ELSEVIER

Simulation Practice and Theory 5 (1997) 573–603

SIMULATION PRACTICE AND THEORY

Parallel biomolecular simulation: Theory, algorithms and implementation

O. Steinhauser [a,*], H. Schreiber [a,1], G. Löffler [a,2], W. Kleinert [b,3]

[a] *Institute for Theoretical Chemistry, University of Vienna, Währingerstrasse 17, A-1090 Vienna, Austria*

[b] *Computing Services, Vienna University of Technology, Wiedner Hauptstrasse 8–10, A-1040 Vienna, Austria*

Received 1 May 1996; revised 26 June 1997

Abstract

The essential goal of this work is the unified treatment of quantum mechanical and classical degrees of freedom in biomolecular simulation on all three levels: Theory, algorithms and implementation. In theory this is done within the framework of the Lagrangian model, which handles electronic coordinates and Cartesian nuclear coordinates in a consistent way. Furthermore, there is a 1:1 correspondence between the various algorithmic substeps of self-consistent-force molecular dynamics (SCF-MD) and classical molecular dynamics (CMD): The overlap criterion corresponds to the cut-off principle, the list of integrals to the pairlist and the computation of interaction matrix elements to the computation of pair forces. (The integration step is identical in both schemes.) This complete analogy can be conserved on the implementation level, for which actual benchmarks on three common architectures are given for a CMD simulation of the hydrated protein ubiquitin. © 1997 Elsevier Science B.V.

Keywords: Biomolecular simulation; Quantum and classical mechanics; Replicated data model; Benchmark

1. Preamble

1.1. Building blocks of proteins and their spatial arrangement

In principle, the building plan of proteins is quite simple [1]: They are biopolymers forming a sequential array of 20 different building blocks, the amino acids, which is called *primary structure*. From numerous experimental findings we know that the

* Corresponding author. e-mail: os@mdy.univie.ac.at.

[1]E-mail: hs@mdy.univie.ac.at.

[2]E-mail: gl@mdy.univie.ac.at.

[3]E-mail: kleinert@edvz.tuwien.ac.at.

PII S0928-4869(97)00025-6

primary structure, i.e. the amino acid sequence, determines the three-dimensional structure as well as the functionality of proteins.

The specific electronic state of the carbon atom allows a four-fold coordination by substituents. In case of the C-alpha atoms of amino acids the four substituents are: One amide group –NH, one carbonyl group –C=O, one hydrogen –H and a side chain –R. The variability of the 20 amino acids is brought about exclusively by modifications of this side chain, thereby creating hydrophobic, hydrophilic, polar and apolar features as well as size erects. Linking the carbonyl head –C=O of one amino acid to the amide tail –NH of another one linear chain molecules, polymers, are formed. While the peptide unit –(N–H)–(C=O)– is a rather rigid and planar entity, the $-(N-H)-C^{\alpha}$ and $-C^{\alpha}-(C=O)$ single bonds are characterized by a high torsional freedom, thus enabling a flexible arrangement of the chain segments. Out of this multitude of three-dimensional structures only a small family of related structures is realized under physiological conditions. It is this structural family which determines the final *folding* of a protein.

From experiment and model simulations [2] we know that the process of folding proceeds in several steps. Firstly, local *secondary structure elements* are formed, the α-helix and the β-sheet being the most prominent examples. Both partial structures are stabilized by hydrogen bonding between the amide –(N–H) and carbonyl group –(C=O) of shifted amino acid building blocks. These pre-formed secondary structure elements form a loose compound, which is subsequently strengthened by contraction. Due to this closer spatial proximity secondary structure elements interact and re-adjust, thus forming the characteristic, three-dimensional structure, the so-called *tertiary structure* already pre-determined by the primary structure, i.e. by the sequence of amino acids.

A small protein showing both types of secondary structure elements, α-helix and β-sheet [3], and thus the general structure of proteins, is ubiquitin, which serves as a model system for our current implementations. It is experimentally well characterized [4–6] and from its 76 amino acid residues only three are allowed to vary, while all others are conserved throughout the biochemical world ranging from yeast to human cells.

1.2. Dynamical structures

The mobility of atoms in proteins increases steadily from the inner core to the surface. Already in the inner hydrophobic core there is a certain amount of mobility, which on the surface reaches that of the liquid state. Therefore a description of the atomic positions in terms of fixed coordinates is only possible in the solid state, if at all. Under physiological conditions a characterization of protein structure by a unique set of coordinates is not adequate, but one has to consider a *family of structures*, whose members transform into each other quite rapidly. In this sense one speaks of dynamical structures.

1.3. Effect of hydration

Proteins exhibit their intrinsic structure and function in a native environment only, i.e. when a pronounced hydration shell is there. Experiments on model proteins

have shown that there exist extreme cases where secondary structure elements, like α-helix or β-sheet transform into each other, when changing the conditions of hydration.

1.4. Experiment and biomolecular simulation

Through the advanced techniques of genetic engineering the primary structure of proteins can be manipulated routinely. Even site-specific modifications, i.e. mutations, can be introduced without problems. This almost unlimited flexibility in preparing proteins, however, contrasts with the enormous experimental effort in structure determination. Structural experiments on proteins have to face three essential problems:

- They are expensive and costly.
- The obtained information is far from complete.
- Data analysis is a difficult procedure.

In all three cases *Biomolecular Simulation* can offer assistance:

- By eliminating unpromising systems, thus limiting the number of experiments to the rewarding ones.
- By interpolating and completing experimental data, thus providing a consistent picture.
- By decomposing the complex experimental information into individual contributions, thus facilitating data analysis.

2. Theory

2.1. Lagrangian Model (LM)

In principle, *quantum mechanics* (QM) is the appropriate theory for molecules and particularly biomolecules. It states that the classical concept is no longer valid for elementary particles. In other words the familiar *orbits* have to be replaced by probability tubes, so-called *orbitals* $\varphi_i(\vec{r})$. These orbitals $\{\varphi_i(\vec{r})\}$ may be considered as vectors in *Hilbert space* and represented by linear combinations of basis functions $\{\chi_k(\vec{r})\}$:

$$\varphi_i(\vec{r}, t)=\sum_k c_k^i(t)\chi_k(\vec{r}) \tag{1}$$

Since electrons and protons are indistinguishable within their species, each orbital contributes with equal weight to the one-particle density $\rho(\vec{r})$:

$$\rho(\vec{r}, t)=\sum_i \varphi_i^*(\vec{r}, t)\varphi_i(\vec{r}, t)=\sum_k \sum_l D_{kl}(t)\chi_k^*(\vec{r})\chi_l^*(\vec{r}) \tag{2}$$

Thereby we have introduced the density matrix

$$D_{kl}(t)=\sum_i c_k^{i*}(t)c_l^i(t) \tag{3}$$

again referring to the basis expansion (1).

