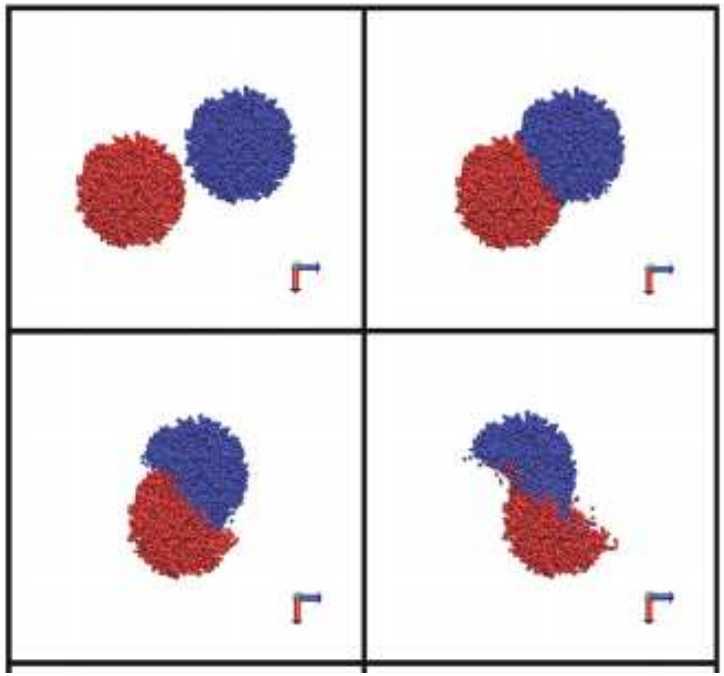


150 simulations. GPU/CPU test is for 5 procs running independent simulations in parallel using six CPU cores, one CPU process with 2 cores and 4 GPU procs.

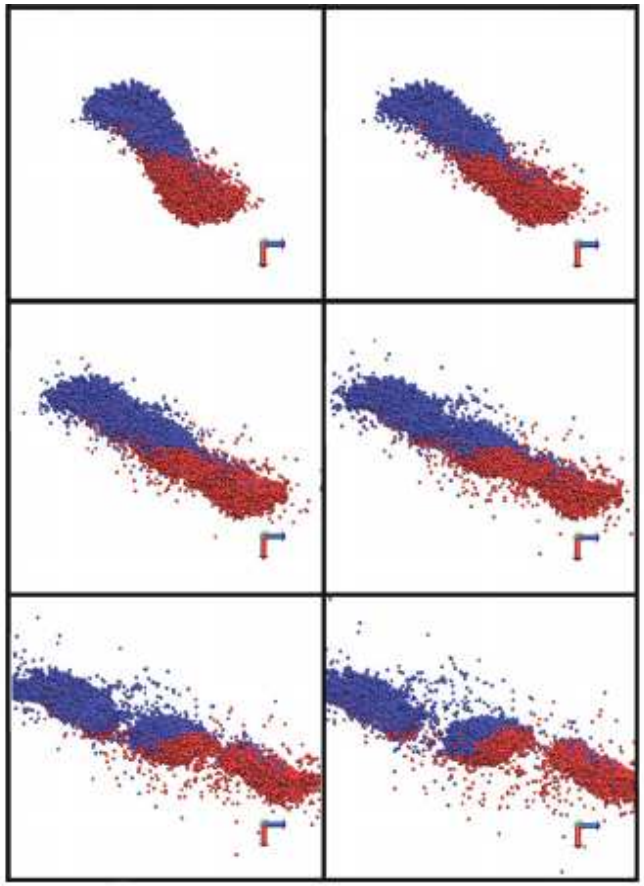
Emmanuel N. Millán, et. al. XVIII CACIC, http://sedici.unlp.edu.ar/handle/10915/23634

Modified Granular simulations on LAMMPS

Can simulate more than 1 million particles in workstation.
Improved adhesive forces and friction (gliding, rolling, and torsional).



Velocity v = 5 m/s (v = 29_{vfrag}) & impact parameter b = 0.8 R



C. RINGL AND H.M. URBASSEK, A LAMMPS implementation of granular mechanics: Inclusion of adhesive and microscopic friction forces, Computer Physics Communications 183 (2012), pp 986-992.

