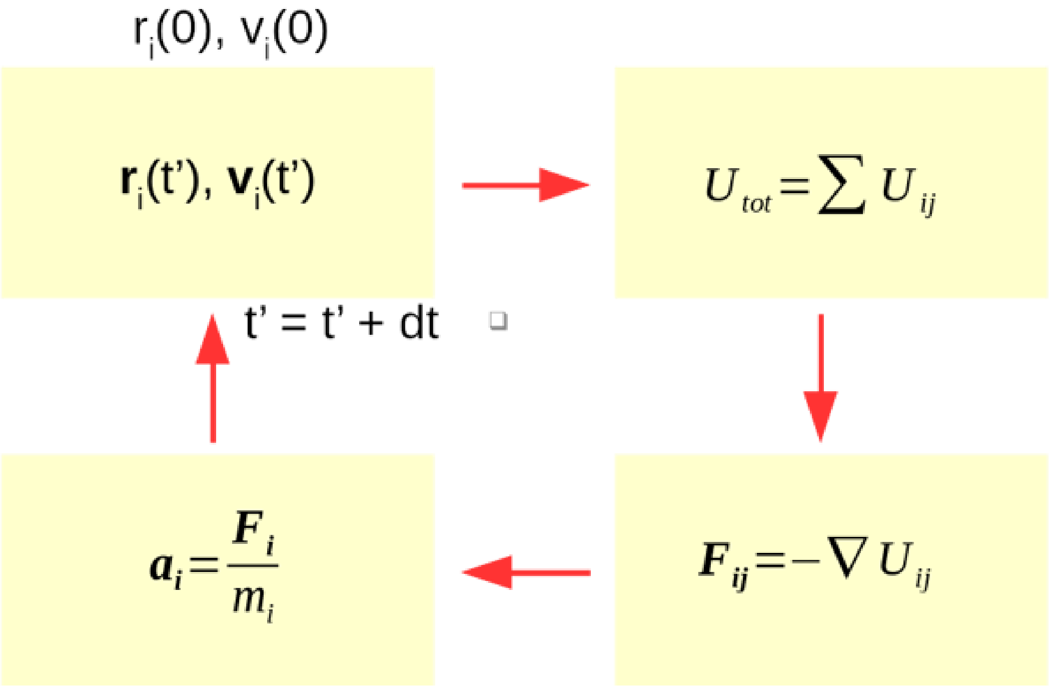
Topic: Molecular dynamics

Objective

- Molecular dynamics (MD)
- Short-range and long-range forces
- Neighbor lists – Verlet and cell lists
- Ingredients for running MD simulations
- Properties from MD simulations
- MD Fortran code for Argon
- Generate initial configurations
- Visualize the trajectory in VMD

Molecular dynamics (MD)



$r_i(0)$, $v_i(0)$

$\mathbf{r}_i(t')$, $\mathbf{v}_i(t')$

$U_{tot} = \sum U_{ij}$

$\boldsymbol{F}_{ij} = -\nabla U_{ij}$

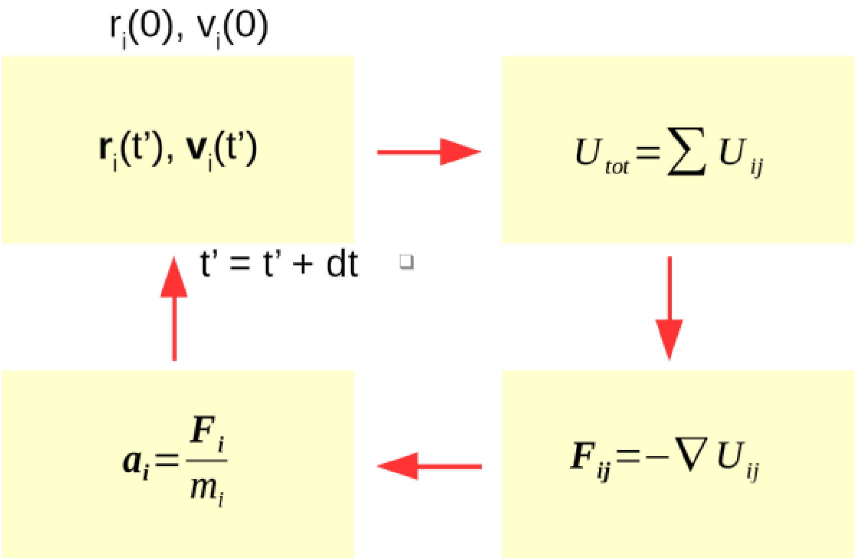
$\boldsymbol{a}_i = \frac{\boldsymbol{F}_i}{m_i}$

t' = t' + dt

Molecular dynamics (MD)

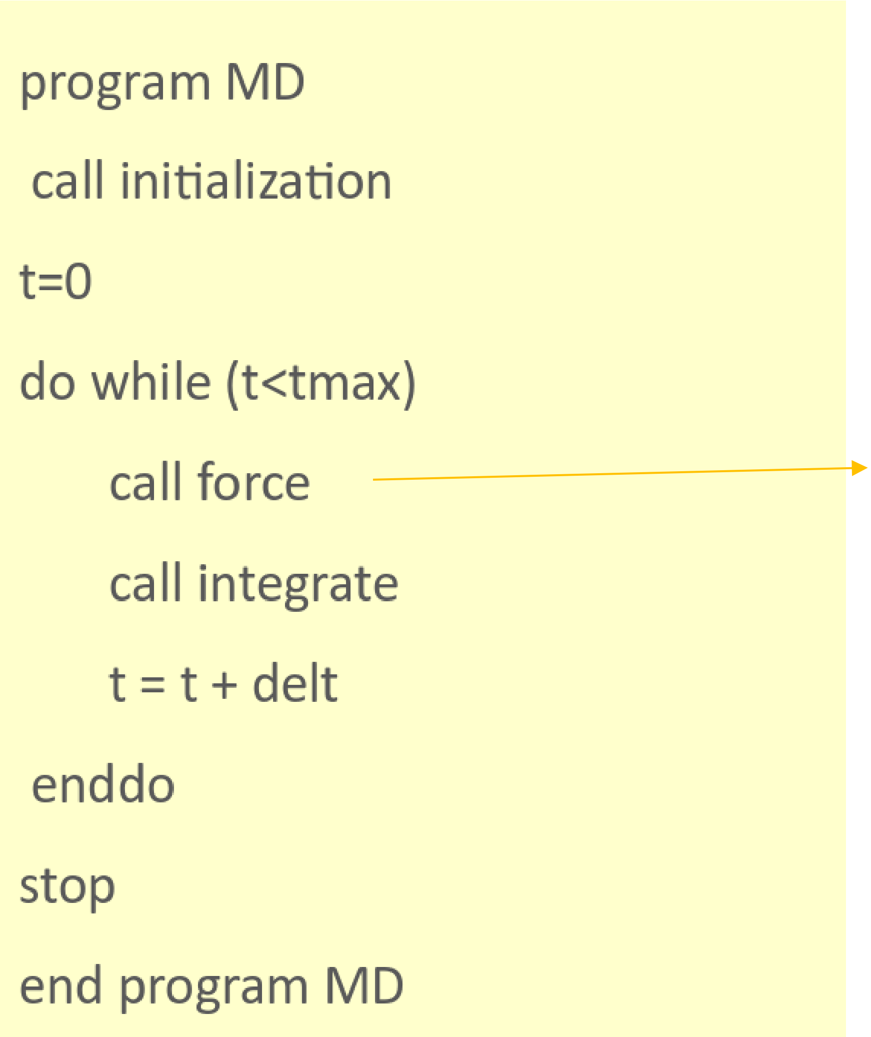
Mainly involves two parts

- Compute forces on all atoms/particles - computationally expensive
- Integration to update the positions and velocities