If one is interested in the simultaneous description of several states of a system, one has to solve the equation of motion for the density matrix, the so-called Liouville–von Neumann equation [7]. For biomolecules, however, such a treatment is not feasible in practice. Therefore one would be highly content to cope with the system's ground state. In contrast to the excited states, the so-called *adiabatic approximation*, being fairly valid for biomolecules, enables a Lagrangian formalism [8–11] which subjects the difference of the kinetic and potential energy, the so-called Lagrange function

$$L=T_{\text{kin}}-U \tag{4}$$

$$T_{\text{kin}}=\tfrac{1}{2}\sum_i m_i \dot{c}_k^{i2} \tag{5}$$

$$U=U(\{c_k^i\}) \tag{6}$$

to an extremum principle; the general result of which are the equations of motions:

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{c}_k^i}=\frac{\partial L}{\partial c_k^i} \tag{7}$$

Upon inserting the special expressions (Eqs. (4)–(6)), one obtains a system of coupled equations of motion

$$m_i \ddot{c}_k^i=-\frac{\partial U}{\partial c_k^i}=F_k^i \tag{8}$$

describing the time evolution of the coefficients or *coordinates* $\{c_k^i(t)\}$. According to Eq. (1) they determine the weighing of the basis functions and thus the migration of the respective particles through space. While in early times hydrogen-like and similar functions have been used as a basis, there is now a long tradition in using Cartesian Gaussians located at the nuclei. However, spherical Gaussians centered at all relevant points in space enable a more intuitive and vivid representation:

$$g_k(\vec{r})=\left(\frac{\eta}{\sqrt{\pi}}\right)^3 \mathrm{e}^{-\eta(\vec{r}-\vec{r}_k)^2} \tag{9}$$

Each of them may be viewed as a *grain* in probability space, i.e. as a small probability volume whose extension is determined by the exponent η. It is up to the weighing coefficients $\{c_k^i\}$ to decide, which grains are favorably occupied and which one disadvantaged. As the weighing changes in time due to the equations of motion (Eq. (8)), one gets the impression of a migrating wave packet.

In the short-time regime evolution of the coordinates, electronic or nuclear, is

described by a Taylor series expansion:

$$c_k^i(t\pm\Delta t)=c_k^i(t)\pm\dot{c}_k^i(t)\Delta t+\tfrac{1}{2}\mu^{-1}F_k^i(t)\Delta t^2 \tag{10}$$

where we have inserted the equations of motion for the second time derivative $\{\ddot{c}_k^i\}$ (cf. Eq. (8)). Forming the sum and difference we get:

$$c_k^i(t+\Delta t)+c_k^i(t-\Delta t)=2c_k^i(t)+\mu^{-1}F_k^i(t)\Delta t^2 \tag{11}$$

$$c_k^i(t+\Delta t)-c_k^i(t-\Delta t)=2\dot{c}_k^i(t)\Delta t \tag{12}$$

Upon rearranging the first relation one gets a simple integration algorithm

$$c_k^i(t+\Delta t)=2c_k^i(t)+\mu^{-1}F_k^i(t)\Delta t^2-c_k^i(t-\Delta t) \tag{13}$$

that promotes the coordinates to the future $(t+\Delta t)$ thereby using the present (time $=t$) coordinates and forces, as well as the coordinates of the past (time $=t-\Delta t$). As compared to the large variety of integration algorithms to solve second order differential equations in numerical mathematics, the simple Verlet algorithm (Eq. (13)) has the advantage of needing force computation only once. As this is the overwhelming computational problem in biomolecular simulation Verlet's algorithm [12] is widely used.

From the difference relation one infers

$$\dot{c}_k^i(t)=\frac{c_k^i(t+\Delta t)-c_k^i(t-\Delta t)}{2\Delta t} \tag{14}$$

thus giving the velocity of the particles and hence the kinetic energy, which in its turn may be related to the temperature T of the system upon averaging over all particles and time

$$\langle T_{\mathrm{kin}}\rangle=\tfrac{1}{2}\left\langle\sum_{i=1}^{n}m_i\dot{c}_k^{i^2}\right\rangle=\tfrac{3}{2}nk_{\mathrm{B}}T \tag{15}$$

with k_{B} being Boltzmann's constant.

It is important to note that in actual simulations a fictitious electronic mass μ much larger than the actual is used. Values up to one tenth of the nuclear mass are quite common. This is done in order to speed up the integration scheme and does not really harm the Lagrangian model, which is mainly intended to explore phase space and not to generate the true trajectory of the electrons.

2.2. *Ab-initio molecular dynamics (AMD)*

2.2.1. *Classical limit for nuclei*

Although the Lagrangian model enables a uniform description of electrons and nuclei, it does not account for the heavier mass of the nuclei, which exceeds that of the electrons by a factor of 2000–20000. As a consequence nuclei are fairly localized in contrast to the spread out probability of electrons. Therefore a representation by a linear combination (Eq. (1)) out of a set of Gaussians covering the entire molecular

volume would be highly inefficient, because most coefficients are close to zero. Of course, the assumption of exactly localized nuclei is the most efficient. Viewed from the basis set this would mean to shrink the half-width of the Gaussians to zero such that they degenerate to delta-functions:

$$\chi_k(\vec{r}) = \lim_{\eta \to 0} g_k(\vec{r}) = \delta(\vec{r} - \vec{r}_k) \tag{16}$$

In this limit nuclei become classical particles whose position is described by a triple of Cartesian coordinates:

$$c_k^I \Rightarrow \vec{R}_I = (X_I,\ Y_I,\ Z_I) \tag{17}$$

This simplifies the equations of motion for the nuclei

$$m_I \ddot{\vec{R}}_I = -\frac{\partial U}{\partial \vec{R}_I} = \vec{F}_I, \quad I = 1, \ldots, N \tag{18}$$

while the time evolution of the electrons is still described in terms of the original coordinates $\{c_k^i\}$ in Hilbert space

$$\mu \ddot{c}_k^i = -\frac{\partial U}{\partial c_k^i} = F_k^i, \quad i = 1, \ldots, n \tag{19}$$

with μ being their uniform mass. Needless to say the simple Verlet algorithm depicted in Section 2.1 is not specific to electronic coordinates, i.e. it applies to the Cartesian nuclear coordinates $\vec{R}_I = (X_I,\ Y_I,\ Z_I)$ likewise.