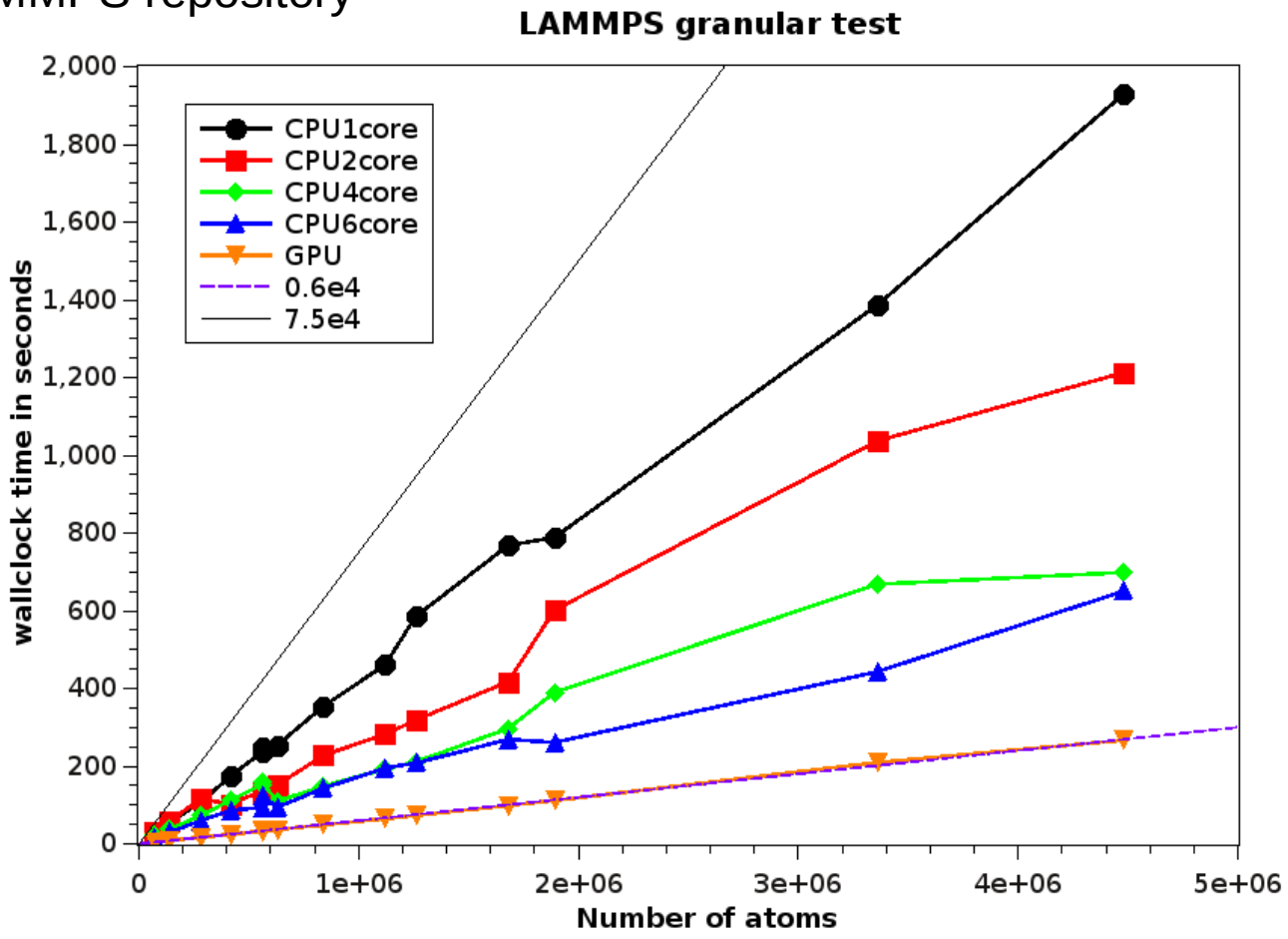
GRANULAR simulations Benchmarks in GPU (extension of USER-CUDA)

GPU version developed by Emmanuel N. Millán
CPU version developed by Christian Ringl
Code to be submitted to LAMMPS repository

The 7.5e4 curve represents the results obtained in C. Ringl, Comp. Phys. **183**, 986 (2012).