Topic: Molecular dynamics

Simple MD program



```
program MD
 call initialization
t=0
do while (t<tmax)
        call force
        call integrate
        t = t + delt
 enddo
stop
end program MD
```

computationally expensive

Topic: Molecular dynamics

Calculating forces

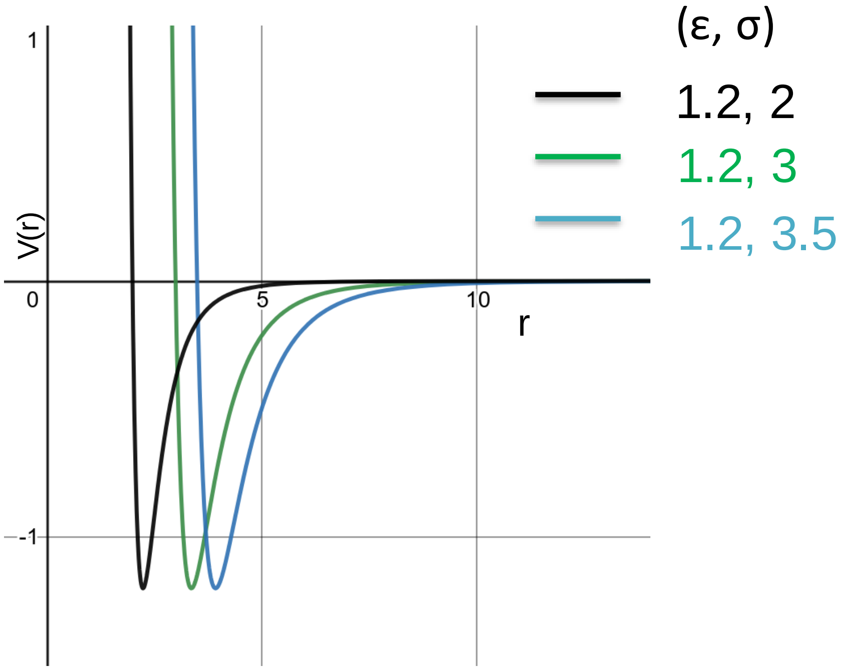
- Bonded forces (bond, angle, etc.) are local computations – not expensive
- Non-bonded forces
  - Van der Waals forces (short-range) – moderately expensive
  - Electrostatic forces (long-range) – very expensive
- Parallelization of MD is mainly associated with
  - parallelization of force calculations and
  - Improve serial performance - reducing the number of iterations (< $O(N^2)$) for force calculation

Topic: Molecular dynamics

vdW forces

- Van der Waals forces (short-range) – moderately expensive

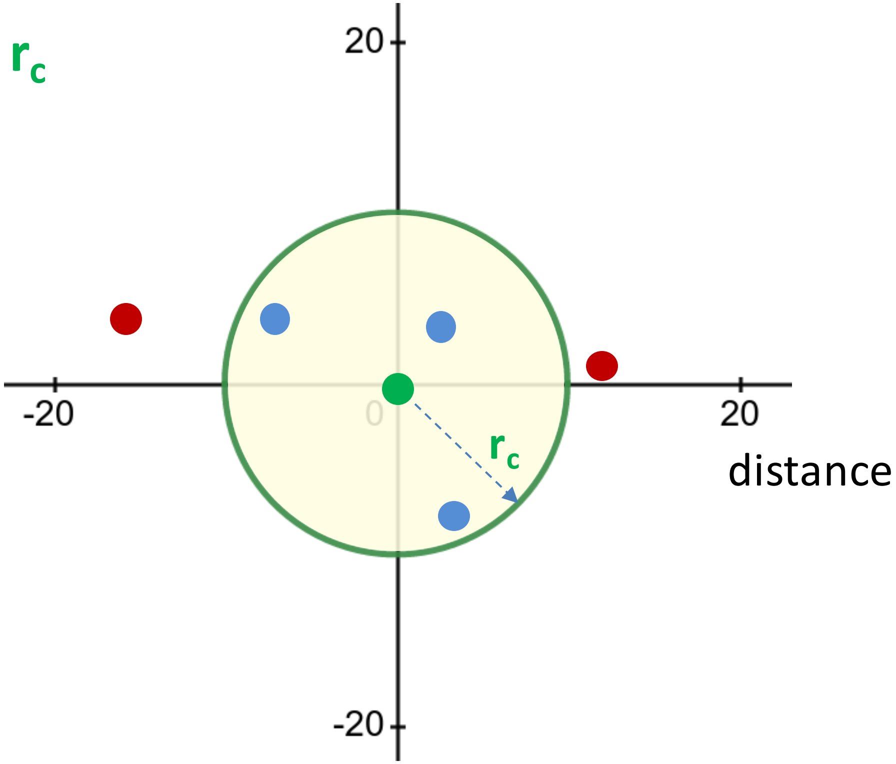
$$V(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right]$$



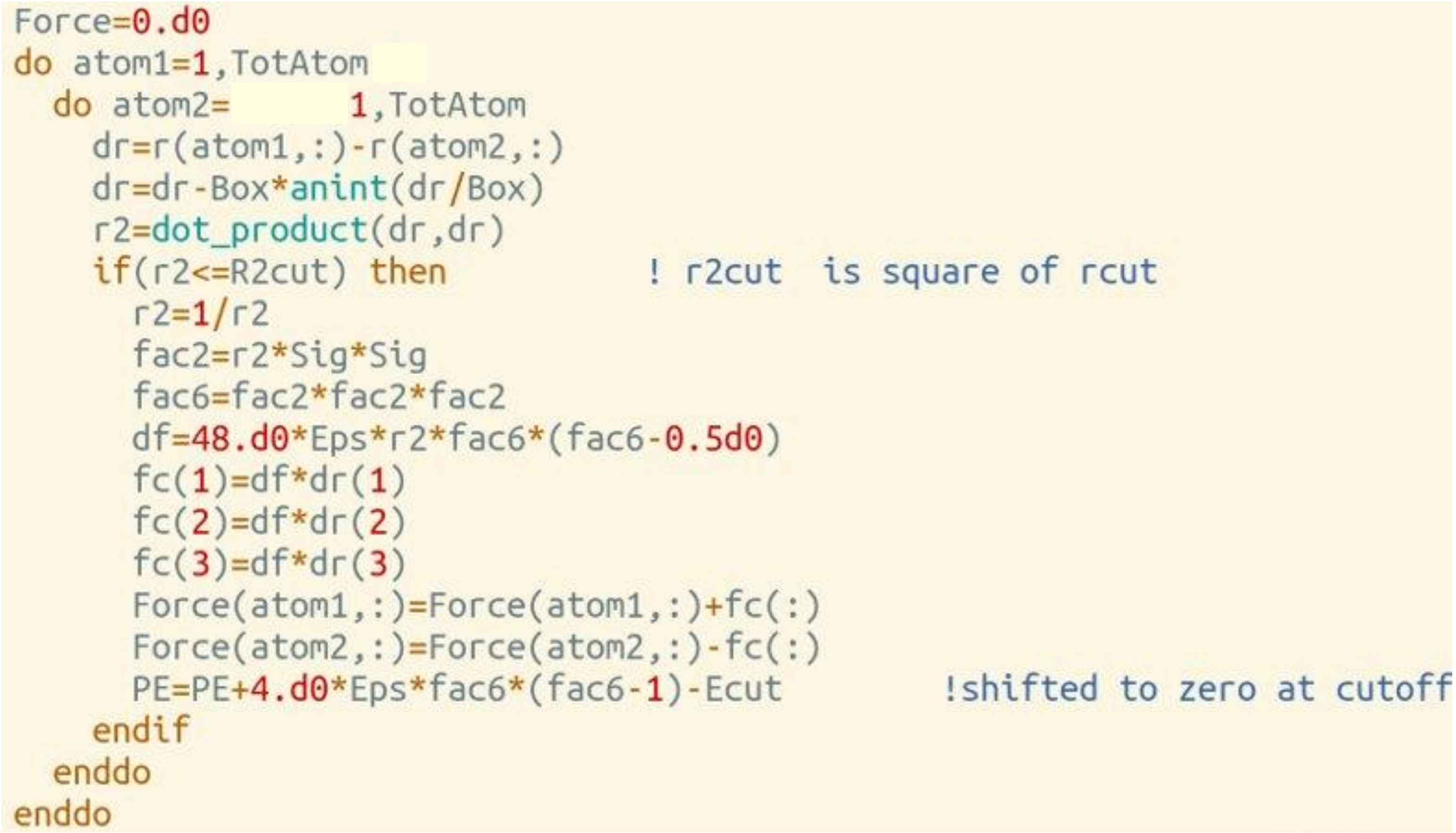
- Interaction goes to zero beyond a certain distance – depends on the value of σ
- Recommended cutoff - 2.5 σ
- Computational complexity – O(N^2)

vdW forces – speed up the calculations

- Energy and forces are computed only when the distance between two particles/atoms is less than $\mathbf{r_c}$



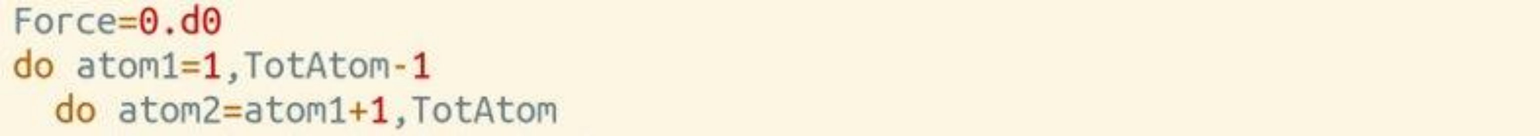
vdW forces – sample Fortran code



```
Force=0.d0
do atom1=1,TotAtom
  do atom2=      1,TotAtom
    dr=r(atom1,:)-r(atom2,:)
    dr=dr-Box*anint(dr/Box)
    r2=dot_product(dr,dr)
    if(r2<=R2cut) then          ! r2cut  is square of rcut
      r2=1/r2
      fac2=r2*Sig*Sig
      fac6=fac2*fac2*fac2
      df=48.d0*Eps*r2*fac6*(fac6-0.5d0)
      fc(1)=df*dr(1)
      fc(2)=df*dr(2)
      fc(3)=df*dr(3)
      Force(atom1,:)=Force(atom1,:)+fc(:)
      Force(atom2,:)=Force(atom2,:)-fc(:)
      PE=PE+4.d0*Eps*fac6*(fac6-1)-Ecut        !shifted to zero at cutoff
    endif
  enddo
enddo
```