2.2.2. *Kohn–Hohenberg theory (KHT)*

According to the first theorem of Kohn and Hohenberg the total energy U, whose derivatives determine the equations of motion (Eqs. (18) and (19)), must be a functional of the density [13,14]:

$$U = U_{\mathrm{NN}} + U_{\mathrm{Ne}} + T_{\mathrm{ee}} + U_{\mathrm{ee}} + U_{\mathrm{xc}} + U_{\mathrm{orth}} \tag{20}$$

$$U_{\mathrm{NN}} = \sum_{I=1}^{N-1} \sum_{J=I+1}^{N} \frac{Z_I Z_J}{|\vec{R}_I - \vec{R}_J|} \tag{21}$$

$$U_{\mathrm{Ne}} = -\sum_{I=1}^{N} \int \frac{Z_I \rho(\vec{r})}{|\vec{R}_I - \vec{r}|}\, \mathrm{d}\vec{r} \tag{22}$$

$$T_{\mathrm{ee}} = \sum_{i=1}^{n} \int \varphi_i^*(\vec{r}) \left(-\tfrac{1}{2}\Delta \right) \varphi_i(\vec{r})\, \mathrm{d}\vec{r} \tag{23}$$

$$U_{\mathrm{ee}} = \tfrac{1}{2} \int \frac{\rho(\vec{r})\rho(\vec{r}')}{|\vec{r} - \vec{r}'|}\, \mathrm{d}\vec{r}\, \mathrm{d}\vec{r}' \tag{24}$$

$$U_{\mathrm{xc}} = \int \epsilon_{\mathrm{xc}}(\rho(\vec{r}))\, \mathrm{d}\vec{r} \tag{25}$$

$$U_{\text{orth}} = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \epsilon_{ij} \int \varphi_i^*(\vec{r}) \varphi_j(\vec{r}) \, \mathrm{d}\vec{r} \tag{26}$$

While the first four contributions, the nuclear repulsion U_{NN}, the electron–nuclear attraction U_{Ne}, the quantum mechanical kinetic energy of the electrons T_{ee} and the Coulomb repulsion of the electrons U_{ee}, have a self-explaining meaning and the last term U_{orth} is merely a constraint potential, ensuring the orthogonality of the orbitals, U_{xc} describes the exchange (x) and correlation (c) effects characteristic for a quantum mechanical many-body system.

Until recently, these effects had to be computed by complicated quantum mechanical procedures, which restricted their application to less than 10 atoms. During last years, however, improved density functionals become available and practically manageable. They are generalizations of the local exchange approximation

$$U_x = -\alpha \int (\rho(\vec{r}))^{4/3} \, \mathrm{d}\vec{r} \tag{27}$$

already derived by Dirac in 1930 [15] and parametrized by Slater in 1951 [16], and involve not only the density $\rho(\vec{r})$, but also its gradient, thereby accounting for the deviations from the uniform electron gas. The major breakthrough in this respect is associated with the names of Becke [17] (exchange) and Lee, Yang and Parr [18] (correlation), which enlarged the applicability of density functional theory by an order of magnitude.

2.2.3. Born–Oppenheimer model (BO): moving nuclei only

When studying the dynamical evolution of biopolymers around their folded structure, the biomolecule usually stays in the electronic ground state and transitions to the excited states have a very low probability. Under these circumstances the so-called Born–Oppenheimer approximation [19] is fairly valid telling us that the electron density $\rho(\vec{r})$ reacts instantaneously to changes in the nuclear geometry thereby minimizing the potential energy U with respect to the electronic coordinates. In mathematical term this means that the $\{c_k^i\}$ are determined by the minimum condition

$$F_k^i = \frac{\partial U}{\partial c_k^i} = 0, \quad i = 1, \ldots, n \tag{28}$$

which states that the forces acting on the electrons vanish. However, this has important consequences on the nuclear forces thus simplifying their computation. Quite generally, there an explicit dependence of U upon the nuclear coordinates and an implicit one via the electronic coordinates, since the electronic problem is now solved for a fixed nuclear geometry thus inferring a parametric dependence of the $\{c_k^i\}$ on the $\{\vec{R}_I\}$:

$$\vec{F}_I = -\frac{\partial U}{\partial \vec{R}_I} - \frac{\partial U}{\partial c_k^i} \frac{\partial c_k^i}{\vec{R}_I} + \text{Pulay forces.} \tag{29}$$

At this point a caveat is necessary. As far as the basis set covers completely

molecular space and is localized at fixed positions, nuclear forces are indeed determined by the first two terms. For reasons of economy, however, basis functions are usually localized at the nuclei. Thus they are coupled to nuclear motion thus introducing a slight implicit dependence, the so-called Pulay forces [20], which enter as a third term.

2.3. Self-consistent-force molecular dynamics (SCF-MD): computing the electron density

Assuming an ideal basis, i.e. neglecting for the moment the Pulay forces, and bearing in mind that the minimum condition (Eq. (28)) eliminates the second term, we are solely left with the explicit R-dependence of the first term, which is present in the nuclear repulsion U_{NN} and the electron–nuclear attraction U_{Ne} only:

$$\vec{F}_I = -\frac{\partial V_{SCF}}{\partial \vec{R}_I} \tag{30}$$

$$V_{SCF} = \sum_{I=1}^{N-1} \sum_{J=I+1}^{N} \frac{Z_I Z_J}{|\vec{R}_I - \vec{R}_J|} - \sum_{I=1}^{N} \int \frac{Z_I \rho(\vec{r})}{|\vec{R}_I - \vec{r}|} \, d\vec{r} \tag{31}$$

This result is known as the Hellmann–Feynman or electrostatic theorem [21] and states that the forces acting on the nuclei are of pure electrostatic origin [22].

Explicit evaluation of the minimum conditions (Eq. (28)) leads to an algebraic matrix equation [23] for the electronic coordinates, collected in the coefficient matrix $C_{ij} = c_j^i$

$$HC = \epsilon SC \tag{32}$$

The matrix elements of the Hamiltonian matrix H and the overlap matrix S are given as integral over the basis functions $\{\chi_k(\vec{r})\}$:

$$S_{ij} = \int \chi_i^*(\vec{r}) \chi_j(\vec{r}) \, d\vec{r} \tag{33}$$

$$H_{ij} = \int \chi_i^*(\vec{r}) \left[-\tfrac{1}{2}\Delta + v_{eff}(\rho(\vec{r})) \right] \chi_j(\vec{r}) \, d\vec{r} \tag{34}$$

with the effective potential

$$v_{eff}(\rho(\vec{r})) = \frac{\delta U}{\delta \rho} \tag{35}$$

being the functional derivative of the potential energy with respect to the electron density $\rho(\vec{r})$, which in its turn depends on the electronic coordinates via the density matrix (Eq. (3)). This problem is overcome by an iteration procedure. Initial estimates for the electronic coordinates are inserted into the Hamiltonian matrix elements and the eigenvalue problem (Eq. (32)) is solved by diagonalization. These

new coordinates are again inserted and the whole procedure is repeated until self-consistency has been achieved.