CPU: AMD Phenom x6 1055t 2.8GHz
GPU: NVIDIA Tesla C2050

AVG speedup
GPU vs **1** CPU core = 7x
GPU vs **6** CPU core = 2.95x



Other GPU-related projects

1) MD and MC simulations to understand friction
S. Manzi & E. Perino (UNSL)

2) Image processing for medical applications
F. Fioretti (ICB, UNCuyo) & Roberto Isoardi (FUESMEN)

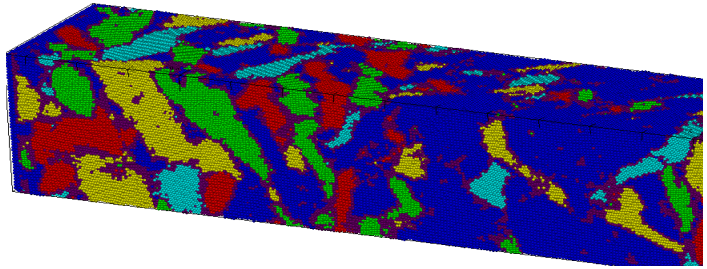
3) MC simulations of settlements in arid environments
L. Forconesi (ITU), S. Goiran & J. Aranibar (ICB, CCT-Mendoza)

4) Cellular automata (CA) and agent-based modeling (ABM)
C. Garcia Garino (UNCuyo-ITIC), M.F. Piccoli, M. Printista (UNSL).

5) Reaction-diffusion equation to model foams
D. Schwen, A. Caro (LANL), D. Farkas (Va Tech)

6) Simulated X-ray diffraction from large-scale atomistic simulations
A. Higginbotham, M. Suggit, J.S. Wark (U. Oxford).

Suggit *et al*, submitted to Phys. Rev. B (2013)



Opportunities for interested students!

Simulation of settlement dynamics in arid environments

Problem: Which factors influence the livestock settlements spatial distribution in the NE of Mendoza? Use Monte Carlo simulation

E. Millan (ICB), C. Garcia Garino (ITIC), L. Forconesi (ITU), S. Goiran & J. Aranibar (ICB, CCT-Mendoza)

Complexity: large parameter space + neighbor search

Objetive: Find optimal solution, minimizing error.

Variables (GIS):

Distance to road

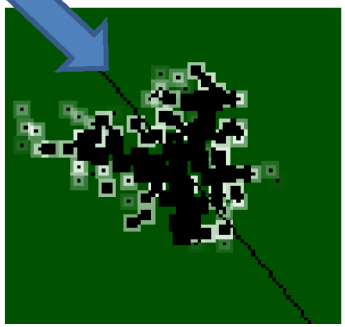
Distance to river

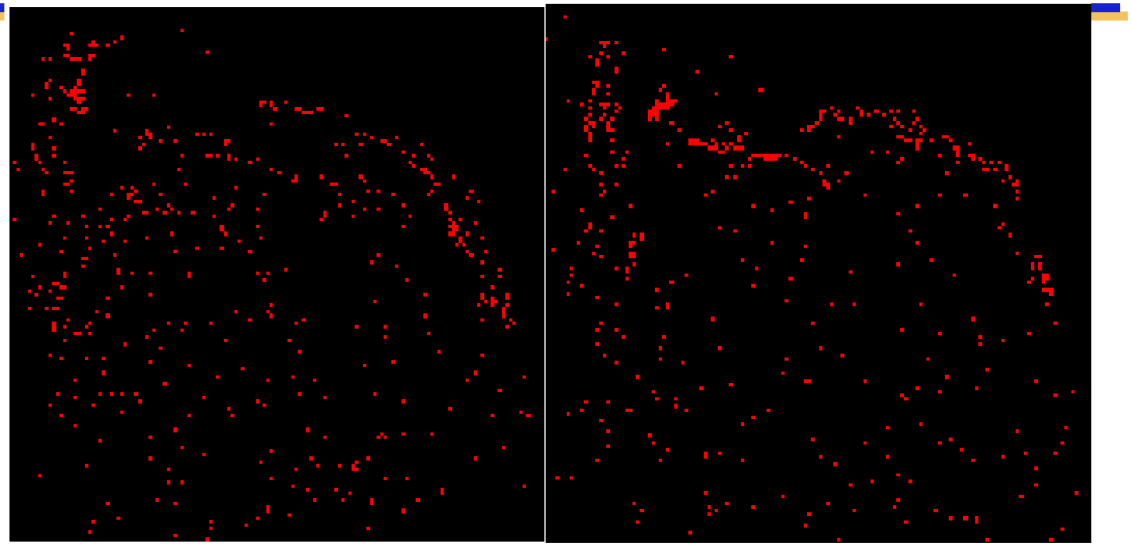
Settlements distance

Water table depth

Vegetation degradation

(need neighbors!!)