Topic: Molecular dynamics

vdW forces

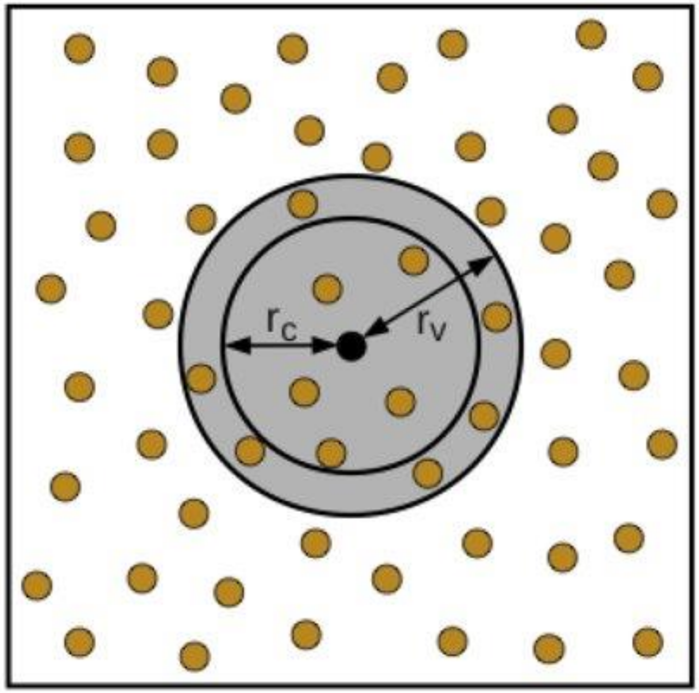


```
Force=0.d0
do atom1=1,TotAtom-1
  do atom2=atom1+1,TotAtom
```

- Even after truncating the potential, we still have to compute N*(N-1)/2 pair distances
- Without any tricks, the time taken for force evaluation is proportional to N^2
- Speedup can be achieved with
  - Verlet list
  - Cell (or linked) list
  - Verlet + Cell list

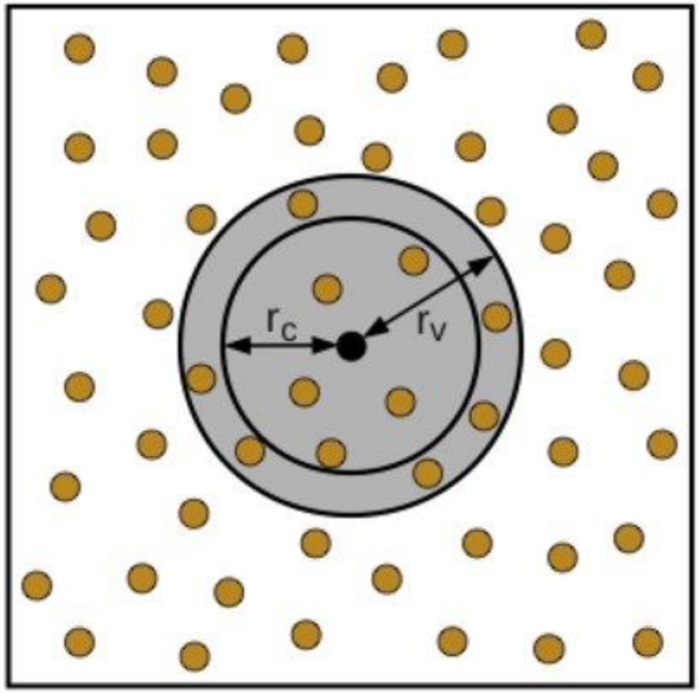
Improves serial performance

Neighbour lists – Verlet neighbor lists



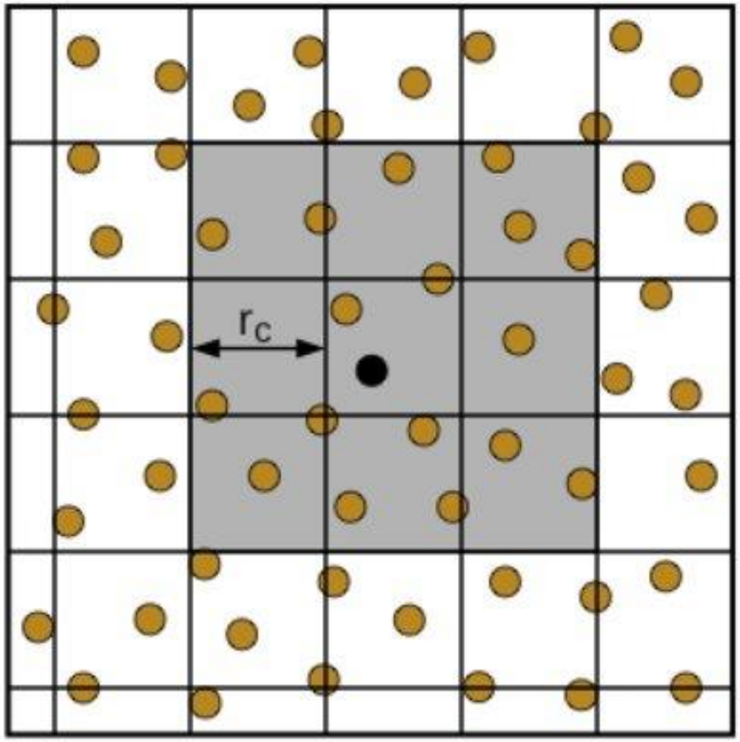
- Verlet list is a bookkeeping technique that contains list of all particles within the radius $r_{i,v}$ around each atom 'i'.
- List is prepared for each atom before calculating the forces whenever the maximum displacement of the atoms is greater than r_c- r_v
- Scales as O(N x Nv) where 'N' is the total number of particles, 'Nv' is the avg. number of particles within cutoff

Neighbour lists – Verlet neighbor lists



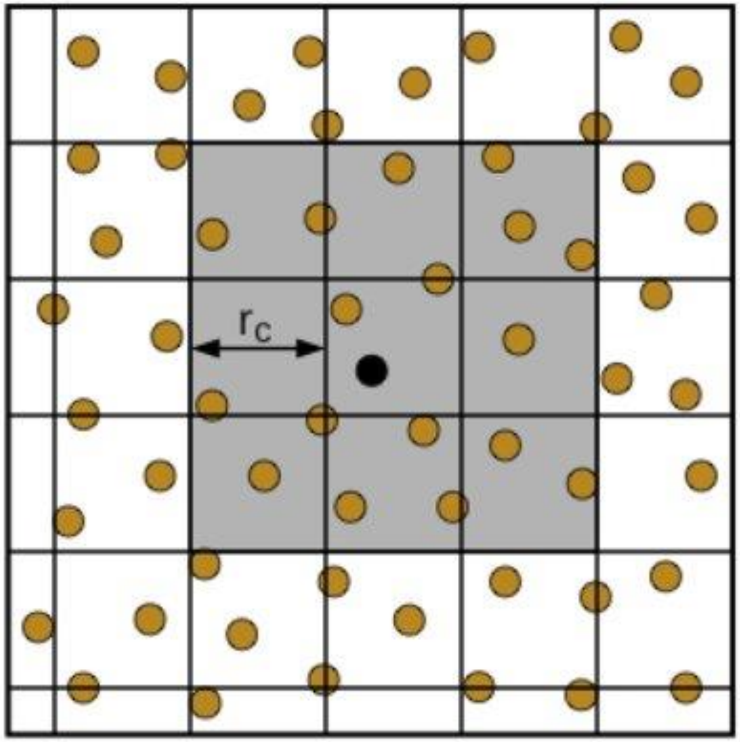
- $r_v - r_c = s =$ skin distance
- Larger skin distance (s), less often we build neighbor lists, but more pair of atoms will be checked for force calculation at every timestep
- Optimal value of skin distance must be chosen for achieving maximum computational efficiency

Neighbour lists – cell lists method



Scales as O(N x Nc) where 'N' is the total number of particles, 'Nc' is the avg. number of particles in 9 cells

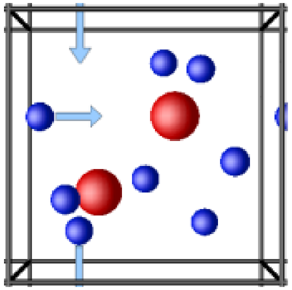
Neighbour lists – verlet + cell lists method