Once the coefficients $\{c_k^i\}$ and hence the density $\rho(\vec{r})$ have converged to self-consistency the Hellmann–Feynman forces (Eq. (30)) can be computed and the nuclear coordinates promoted:

$$\vec{R}_I(t+\Delta t)=2\vec{R}_I(t)+\mu^{-1}\vec{F}_I(t)\Delta t^2-\vec{R}_I(t-\Delta t) \tag{36}$$

Then the Hamiltonian matrix elements are evaluated again and the matrix eigenvalue problem is solved iteratively to obtain self-consistency. Step by step this combination of solving the eigenvalue problem and promoting the nuclear coordinates is carried on.

2.3.1. Gaussian representation

Inserting the basis expansion (Eq. (3)) of the electronic density into the Hellmann–Feynman formulae (Eqs. (30) and (31)) one gets

$$V_{\mathrm{SCF}}=\sum_{I=1}^{N-1}\sum_{J=I+1}^{N}\frac{Z_I Z_J}{|\vec{R}_I-\vec{R}_J|}-\sum_{I=1}^{N}\sum_{\{kl\}}Z_I D_{kl}\int\frac{\chi_k^*(\vec{r})\chi_l(\vec{r})}{|\vec{R}_I-\vec{r}|}\,\mathrm{d}\vec{r} \tag{37}$$

A basis set of spherical Gaussians has the property that the product of two Gaussians can be converted to a single Gaussian

$$\chi_k^*(\vec{r})\chi_l(\vec{r})=\chi_{kl}(\vec{r}-\vec{r}_{kl}) \tag{38}$$

with a modified exponent

$$\eta_{kl}=\eta_k+\eta_l \tag{39}$$

and being localized at point $\vec{r}_{kl}$ between the centers of the two partners

$$\vec{r}_{kl}=\frac{\eta_k\vec{r}_k+\eta_l\vec{r}_l}{\eta_{kl}} \tag{40}$$

Exploiting this special relation the Hellmann–Feynman potential (Eq. (31)) takes the simple form

$$V_{\mathrm{SCF}}=\sum_{I=1}^{N-1}\sum_{J=I+1}^{N}\frac{Z_I Z_J}{|\vec{R}_I-\vec{R}_J|}-\sum_{I=1}^{N}\sum_{\{kl\}}Z_I D_{kl}\frac{\mathrm{erf}(\eta_{kl}|\vec{R}_I-\vec{r}_{kl}|)}{|\vec{R}_I-\vec{r}_{kl}|} \tag{41}$$

Thus the density matrix elements may be viewed as electronic Coulomb charges interacting with the nuclear charges, while their spatial distribution is accounted for by the error-function $\mathrm{erf}(\eta_{kl}r)$. Unfortunately, there is a practical obstacle to this appealing picture: The electronic charge elements D_{kl} depend on the nuclear configuration

$$D_{kl}=D_{kl}(\vec{R}_1,\ \ldots,\ \vec{R}_N) \tag{42}$$

i.e. they have to be recomputed — by solving the eigenvalue problem (Eq. (32)) — once the nuclear coordinates have been promoted in time by some increment Δt.

This limits the application to roughly 100 atoms and an elapsed time of a few ps. Therefore only a small portion of a protein, maybe the active site or the co-factor, can be treated in this way. The overwhelming part must still be handled by classical mechanics as described in the next section. Currently, hybrid simulations treating a small region by SCF-MD and the remainder by CMD are becoming tractable [24–26]. Nevertheless, the key problem, how to match the coupling of the two regions in SCF-MD/CMD simulations, seems to have not found a satisfactory solution yet.

2.4. Classical molecular dynamics (CMD)

In order to cope with 10000 and more atoms over at least a few nanoseconds (ns) the classical molecular dynamics (CMD) method replaces the quantum mechanical potential function V_{SCF} (Eq. (41)) by a chemically intuitive model potential V_{CMD} whose respective contributions

$$V_{\mathrm{CMD}} = V_{\mathrm{Coulomb}} + V_{\mathrm{shape}} + V_{\mathrm{bonded}} + V_{\mathrm{constraint}} \tag{43}$$

are calibrated for a broad class of molecules or molecular building blocks [27–32]. In order to include experimental structural information, like dihedral angle constraints or inter-residue proton–proton distances derived from NMR, V_{CMD} is usually augmented by a constraint potential [33].

The various contributions to V_{CMD} are modelled in the following way:

2.4.1. Coulomb forces

First, the electronic charges D_{kl} are confined to constant values representative for the equilibrium nuclear coordinates. Subsequently, they are fused with the nuclear charges $\{Z_I\}$ to give some effective charges $\{q_i\}$:

$$V_{\mathrm{Coulomb}} = \sum_{I=1}^{N} \sum_{J=I+1}^{N} \frac{(q_I + \delta q_i^I)(q_J + \delta q_j^J)}{R_{IJ}} \tag{44}$$

In the next step the permanent charges are augmented by so-called flickering charges located at fixed positions, but of variable charge strength $\{\delta q_i^I\}$. (The simplest choice may be a octahedral or tetrahedral charge arrangement around each atom [34].) The flickering charges are the solution of the set of equations

$$\delta q_i^I = \sum_i T_{ij}^I V_j^I(\{q_i^I\}, \{\delta q_i^I\}) \tag{45}$$

Here, T_{ij}^I is a coupling tensor describing the fixed geometrical arrangement of the flickering charges as well as the specific polarizability of each atom, i.e. its ability to deviate from the permanent charge, both with respect to strength as well as spatial distribution. V_j^I is the electrostatic potential originating from the complete set of permanent and flickering charges. Consequently, a self-consistent procedure is carried out analogous to the quantum mechanical computation of the electronic density as described in Section 2.3. In other words, the model of flickering charges is a

mimicry of its quantum mechanical original. So far the flickering charge model has been applied scarcely ever, as it needs a re-design of the complete V_{CMD} potential.