One is real, the other simulated: which is which?

Millan *et al*., submitted to Ecological modeling (2013)

Need ~1.2 million independent simulations:

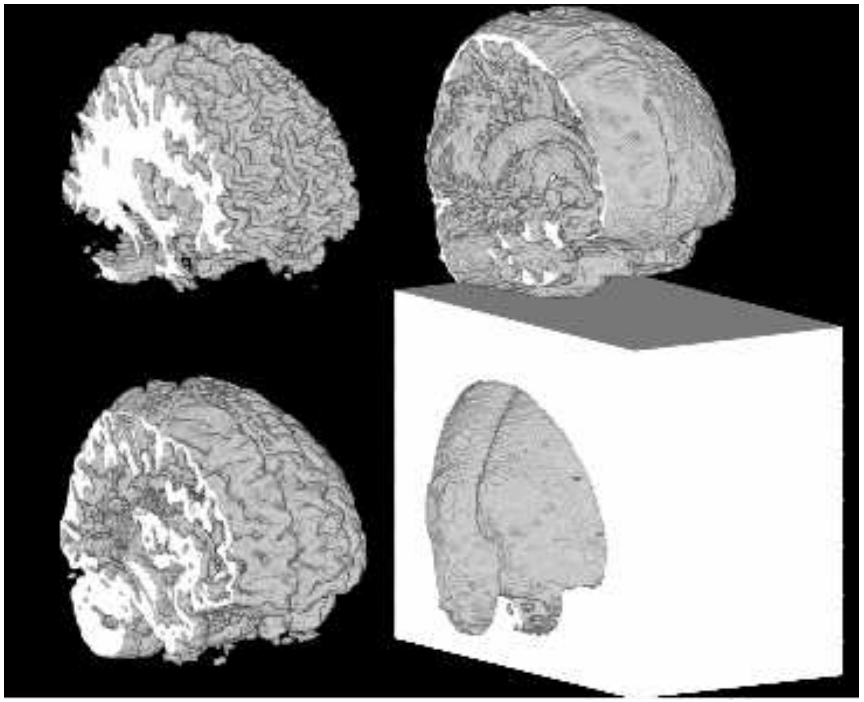
a) use multicores

b) use GPU: speed-up?

Image processing for medical applications

F. Fioretti, D. Lemma, E. Millan, E. Bringa (ICB, UNCuyo)& R. Isoardi (FUESMEN)

“Implementation of a Bayesian Statistics Segmentation Algorithm in GPUs”, HPCLatAm (2012) conference proceedings.