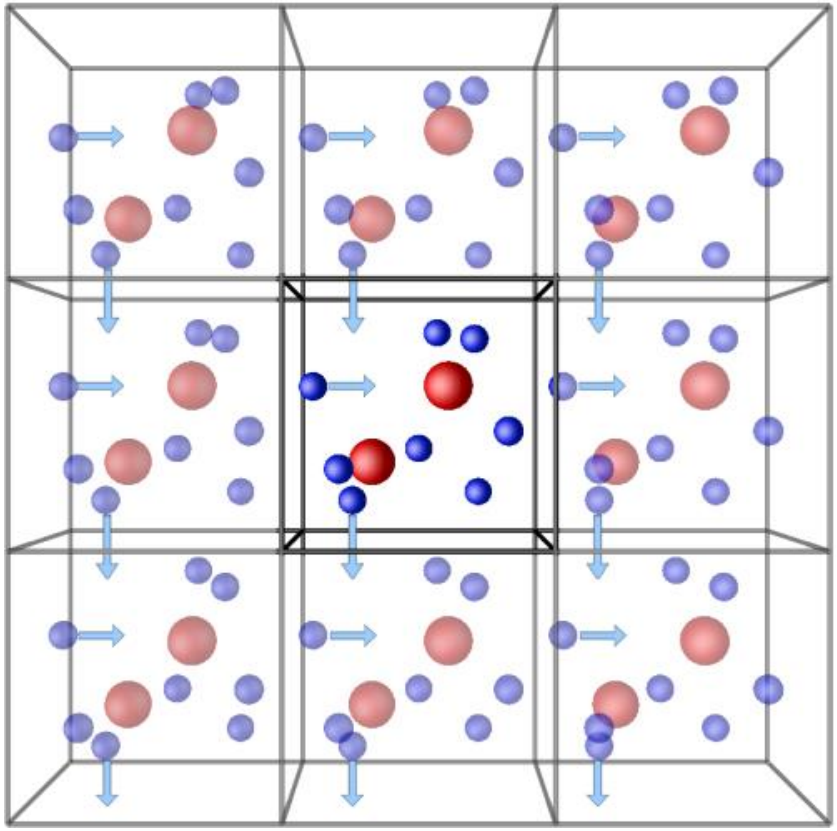
Used when neighbour list needs to be regenerated

Surface atoms

- Atoms near the faces of the box have different environment
- Surface atoms are present



Periodic boundary conditions



No surface atoms !

Periodic boundary conditions

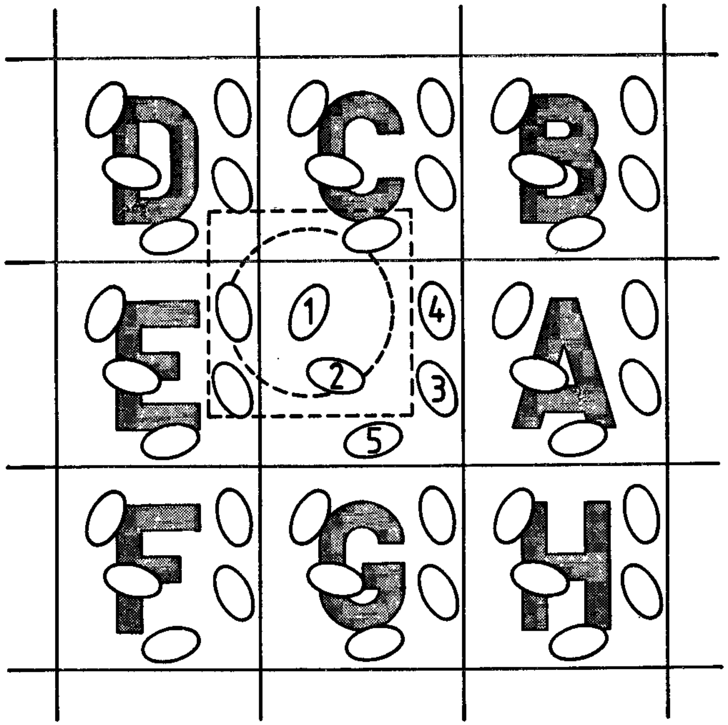
- Advantages: simulations can be performed in a relatively small number of atoms/particles
- If an atom/particle leaves simulation box, it is replaced by its image atom/particle
- 2D box – 8 neighbors; 3D box – 26 neighbors
- Disadvantage: small box sizes can lead to inconsistency in force/energy calculation
- Coordinates of the Image atoms/particles can be obtained from

  $x' = x + n1L \qquad y' = y + n2L \qquad z' = z + n3L$

  where

  $n1/n2/n3=-1,0,1$

Minimum image convention

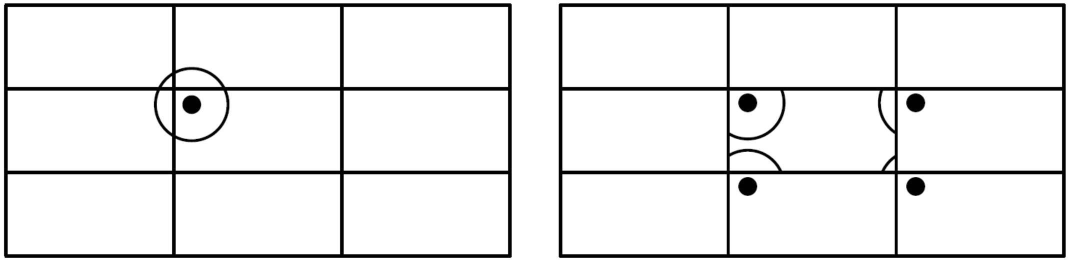


- Each i^{th} particle interacts with other particles as well as their images i_A, i_B..in neighboring boxes.
- Infinite no. of calculations
- Easy to calculate short-range forces
- Molecule 1 interacts with all those which lie within a certain radius,
- Each atom interacts with just one image of the other atoms

Computer simulations of liquids; Allen and Tildesley; Chapter 1

Verlet list when PBC is used

In two dimensions,



- Avoids nearest image calculation in the inner loop of MD
- each periodic image of a particle has its own Verlet list that contains particles only those in the central box.

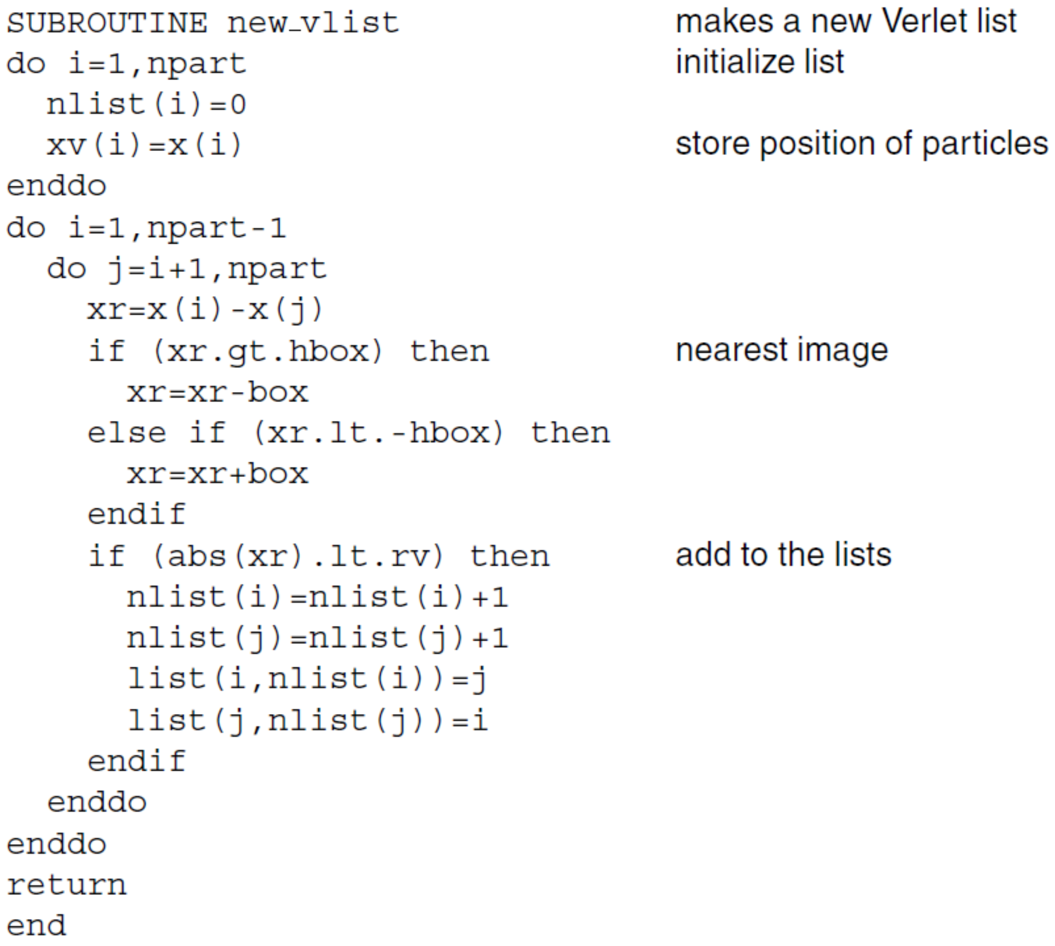
In three dimensions,

$$\mathbf{F}_i = \sum_{j=1}^{N} \sum_{k=-13}^{13} {}' \mathbf{F}_{i(j.k)} \Longleftrightarrow \mathbf{F}_i = \sum_{j=1}^{N} \sum_{k=-13}^{13} {}' \mathbf{F}_{(i.k)j}$$

$$\left.\begin{array}{l} \text{Box index } k = 9n_x + 3n_y + n_z \\ \text{Translation vector } t_k = n_x L_x + n_y L_y + n_z L_z \end{array}\right\} \text{ where } n_x/n_y/n_z = -1, 0, 1$$

Understanding Molecular Simulation, Daan Frenkel and Berend Smit, see Appendix F; An Efficient, Box Shape Independent Non-Bonded Force and Virial Algorithm for Molecular Dynamics, Bekker, et al.

