Parallel Biomolecular Simulation: An Overview and Analysis of Important Algorithms

Gerald Löffler and Hellfried Schreiber

University of Vienna
Institute for Theoretical Chemistry, Theoretical Biochemistry Group
Währingerstraße 17/Parterre, A-1090 Wien, Austria

1 Introduction

Molecular Dynamics (MD) delivers dynamical information for large biomolecular systems — using enormous CPU-resources. Consequently, several implementations of parallel MD programs have been presented.[1–13]

We will analyse the characteristics of MD simulations of real-live biomolecular systems, summarise the serial algorithms that have been developed to significantly speed up these simulations and take a look at the fitness of these algorithms for parallelisation.

2 Serial Algorithms and Their Impact on Parallelisation

2.1 Straightforward MD

In MD the movement of atoms is described by the equations of motion of Classical Mechanics. Most of the atoms are interconnected via strong chemical bonds, thereby forming molecules. A prototypical biomedical MD simulation deals with one big molecule (the *macromolecule*) consisting of the order of 3000 atoms immersed in a box of approx. 8000 water molecules, each of which consists of 3 atoms, which gives a total of almost 30000 moving atoms!

The principal steps are 1) to calculate the *force* $\mathbf{F}_i$ acting on each atom i due to the interaction with all other atoms and 2) to solve the equation of motion of each atom i using $\mathbf{F}_i$ to get the *position* $\mathbf{r}_i$ of the atom at the next time-step. These two steps are repeated to simulate the evolution of the system in time for as long as is necessary to answer the biomedical question at hand. Step 1 typically takes 97% of the total runtime.

$\mathbf{F}_i$ depends on $\{\mathbf{r}_1 \ldots \mathbf{r}_N\}$ (there are N atoms in the system), and can therefore be calculated in parallel with any other force calculation. Once this is done, the equation of motion can be solved for every atom i independently, because we need only $\mathbf{F}_i$ to do this.

On a *distributed-memory machine* (DM), the atomic data ($\mathbf{r}_i$, $\mathbf{F}_i$, ...) can either be distributed over the processors (*distributed data scheme* (DisD)) or replicated on every processor (*replicated data scheme* (RepD)). On a *shared-memory machine* (SM), data distribution is of course no issue.