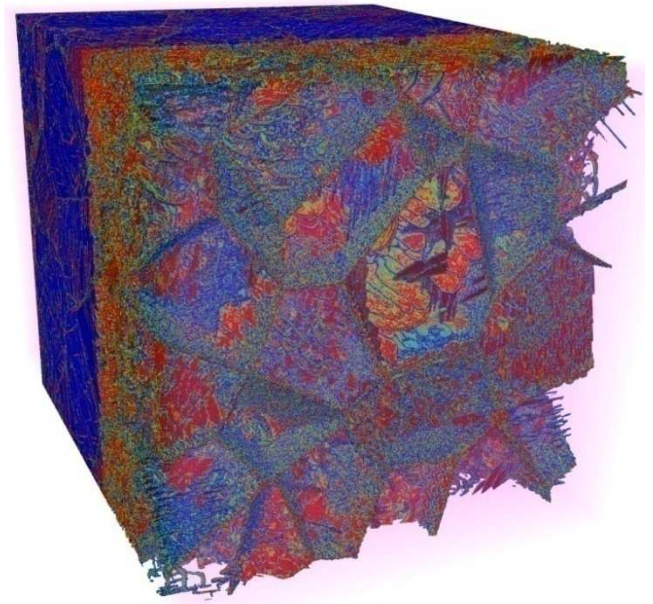


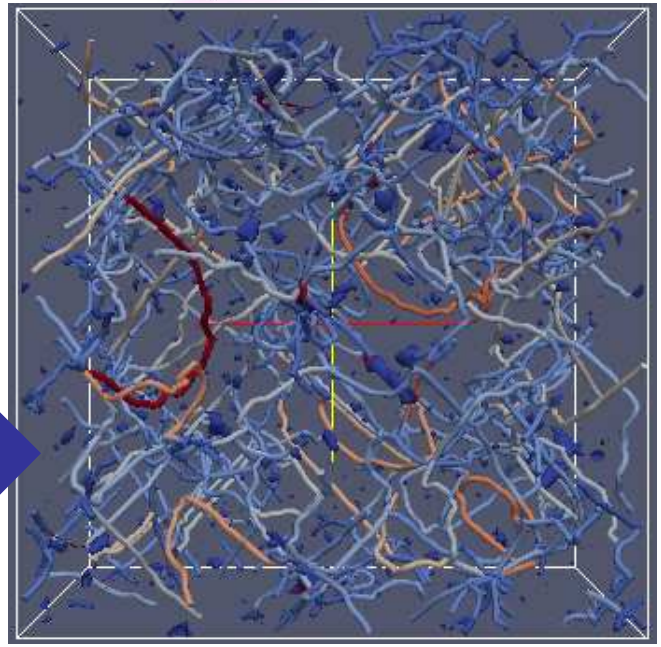
Data reduction in GPU to clasify voxels as:
white matter, gray matter or cerebrospinal fluid.

Speed-up: 60x in Tesla 2050 respect to original CPU code.
15x respect to highly optimized CPU-OMP code.

Future perspectives

- Multicore +GPU clusters: challenges on load balancing, communication and synchronization.
- MD often requires more CPU/GPU time for analysis than for time evolution (months versus days) ➔ need parallel processing for big data.
- Need smart algorithms which scale well in parallel archs taking advantage of link-cells.
- Need to use parallel viz tools for samples with +few million atoms (generally not the case in chemistry/biology).
- New computers and novel algorithms are allowing comparison of simulation and experimental data.
- GPU processing has bright future!





Cluster ICB-ITIC + Escuela HPC + HPCLatAm 2013

Cluster ICB-ITIC

2 nodos con 64 cores, 128 GB RAM, ~20 TB disco, 1 Tesla 2070
(~$135,000, fondos Agencia Bringa, Santos, Aranibar, Del Popolo)
Cluster ITIC (~$60,000, fondos Agencia Garcia Garino)
Adherido a SNCAD, fondos para (poder de computo)x4, incluyendo 4 PICTs y PME ICB

Escuela de Computación de Alto Rendimiento (ECAR 2013)

ECT, UNCuyo, 22 de Julio al 26 de Julio de 2013, 40 becas para estudiantes

http://ecar2013.hpclatam.org/

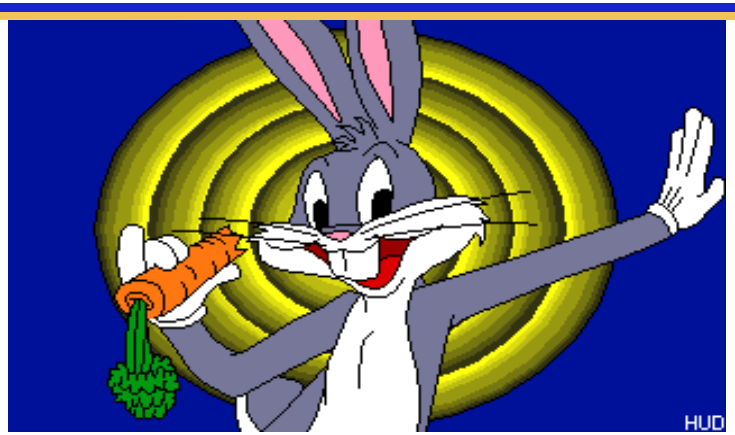
VI Latin American Symposium on High Performance Computing HPCLatAm 2013

ECT, UNCuyo, July 29-30, 2013.

http://hpc2013.hpclatam.org/

HPC
High-Performance Computing - LATAM

That's all folks!



SiMAF:
Simulations in Materials Science, Astrophysics, and Physics

SiMAF
Simulaciones en Materiales Astrofísica y Física

https://sites.google.com/site/simafweb/

Web master: M.J. Erquiaga; design: E. Rim

Funding: PICT2008-1325, PICT2009-0092, SeCTyP U.N. Cuyo