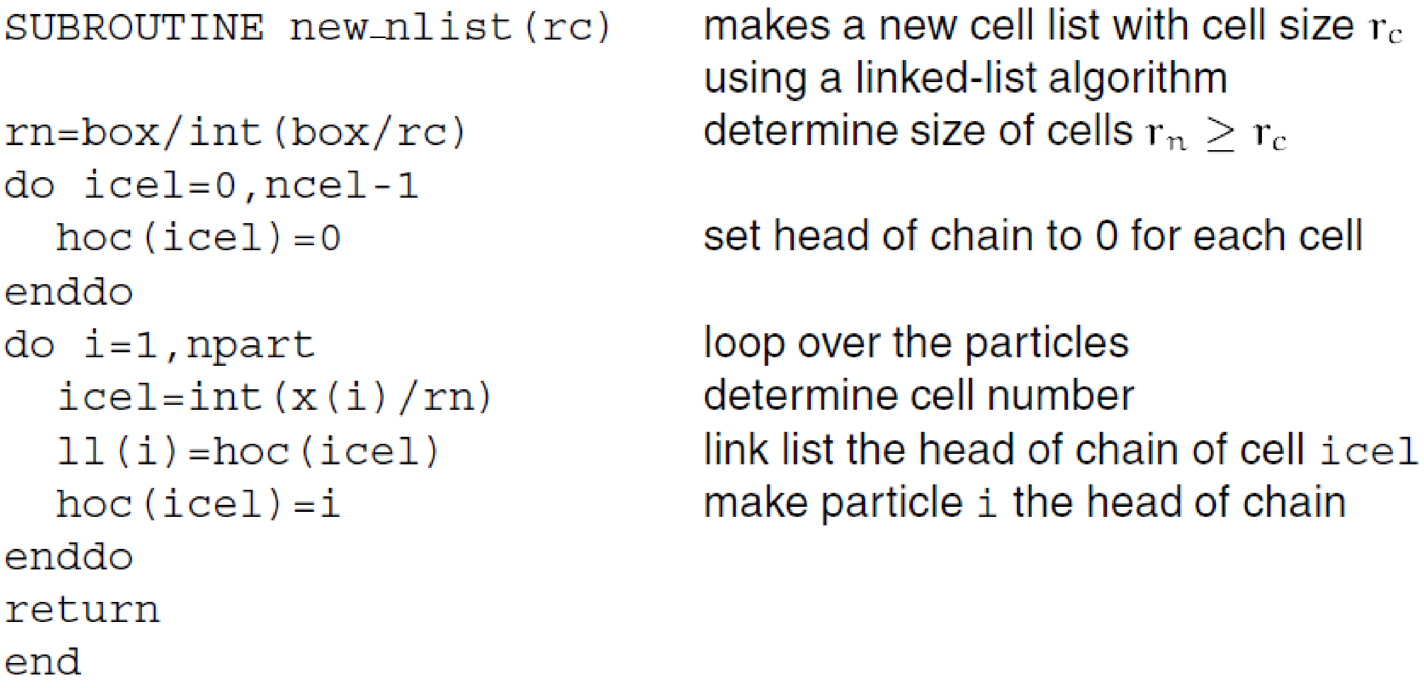
Fortran code – Verlet method



```
SUBROUTINE new_vlist                 makes a new Verlet list
do i=1,npart                         initialize list
  nlist(i)=0
  xv(i)=x(i)                         store position of particles
enddo
do i=1,npart-1
  do j=i+1,npart
    xr=x(i)-x(j)
    if (xr.gt.hbox) then             nearest image
      xr=xr-box
    else if (xr.lt.-hbox) then
      xr=xr+box
    endif
    if (abs(xr).lt.rv) then          add to the lists
      nlist(i)=nlist(i)+1
      nlist(j)=nlist(j)+1
      list(i,nlist(i))=j
      list(j,nlist(j))=i
    endif
  enddo
enddo
return
end
```

Understanding Molecular Simulation, Daan Frenkel and Berend Smit, see Appendix F

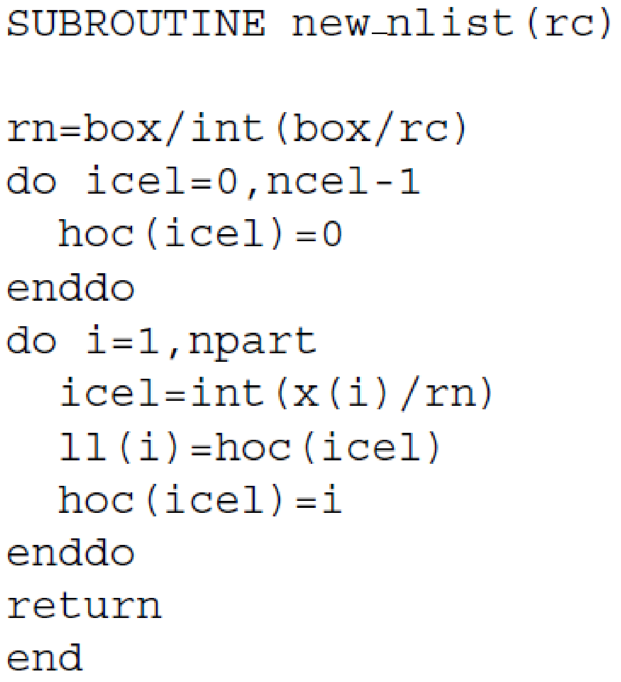
Fortran code – cell list method



```
SUBROUTINE new_nlist(rc)     makes a new cell list with cell size rc
                             using a linked-list algorithm
rn=box/int(box/rc)           determine size of cells rn ≥ rc
do icel=0,ncel-1
   hoc(icel)=0               set head of chain to 0 for each cell
enddo
do i=1,npart                 loop over the particles
   icel=int(x(i)/rn)         determine cell number
   ll(i)=hoc(icel)           link list the head of chain of cell icel
   hoc(icel)=i               make particle i the head of chain
enddo
return
end
```

Understanding Molecular Simulation, Daan Frenkel and Berend Smit, see Appendix F

Fortran code – cell list method



```
SUBROUTINE new_nlist(rc)

rn=box/int(box/rc)
do icel=0,ncel-1
  hoc(icel)=0
enddo
do i=1,npart
  icel=int(x(i)/rn)
  ll(i)=hoc(icel)
  hoc(icel)=i
enddo
return
end
```

i=1

icel = 0
ll(1) = 0
hoc(0)=1

i=2

icel = 1
ll(2) = 0
hoc(1)=2

i=3

icel = 0
ll(3) = 1
hoc(0)=3

i=4

icel = 1
ll(4) = 2
hoc(1)=4

i=5

icel = 1
ll(5) = 4
hoc(1)=5

icel =0
hoc(0)=3,ll(3)=1, ll(1)=0

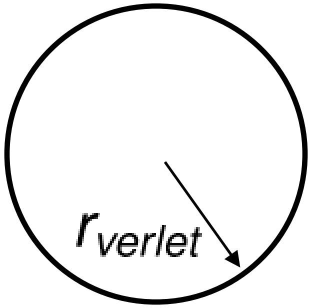
icel =1
hoc(1)=5,ll(5)=4, ll(4)=2,
ll(2)=0

Understanding Molecular Simulation, Daan Frenkel and Berend Smit, see Appendix F

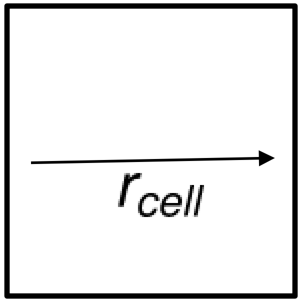
Neighbor summary

- Typical update frequency – 10 to 20 MD steps
  - Too high --> no gain in the performance of the code
  - Too low --> forces calculated incorrectly

