

Topic 2: Molecular Dynamics of Lennard-Jones System (continued)

The most time consuming part of an MD calculation is the evaluation of the accelerations of the particles at each time step. Between each pair of particles there action force and an equal and opposite reaction force. Since there are $N(N - 1)/2$ pairs of particles in the system, the time to calculate the forces scales like $\mathcal{O}(N^2)$. MD simulations are therefore natural candidates for implementation on parallel computers.

Systolic Algorithm for N particle system

In this algorithm the N particles are divided among p processes (CPU's or nodes) on the parallel system. Assume for simplicity that N is a multiple of p , e.g., if $N = 256$ and $p = 8$ each node is assigned 32 particles. The particles assigned to each node are called *local particles*. Each node stores information on the positions and velocities of its local particles. In addition, it has memory to accumulate the forces (accelerations) on each of its local particles.

In addition to its local particles, each node has memory allocated to store the positions and velocities of N/p *traveling particles*.

The p nodes are connected in a periodic ring topology. Each node communicates with its left and right neighbors in the ring. The right neighbor of the last node is the first, and the left neighbor of the first node is the last.

The problem now is to calculate the forces between all pairs of particles. This is done as follows:

- Each node calculates the forces between all of its local particles.
- Each node copies the positions and velocities of its local particles into the traveling particle memory area.
- The following steps are repeated $p - 1$ times:
 - Each node sends its traveling particle data to its right neighbor, and simultaneously receives traveling particle data from its left neighbor into its traveling particle memory area.
 - Each node calculates the forces between all of its local particles and the particles currently in its traveling particle memory area.
- Each node then applies the integration algorithm to advance the positions and velocities of its local particles by one time step.

Systole is Greek for the regular contraction of the heart which drives blood outward. A systolic algorithm rhythmically computes and passes data through a network of processors. The behavior of the algorithm is similar to the rhythmic pumping of blood by the heart in the circulatory system, or the rhythmic flow of traffic regulated by stop lights.

This algorithm scales quite well on parallel systems and is relatively straightforward to implement using MPI.

Spatial decomposition algorithm for MD

However, the most popular algorithm for parallelizing an MD calculation is based on *spatial decomposition*. In this type of algorithm, the volume of the physical system is divided into equal sub-volumes, and one sub-volume is assigned to each parallel process. For example, if we choose a periodic volume $V = L^3$, and use 8 processors, then each processor is assigned a sub-volume $(L/2)^3$.

This algorithm has advantages principally in the case of short range forces, like the Lennard-Jones interaction, which fall off rapidly with distance. If the linear size of a sub-volume is larger than the distance at which the force can be neglected, then particle in a sub-volume interact only with particles in adjacent subvolumes. This approximation can considerably reduce the amount of communication required between different processors: compare with the systolic algorithm where every particle gets to travel to every processor!

Actually, each sub-volume has a rather large number of adjacent sub-volumes: think about Rubick's Cube! So this algorithm is most useful when there are a relatively large number of processors.

Implementing this algorithm using MPI involves a considerable amount of bookkeeping because

- particles move from one sub-volume to another as the simulation proceeds,
- and interactions between particles in neighboring sub-volumes must be computed.