2.4.2. Shape forces

Of course the charge model, maybe permanent or flickering, is strictly valid in the asymptotic regime only, i.e. for large interatomic distances. In the overlap region, i.e. at close distances, the so-called Pauling forces representing the penetration of the atomic charge clouds are very important. From theory one knows that the Pauling forces [35] are repulsive and show an exponential dependence on interatomic distances. Additionally, there is an attractive contribution, which comes from the correlated motion of the electrons, the so-called dispersion interaction. Within the model of polarizable dipoles dispersion is found to depend on the inverse sixth power of the interatomic distance. Combining both contributions leads to the Buckingham potential

$$v_{\mathrm{Buck}} = A_{IJ}\, \mathrm{e}^{-B_{IJ}R_{IJ}} - \frac{1}{R_{IJ}^{-6}} \tag{46}$$

Although the Buckingham potential is theoretically better founded, most actual simulations prefer the Lennard-Jones (LJ) potential

$$V_{\mathrm{shape}} = \sum_{I=1}^{N} \sum_{J=I+1}^{N} v_{\mathrm{LJ}}(R_{IJ}) \tag{47}$$

$$v_{\mathrm{LJ}} = 4\epsilon_{IJ}\left\{\left(\frac{\sigma_{IJ}}{R_{IJ}}\right)^{12} - \left(\frac{\sigma_{IJ}}{R_{IJ}}\right)^{6}\right\} \tag{48}$$

which replaces the exponential term by an inverse 12th power law, thus saving one additional parameter and allowing a better transferability of parameters.

The shape forces V_{shape}, describing the space filling or packing of atoms in the molecule, couple all pairs of atoms, at least in principle. Their strong decay $(1/r_{ij}^6)$ with respect to the interatomic distance r_{ij}, however, permits the following distance criterion

$$v_{\mathrm{LJ}}(R_{IJ}) = 0, \quad \text{if } R_{IJ} \geq r_{\mathrm{c}} \tag{49}$$

beyond which interactions ma be neglected for numerical reasons. This *cut-off principle* enables a very efficient evaluation of the shape forces.

2.4.3. Bonded forces

Both Coulomb and shape forces do not account for the stereospecificity of biomolecular interactions. In order to include this essential feature one needs terms that mimic the connectivity through the molecule, i.e. the chemical bonds

$$V_b = \tfrac{1}{2} \sum_{\mathrm{bonds}} k_b(b - b_0)^2 \tag{50}$$

angular terms

$$V_\theta = \tfrac{1}{2} \sum_{\text{angles}} k_\theta (\theta - \theta_0)^2 \tag{51}$$

describing the direction of the bonds and dihedral terms

$$V_\phi = \tfrac{1}{2} \sum_{\text{dihedrals}} k_\phi (1 + \cos(n\phi + \delta) \tag{52}$$

that hinder free rotation about bonds by introducing rotational barriers. In case of specific local arrangements of atoms — like planar rings, peptide bonds etc. — 'dihedral springs'

$$V_\xi = \tfrac{1}{2} \sum_{\text{imp.dih}} k_\xi (\xi - \xi_0)^2 \tag{53}$$

are added.

Altogether, the total bonded potential U_{bonded} is made up of these four terms

$$V_{\text{bonded}} = V_b + V_\theta + V_\phi + V_\xi \tag{54}$$

and comprises all interactions tieing together the set of atoms to form a molecule, i.e. constituting the molecular topology. These contributions of bonds (2 atoms), bond angles (3 atoms) and dihedral angles (4 atoms) are strictly local and involve 4 atoms at maximum.

2.4.4. The exceptional role of Coulomb forces

In some sense bonded and shape forces $V_{\text{bonded}} + V_{\text{shape}}$ and Coulomb forces V_{Coulomb} are antagonists. The former are essential for the biochemical properties of a molecule and exhibit a rather complicated functional form. Once this form and the necessary parameters have been calibrated, however, their computation can be done with moderate computational effort. Coulomb forces, on the other hand, have a very simple form, but their long range — $1/r_{ij}$ is the force law of longest range in nature — prohibits the application of a cut-off principle thus leading to a huge number of interacting Coulomb pairs. These long range couplings are also responsible for biomolecular recognition at large intermolecular distances. Thereby the solvent acts as transmitting medium. Only at close contact, shape forces come into action and play the dominant role.

According to the exceptional role of the Coulomb forces we shall exclusively concentrate on these forces within the framework of the classical molecular dynamics (CMD) method. All other forces may be computed by analogous techniques (shape forces) or make a negligible contribution to the numerical effort (bonded forces).

2.5. Simulated annealing

Irrespective of the treatment of the interaction function, quantum mechanically (SCF-MD) or classically (CMD), the number of local minima in the potential function increases tremendously with the size of the molecule. Therefore local

minimizers, like gradient or Newton–Raphson techniques, which can find the local minimum close to some starting geometry, are unable to cope with the *rugged energy landscape* of biomolecules.

A way out of this dilemma is the method of *Simulated Annealing* [36]. It is known from Statistical Mechanics that the accessible volume of phase space is a monotonous function of temperature T. Thus one starts with an unphysically high T of several thousand degrees to give the biomolecule a chance to cover a volume of phase space as large as possible, thereby using the high kinetic energy to cross barriers. In a so-called *temperature protocol* T is systematically lowered, i.e. the system is annealed, thus confining the biomolecule to regions of favorable potential energy. Under best circumstances the global minimum can be approached fairly close.

In technical terms lowering of temperature is brought about by a uniform scaling of all velocities with a parameter

$$\lambda = \sqrt{1 + \frac{\Delta t}{\tau}\left(\frac{T_0}{T(t)} - 1\right)} \tag{55}$$

that measures the deviation of the actual temperature $T(t)$ — calculated from the velocities according to Eq. (27) — from the value T_0 of the protocol. If $T(t)$ is above the desired value T_0, $\lambda < 1$ and velocities are scaled down. This is not done within one time step, but during a relaxation time τ.

As we have uniformly applied the Verlet algorithm to quantum and classical degrees of freedom, the procedure of velocity scaling is quite unique:

$$\text{electrons: } c_k^i(t+\Delta t) = \left[2c_k^i(t) + \mu^{-1}F_k^i(t)\Delta t^2 - \left(2 - \frac{1}{\lambda}\right)c_k^i(t-\Delta t)\right]\lambda \tag{56}$$

$$\text{nuclei: } \vec{R}_I(t+\Delta t) = \left[2\vec{R}_I(t) + \mu^{-1}\vec{F}_I(t)\Delta t^2 - \left(2 - \frac{1}{\lambda}\right)\vec{R}_I(t-\Delta t)\right]\lambda \tag{57}$$

If the actual temperature $T(t)$ equals the desired value, i.e. $T(t) = T_0$, the scaling parameter λ is unity and the Verlet algorithm remains unchanged.