Verlet



Cell



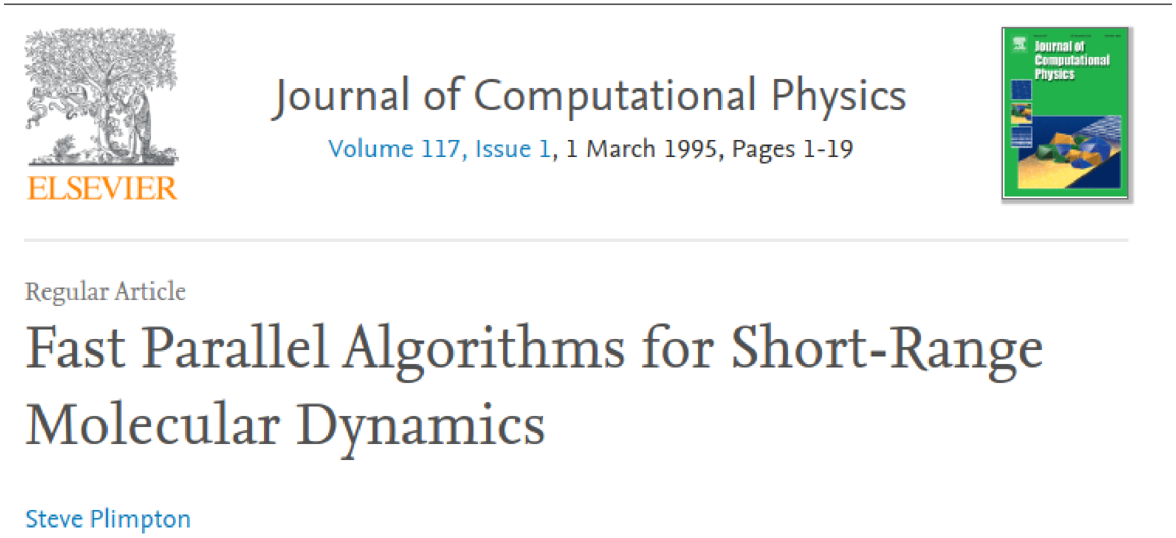
$$\frac{n_{cell}}{n_{verlet}} = 5$$

$$n_{verlet} = \frac{4}{3}\pi r_{verlet}^3 \rho$$

$$n_{cell} = 27 r_{cell}^3 \rho$$

Topic: Molecular dynamics

Parallel performance



ELSEVIER

Journal of Computational Physics

Volume 117, Issue 1, 1 March 1995, Pages 1-19

Regular Article

Fast Parallel Algorithms for Short-Range Molecular Dynamics

Steve Plimpton

https://www.sciencedirect.com/science/article/pii/S002199918571039X

Parallel MD algorithms

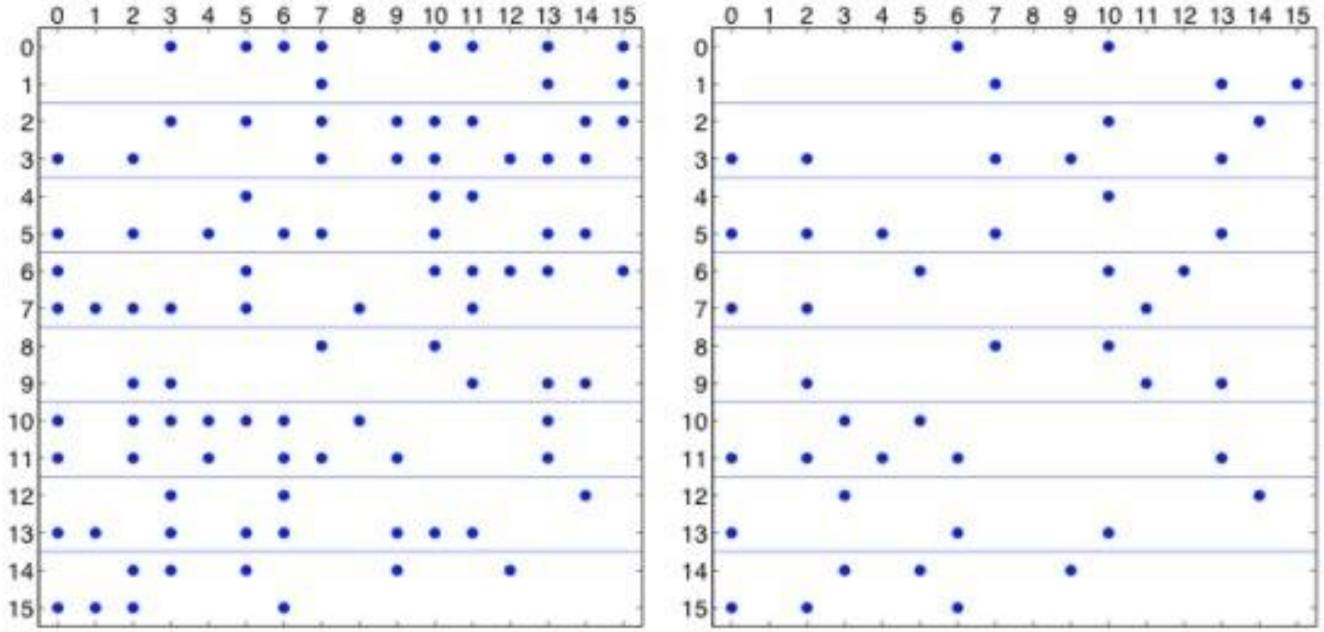
- Force calculations and positions/velocity updates
- Good algorithm ensures
  - Load balancing
  - Less communication among processes
- Assign a set of force computations to each processor and it remains fixed throughout the simulations
  - Atom decomposition
  - Force decomposition
- Assign a portion of the physical simulation domain and processes exchange atoms throughout the simulations
  - Spatial decomposition (geometric methods)

Particle decomposition

- Identical copies of atom information are stored on all processors
- Each processor (P) is assigned N/P atoms where N is total atoms
- At every MD step, the communication is needed among all processes to update the forces, etc
- Cost for communication is mainly dictated by number of messages and total volume of data (sent and received)
- Communications scales as N, independent of P ---> limits the performance for large systems
- Load balancing is maintained

Topic: Molecular dynamics

Particle (or atomic) Decomposition



(a) Force matrix.

(b) Force matrix, redundancies removed.

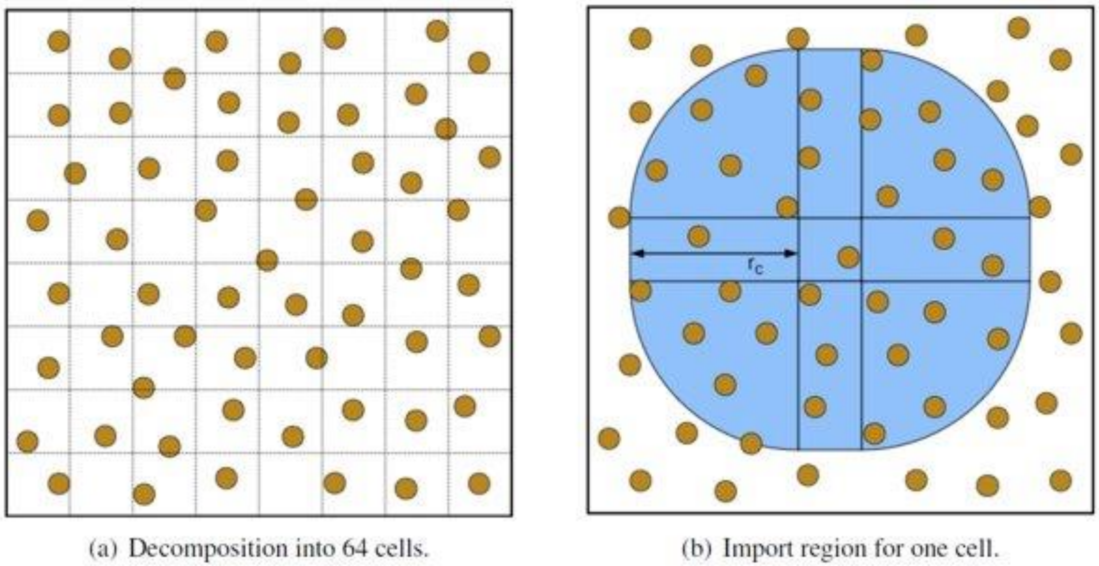
Atom decomposition: showing a force matrix of 16 particles distributed among 8 processors.