3. Algorithms

3.1. Classical molecular dynamics (CMD): Ewald transformation

This is the algorithmic counterpart of the theory Section 2.4 on classical molecular dynamics (CMD). While using simple, chemically intuitive interaction functions, this method is applicable to systems of 10000 and more atoms. This is the reason why we treat this method prior to the more rigorous, quantum mechanical SCF-MD method described in Section 2.3, which at present can manage typically 100 atoms.

In atomistic solvation the huge number of water molecules in the hydration shell — typically several thousand — plays the major part in force computation. Since water molecules contribute very little to the computation of bonded and shape

forces, the already high weight of the Coulomb forces is even more exaggerated and the evaluation of Coulomb sums becomes the essential problem.

Coulomb sums are conditionally convergent series [37], thus rendering application of the cut-off principle (Eq. (49)) impossible. In order to enforce absolute convergence it is advisable to perform the so-called Ewald transformation [38], which splits the direct Coulomb potential $1/r$ into the interaction of two Gaussian charge distributions and the respective complement:

$$\frac{1}{r} = \frac{\mathrm{erf}(\eta r)}{r} + \frac{\mathrm{erfc}(\eta r)}{r}. \tag{58}$$

Simultaneously, one considers the lattice consisting of the replica of the simulation cell and computes not only the interaction of a reference charge with its partner, but includes all periodic images, too (periodic boundary conditions (cf. p. 155 of [39])). Depending on their convergence rate, however, the error function term and its complement are summed in different lattices. The complement $\mathrm{erfc}(\eta r)/r$ is summed over the real lattice, while the summation of Gaussian interaction $\mathrm{erf}(\eta r)/r$ is shifted to reciprocal space. Since η is an arbitrary parameter in the splitting of the Coulomb potential (Eq. (58)), the sum of real and reciprocal contributions must be independent of the actual value of η. For reasons becoming evident shortly, η is usually chosen high enough to ensure that the real space part falls off quite rapidly thus confining the real space sum to the original simulation cell. For such high η values the Ewald potential takes the special form [40]:

$$\Phi_{\mathrm{EW}}(\vec{r}) = \sum_{\vec{k}} A(k)\cos(\vec{k}\cdot\vec{r}) + \frac{\mathrm{erfc}(\eta r)}{r} \tag{59}$$

Correspondingly, the total Coulomb interaction U_{EW} of an ensemble of charges $\{q_i\}$ is given by

$$U_{\mathrm{EW}} = \sum_{I=1}^{N-1} \sum_{J=I+1}^{N} (q_I + \delta q_i^I)\cdot(q_J + \delta q_j^J)\Phi_{\mathrm{EW}}(\vec{R}_I - \vec{R}_J) \tag{60}$$

(cf. the corresponding expression (Eq. (44)) without Ewald transformation). Due to the splitting of the Ewald potential into k- and r-sums

$$U_{\mathrm{EW}} = U_{\mathrm{EW}}^{k} + U_{\mathrm{EW}}^{r} \tag{61}$$

one follows completely different strategies in computing both contributions.

3.1.1. *Summing in k-space*

Using the trigonometric identity

$$\cos(\vec{k}\cdot(\vec{r}_i - \vec{r}_j)) = \cos(\vec{k}\cdot\vec{r}_i)\cos(\vec{k}\cdot\vec{r}_j) + \sin(\vec{k}\cdot\vec{r}_i)\sin(\vec{k}\cdot\vec{r}_j) \tag{62}$$

the k-part of the double sum (Eq. (60)) can be converted to the square of two single

sums [40]:

$$U_{\mathrm{EW}}^{k}=\frac{1}{2}\sum_{k} A(k)\left\{\left[\sum_{i=1}^{N} q_i \cos(\vec{k}\cdot\vec{r}_i)\right]^2+\left[\sum_{i=1}^{N} q_i \sin(\vec{k}\cdot\vec{r}_i)\right]^2\right\} \tag{63}$$

Thus one obtains two independent loops referring to the k-vectors and to the number of charges, respectively. Fortunately, the latter loop is the longer one thus enabling an optimal exploitation of modern RISC processors. On the other hand, the outer looping over k-vectors is ideally suited for data partitioning on parallel architectures giving excellent load-balancing. *Of course, on the level of implementation, a data parallel strategy in connection with global summation and communication is best for summing in k-space.*

3.1.2. Neighborlists in r-space

Due to the strong damping of the complementary error function $\mathrm{erfc}(\eta r)$ potential interaction partners of a reference atom are actually confined to a sphere of radius r_c. Therefore one can gain a lot of numerical efficiency by eliminating all charge pairs separated by a distance greater than r_c. In symbolic code this elimination process appears as a double loop:

```
DO I=1,N-1
XI=X(I)
YI=Y(I)
ZI=Z(I)
QI=Q(I)
DO J=I+1,N
DX=XI-X(J)
DY=YI-Y(J)
DZ=ZI-Z(J)
R2=DX*DX+DY*DY+DZ*DZ
RC2=RC*RC
IN=INT(R2/RC2)
IF(IN.EQ.0) THEN
R=SQRT(R2)
U^r_EW = U^r_EW + QI*Q(J)*erfc(η*R)
END IF
END DO
END DO
```

However, a direct computation of this kind is disadvantageous: One computes a huge number of charge pair distances, which are subsequently discarded by the cut-off criterion as represented by the IF-statement.

At this point a little physical thinking helps a lot (cf. p. 147 of [39]). The migration of interaction partners into or out of the cut-off sphere of a reference atom is continuous process changing slowly with time. Therefore it is reasonable to freeze the set of interaction partners at least for a few time steps, i.e. to sort all interacting charge pairs with respect to the cut-off principle and to collect all pairs close than

r_c in a list, the so-called neighbor list. Quite efficiently, pair forces are calculated exclusively out of this neighborlist, which is kept constant over a couple of time steps. Changes in neighborhood relations are accounted for by a rhythmic updating of the list.