Spatial decomposition

- Space is decomposed into cells
- Each cell is assigned to a processor, and it is responsible for computing the forces on particles that lie inside this cell
- Assignment of particles to cells changes during simulation
- Communication is required between processors corresponding to nearby particles
- For high P, the import region is large compared to the number of particles contained inside each cell

Topic: Molecular dynamics

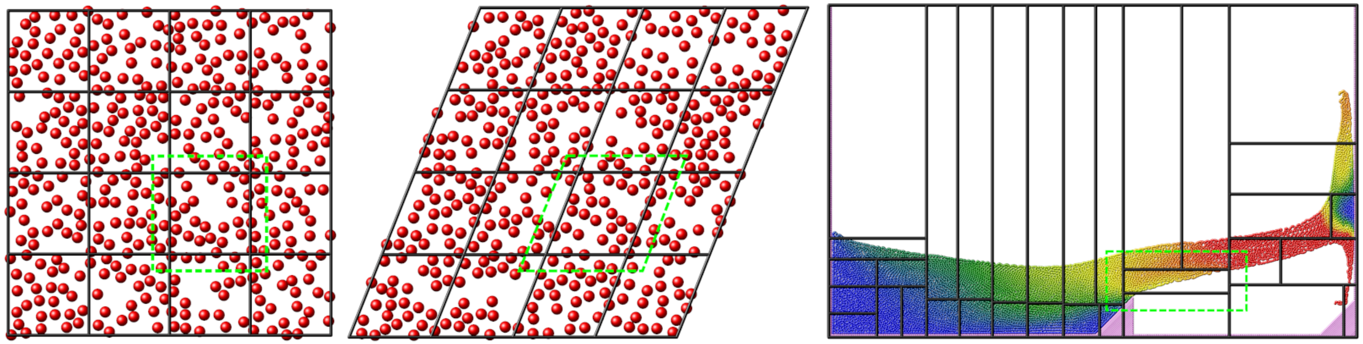
Spatial decomposition

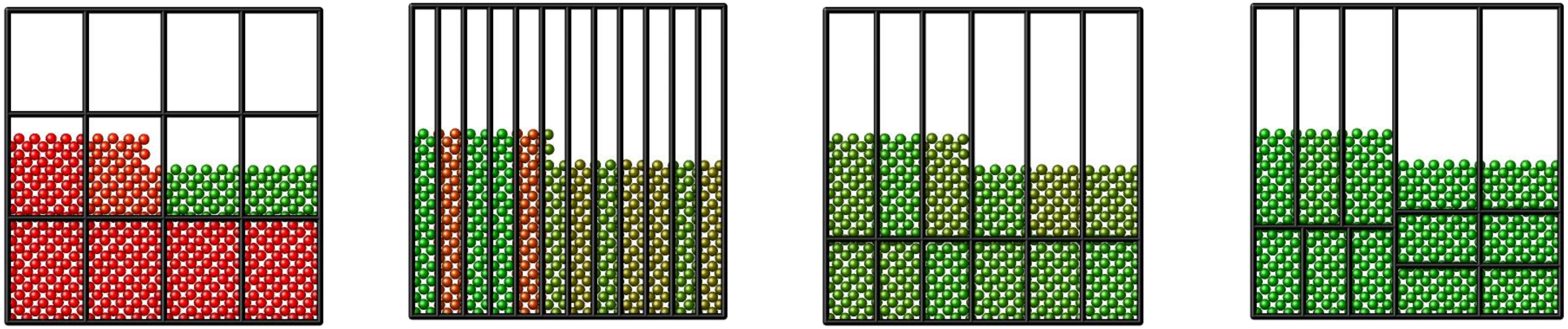

(a) Decomposition into 64 cells.
r_c
(b) Import region for one cell.

Spatial decomposition: showing particles in a 2-D computational box, partitioned into 64 cells (left), import region for one cell (right)

Topic: Molecular dynamics

Spatial decomposition





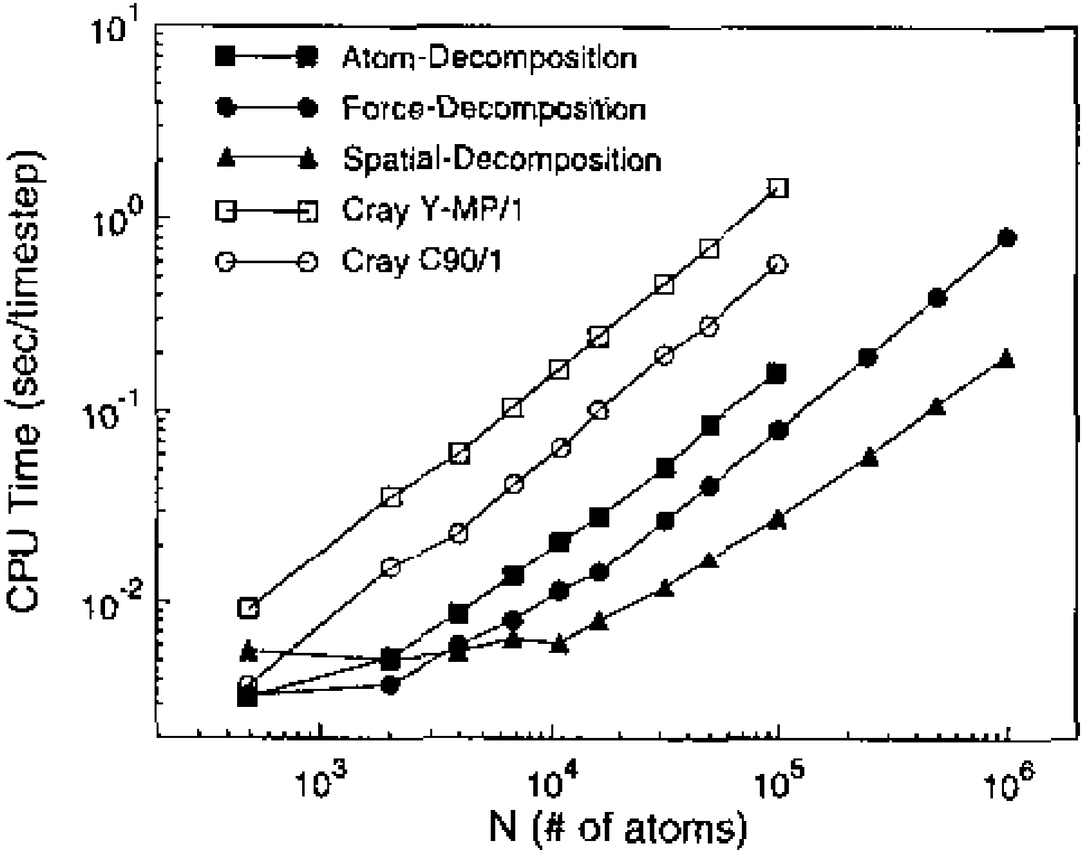
Different decompositions for a 2d system with 12 MPI ranks

Load imbalance is reduced

https://docs.lammps.org/Developer_par_part.html

Topic: Molecular dynamics

Parallel performance



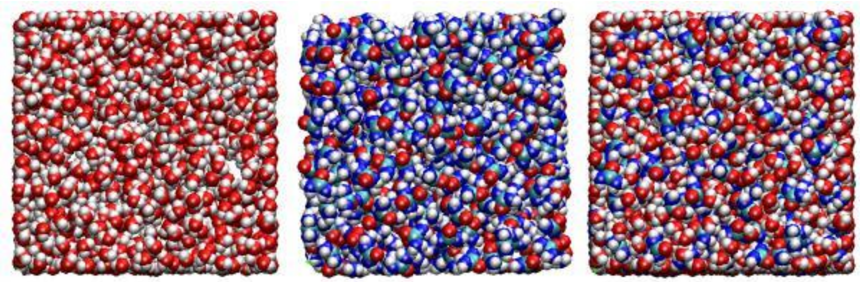
https://www.sciencedirect.com/science/article/pii/S002199918571039X

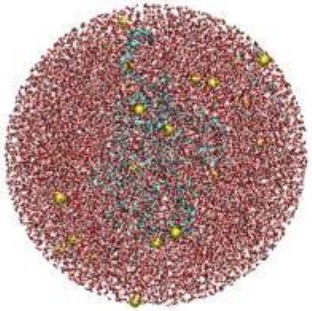
Topic: Molecular dynamics

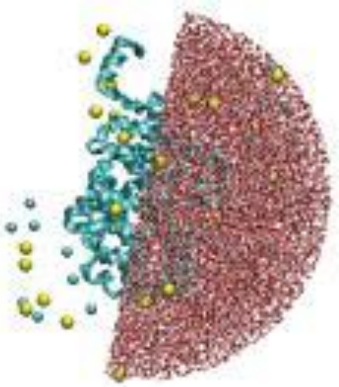
Setting up systems for molecular dynamics

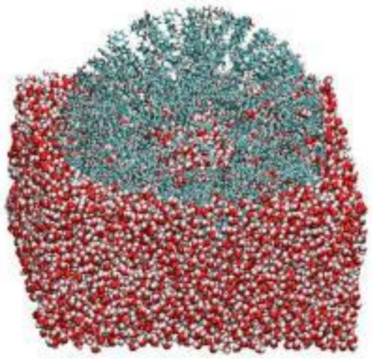
Initial configuration

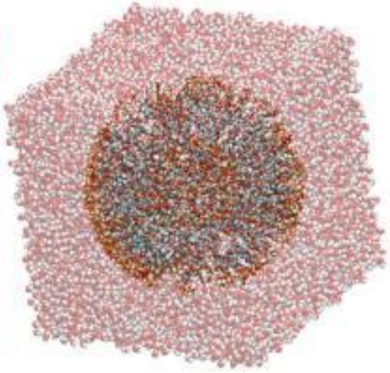
- Use packmol http://m3g.iqm.unicamp.br/packmol/