A symbolic double loop for creating the neighbor list is given below:

```
NNB=0
DO I=1,N-1
XI=X(I)
YI=Y(I)
ZI=Z(I)
DO J=I+1,N
DX=XI-X(J)
DY=YI-Y(J)
DZ=ZI-Z(J)
R2=DX*DX+DY*DY+DZ*DZ
RC2=RC*RC
IN=INT(R2/RC2)
IF(IN.EQ.0) THEN
INB(NNB+1)=I
JNB(NNB+1)=J
NNB=NNB+1
END IF
END DO
END DO
```

3.1.3. Stair case algorithm

This form of a triangular loop is highly inefficient for parallelization, because the computational work load for the inner loop is not constant when partitioning the outer loop. The *stair case* algorithm circumvents this problem by loop transformation [41]. For an odd number of particles the inner loop is carried out for $(N-1)/2$ elements, starting from the first off-diagonal element. Once the end of a row is reached, the corresponding overflow is corrected by subtracting the total number of particles N. For an even number N. $(N-1)/2=(N/2)-1/2$ is no longer an integer. In this case one half of the rows is carried out with $(N/2)$, while the other half uses $(N/2)+1$. In the following loop structure this is managed by the index IKEY:

```
IKEY=(N/2)*2-N+1
NNB=0
DO I=1,N
JEND=(1-IKEY)*((N-1)/2)+IKEY*((N/2)-1)*IKEY*(I/((N/2)+1))
XI=X(I)
YI=Y(I)
ZI=Z(I)
DO JJ=1,JEND
J=JJ-((JJ+I)/(N+1))*N+I
DX=XI-X(J)
```

```
DY=YI-Y(J)
DZ=ZI-Z(J)
R2=DX*DX+DY*DY+DZ*DZ
RC2=RC*RC
IN=INT(R2/RC2)
IF(IN.EQ.0) THEN
INB(NNB+1)=I
JNB(NNB+1)=J
NNB=NNB+1
END IF
END DO
END DO
```

3.1.4. Computing the pair forces

Once the neighbor list has been created, it can be used for several time steps — typically ten — to compute the r-part of the Coulomb sum as well as the pair forces

$$\vec{F}_{ij} = -\frac{\partial}{\partial \vec{r}_{ij}} \frac{q_i q_j \operatorname{erfc}(\eta r_{ij})}{r_{ij}} = f(\eta r_{ij}) \cdot \vec{r}_{ij} \tag{64}$$

by looping over the total number NNB of interacting pairs i,j. Due to the principle of *actio=reactio*, however, each pair force $\vec{F}_{ij}$ contributes to the net force of both interaction partners with opposite sign:

$$\vec{F}_i = \vec{F}_i + \vec{F}_{ij} \tag{65}$$

$$\vec{F}_j = \vec{F}_j - \vec{F}_{ij} \tag{66}$$

In symbolic code we thus have:

```
U_EW^r=0.0
DO I=1,N
FX(I)=0.0
FY(I)=0.0
FZ(I)=0.0
END DO
DO II=1,NNB
I=INB(II)
J=JNB(II)
DX=X(I)-X(J)
DY=Y(I)-Y(J)
DZ=Z(I)-Z(J)
RIJ2=DX*DX+DY*DY+DZ*DZ
RIJ=SQRT(RIJ2)
QIJ=QI*QJ
ETARIJ=ETA*RIJ
U_EW^r=U_EW^r+QIJ*ERFC(ETARIJ)
FETARIJ=F(ETARIJ)
```

```
FXIJDX=FETARIJ*DX
FYIJDY= FETARIJ*DY
FZIJDZ=FETARIJ*DZ
C
CACTIO=REACTIO
C
FX(I)=FX(I)+FXIJDX
FY(I)=FY(I)+FYIJDY
FZ(I)=FZ(I)+FZIJDZ
FX(J)=FX(J)-FXIJDX
FY(J)=FY(J)-FYIJDY
FZ(J)=FZ(J)-FZIJDZ
END DO
```

In this final form the computation of pair forces consists of a single big loop, optimal for both, parallel and super scalar architectures.

Needless to say, the neighborlist and stair case algorithms may be applied to any short-range interaction in real space. Thus the Lennard-Jones potential (Eqs. (47) and (48)) and the real space part of the Ewald potential can be treated in a uniform way.

3.2. Self-consistent-force molecular dynamics (SCF-MD)

This is the algorithmic counterpart of the theory Section 2.3 on SCF-MD. The major goal of this section is to work out the perfect analogy to classical molecular dynamics (CMD) as treated in the previous section. Thus identical implementation strategies can be followed for SCF-MD and CMD.

3.2.1. Evaluation of integrals

The elements of the Hamiltonian matrix (Eq. (34)) are made up by contributions of kinetic and potential energy:

$$H_{ij} = T_{ij} + V_{ij} \tag{67}$$

While the matrix elements of kinetic energy

$$T_{ij} = \int \chi_i^*(\vec{r}) \left[-\tfrac{1}{2}\Delta \right] \chi_j(\vec{r})\, \mathrm{d}\vec{r} \tag{68}$$

are given by simple analytic expressions, the functional dependence of the effective potential on the electron density prohibits direct analytic integration.

$$V_{ij} = \int \chi_i^*(\vec{r}) [v_{\mathrm{eff}}(\rho(\vec{r}))] \chi_j(\vec{r})\, \mathrm{d}\vec{r} \tag{69}$$

In the literature two strategies have evolved. On the one hand, efficient numerical integration is performed over a grid [17]. This contributes a loop over the number of grid points in addition to the double loop over basis functions. As both, the

number of grid points and the number of basis functions NB, are roughly proportional to the size of the respective molecule, evaluation of integrals is a NB^3-algorithm. This becomes immediately evident for the second method, which expands the effective potential with respect to the basis functions [42],

$$v_{\text{eff}}(\rho(\vec{r})) = \sum_k v_k \chi_k(\vec{r}) \tag{70}$$

thus transforming the v-elements to a sum over analytic three-fold integrals

$$V_{ij} = \sum_k v_k \int \chi_i^*(\vec{r}) \chi_k(\vec{r}) \chi_j(\vec{r}) \, \mathrm{d}\vec{r} \tag{71}$$

3.2.2. Overlap criterion

With increasing size of molecules the number of pairs of basis functions, whose overlap is practically negligible because of the large spatial separation, is even more increasing. In other words, the corresponding overlap integral

$$S_{ij} = \int \chi_i^*(\vec{r}) \chi_j(\vec{r}) \, \mathrm{d}\vec{r} < S_c \tag{72}$$

is below a certain threshold S_c [43], which in its turn justifies the neglection of the matrix elements V_{ij}:

```
DO I=1,NB
DO J=I,NB
V_IJ=0.0
S_ij=∫χ_i*(r)χ_j(r) dr
IF(S_ij≥S_c) THEN
V_ij=∫χ_i*(r)[v_eff(ρ(r))]χ_j(r) dr
END IF
END DO
END DO
```