Topic: Molecular dynamics

MD softwares

- LAMMPS
- DL POLY
- NAMD
- Gromacs
- CHARMM
- AMBER

and many more

Topic: Molecular dynamics

Time scales associated various phenomena

S. No.	Internal motion	Timescale [s]
1	Protein folding	10^{-5} -10^{-1}
2	Interior-sidechain rotation (proteins)	10^{-4} -10^{0}
3	Site-juxtaposition (super helical DNA)	10^{-6} -10^{0}
4	Collective subgroup motion for e.g. For hinge bending, for allosteric transitions Global DNA bending	10^{-11} -10^{-7} 10^{-10} -10^{-7}
5	Surface-side chain rotation (Protein)	10^{-11} -10^{-10}
6	Sugar puckering (Nucleic acids)	10^{-12} -10^{-9}
7	Global DNA twisting	10^{-12}
8	Heavy atom angle bend	$5x10^{-14}$
9	Heavy atom bond stretch	$3x10^{-14}$
10	Light atom angle bend	$2x10^{-14}$
11	Double bond stretch	$2x10^{-14}$
12	Light atom bond stretch	10^{-14}

Saurabh Gupta and Pritish Kumar Varadwaj https://ijpsr.com/

Properties from MD trajectories

- Potential energy
- density
- Temperature
- Pressure

$$P = \frac{N k_B T}{V} + \frac{\sum_i^{N'} r_i \bullet f_i}{dV}$$

- Enthalpy of vaporization

$$\Delta H_{vap} = H_{gas} - H_{liquid} = U_{gas} - U_{liquid} + P(V_{gas} - V_{liquid})$$

- Isothermal compressibility

$$\kappa_T = -\frac{1}{V}\left(\frac{\partial V}{\partial P}\right)_T$$

- Thermal expansion coefficient

$$\alpha_P = -\frac{1}{V}\left(\frac{\partial V}{\partial T}\right)_P = -\frac{1}{\rho}\left(\frac{\partial \rho}{\partial T}\right)_P$$

Topic: Molecular dynamics

Properties from MD trajectories

- Self-diffusion coefficient

$$D = \frac{1}{3}\int_0^\infty \langle v_i(t) v_i(0) \rangle \, dt$$

- Static dielectric constant

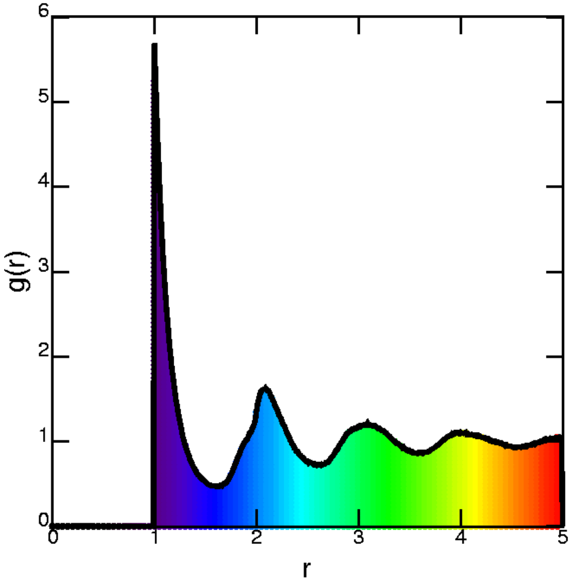
$$\epsilon = 1 + \frac{4\pi}{3Vk_B\langle T\rangle}\left(\langle \mathbf{M}^2\rangle - \langle \mathbf{M}\rangle^2\right)$$

- Surface tension

$$\gamma = \frac{1}{2}L_z\left\langle P_{zz} - \frac{1}{2}(P_{xx}+P_{yy})\right\rangle$$

- Pair correlation function or

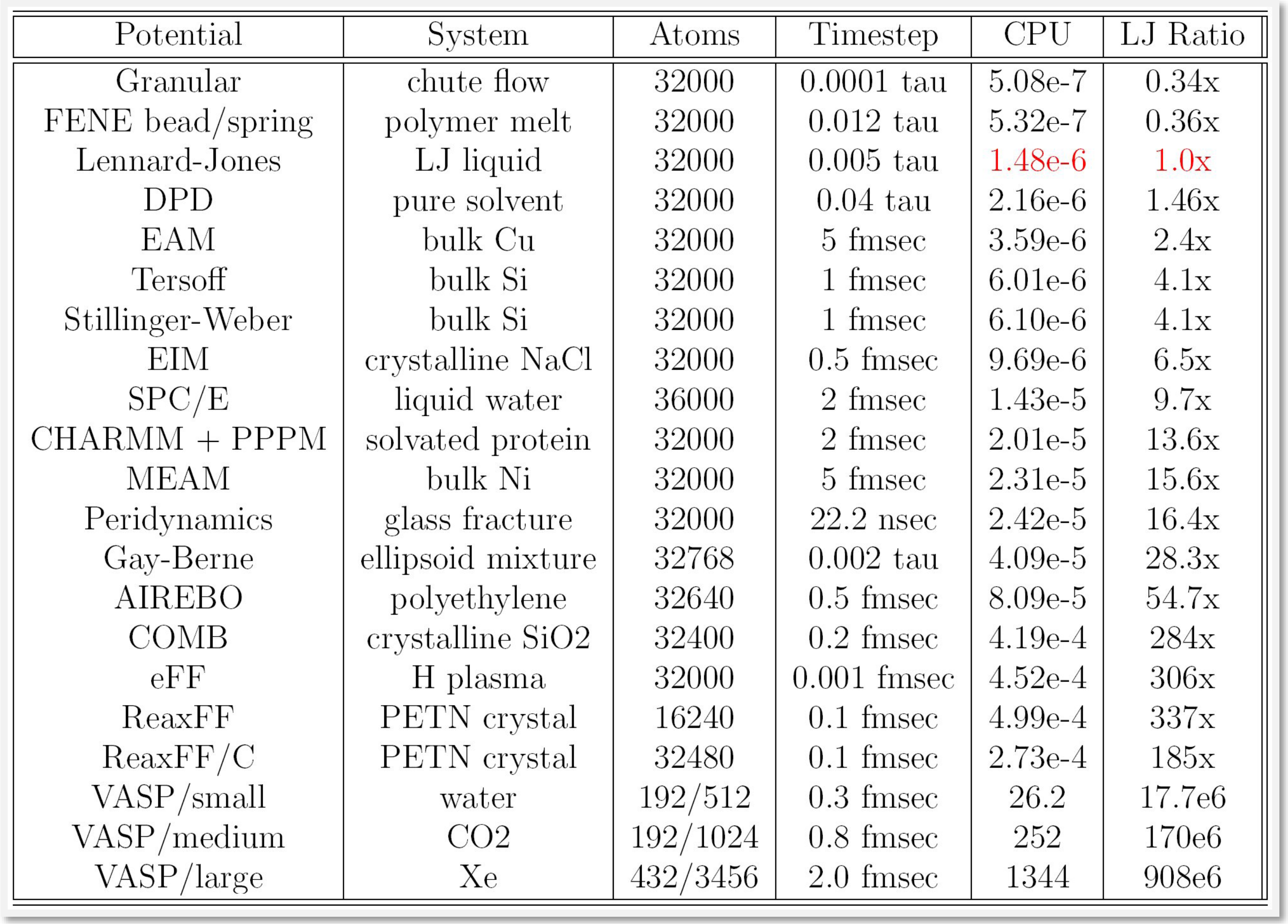
radial distribution function



http://www.physics.emory.edu/faculty/weeks//idl/gofr.html

Topic: Molecular dynamics

LAMMPS Benchmark



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

https://www.lammps.org/bench.html

Topic: Molecular dynamics

Projects

- Submit one page report along with modified code in two weeks.
- **Project 2**: Implement cell list algorithm in the given MD code and check for its performance by plotting the CPU/wall time for different system size. Also include a note on changes in wall time before and after implementing the neighbour lists.
- **Project 3**: Implement Verlet+cell list (neighbour list) algorithm in the given MD code and check for its performance by plotting the CPU/wall time for different system size. Also include a note on changes in wall time before and after implementing the neighbour lists.
- **Project 4**: Implement MPI model in the given MD code and check for its performance by plotting the parallel speedup and parallel efficiency, for different system size.

https://www.lammps.org/bench.html

Summary

- Simple algorithm
- Force/energy calculation is computationally expensive part
- Serial performance is improved using neighbor lists
- PBC and minimum image convention

Topic:

Reading material

- Understanding Molecular Simulation. From Algorithms to Applications, Daan Frenkel and Berend Smit
    - For cell and Verlet lists, read Appendix F "Saving CPU Time"
- Computer simulations of liquids; Allen and Tildesley
- Fast Parallel Algorithms for Short-Range Molecular Dynamics, Steve Plimpton
- For packmol, See http://m3g.iqm.unicamp.br/packmol/home.shtml