Comparing this double loop with the corresponding one in CMD (see symbolic code following Eq. (63)) complete analogy is found on algorithmic grounds. Strictly speaking, there is a slight difference, as in quantum mechanics the inner loop starts at the diagonal element V_{ii}, while the classical loop is shifted by one element, because there are no self-forces $\vec{F}_{ii}$. However, this is merely a counting problem and does not harm the analogy as such. Therefore one can exploit this obvious analogy by creating a list of integrals, which contains only those pairs $\{i, j\}$ of basis functions, whose overlap integral is sufficiently large. A symbolic double for creating a list of integrals using the overlap criterion is the following:

```
NNB=0
DO I=1,NB
DO J=I,NB
S_ij=∫χ_i*(r)χ_j(r) dr
```

```
IN=INT(SC/SIJ)
IF(IN.EQ.0) THEN
INB(NNB+1)=J
JNB(NNB+1)=J
NNB=NNB+1
END IF
END DO
END DO
```

3.2.3. Stair case algorithm

Proceeding in analogy to the neighborlist algorithm this triangular loop is transformed again using the stair case algorithm in order to achieve optimal parallelization:

```
IKEY=(NB/2)*2 NB+1
NNB=0
DO I=1,NB
JEND=(1 IKEY)*((NB 1)/2)+IKEY*((NB/2) 1)*IKEY
*(I/((NB/2)+1))
DO JJ=0,JEND
J=JJ-((JJ+I)/(NB+1))*NB+I
```

$$S_{ij} = \int \chi_i^*(\vec{r}\chi_j(\vec{r})\,\mathrm{d}\vec{r}$$

```
IN=INT(SIJ/SC)
IF(IN.EQ.0) THEN
INB(NNB+1)=I
JNB(NNB+1)=J
NNB=NNB+1
END IF
END DO
END DO
```

3.2.4. Computing the matrix elements V_{ij}

Once the list of integrals has been set up, during all iteration cycles of the SCF–procedure the matrix elements V_{ij} can be computed by looping over the total number NNB of interacting pairs i, j of basis functions χ_i, χ_j:

```
DO II=1,NNB
I=INB(II)
J=JNB(II)
```

$$V_{ij} = \int \chi_i^*(\vec{r})[v_{\mathrm{eff}}(\rho(\vec{r}))]\chi_j(\vec{r})\,\mathrm{d}\vec{r}$$

```
END DO
```

As with the pair forces in CMD we have now a single big loop, optimal for both, parallel and super scalar architectures.

3.2.5. Diagonalization

Not to forget diagonalization is a major step during the SCF procedure. However, it makes no sense to develop any general algorithm, because diagonalization is so ubiquitous in numerical mathematics that there exist library routines for every computer architecture.

4. Parallel implementation

In recent years several strategies in parallel macromolecular simulation e.g. [44–47] have emerged with special emphasis on large processor numbers. Centers for Massively Parallel Computers, however, are and will be rare in the scientific world. Thus we have in mind the usual equipment of a typical biomolecular research group, which works with RISC workstations and profits from the continuously raising price–performance ratio of this technology. Consequently, we follow a pragmatic route [41] in developing parallel algorithms.

Of course, much more important than our group philosophy are the findings of the international scientific community. Quite recently, at the second international conference on Macromolecular Modelling organized by Prof. Deuflhard at ZIB of FU Berlin, where many experts in parallelization participated, there was general agreement that spatial decomposition for the real space part of the Coulomb forces becomes superior to data parallel algorithms for more than 32 processors only. This number exceeds by far the typical application we had and still have in mind with our algorithms and implementations.

4.1. Workload partitioning

Fig. 1 is a graphical representation of the basic algorithms applied in classical molecular dynamics (CMD). Replacing the terms PAIRLIST by LIST of INTEGRALS and FORCES by INTERACTION MATRIX ELEMENTS this flow-chart applies to SCF-MD likewise. The substep INTEGRATION is common to both methods.

Apart from data initialization, performed at the beginning once and for all, each simulation step or cycle in CMD consists of three (four) algorithmic substeps, depending on whether the pairlist is updated or not:

(1) creation of the pairlist (optional);
(2) computation of pair forces;
(3) integration of the equations of motion;
(4) fulfillment of constraints, i.e. fixing bond lengths at their experimental values.

Viewed exclusively from the aspect of workload partitioning three of these four steps lend themselves to highly efficient parallelization. Due to the stair-case transformation the workload carried out by the inner loop is fairly equal such that only the outer has to be partitioned. This benefit propagates to the pair force computation, where again the big loop over interacting pairs — constituting up to 80% of the total elapsed computer time — has to be cut into portions of equal size. The third

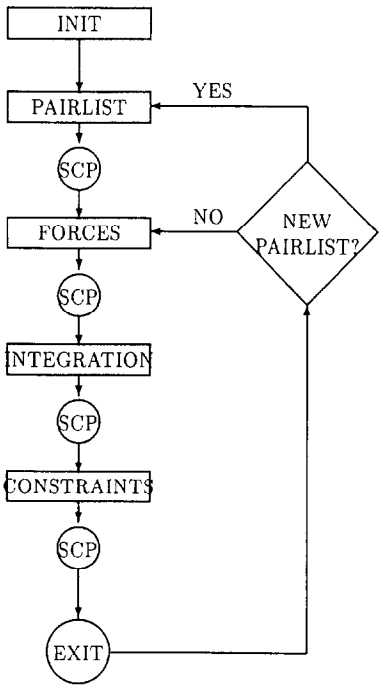

Fig. 1.

substep, integration, is parallel per se, provided the net force on each atom is available. Integration time, however, is usually only a small fraction of the complete CMD-cycle. Therefore, the gain in parallelization is usually overruled by additional overhead like communication between processors etc. Thus it makes sense to carry out the complete integration loop repeatedly on each processor and to avoid partitioning at all.

Within a molecule traditional constraint algorithms are strictly sequential, i.e. they loop through a stream of selected bonds, thereby instantaneously updating atomic coordinates. For proteins, however, this is not a real obstacle. On the one hand, one may exploit the specific topology of these chain molecules, where subsequent building blocks are linked by a single bond. Thus the constraint cycle in its turn may be broken up into a series of sub-substeps similar to the flow diagram of the general CMD-cycle [48]. A second and even more strong argument concerns the hydration of proteins. Due to the huge number of water molecules they dominate constraint time, thus reducing the protein's contribution to a rather small amount. The stream of water constraints, on the other hand, consists of very small independent, unlinked building blocks and can thus be parallelized with high efficiency. In addition we have developed a direct solution to the constraint force problem in water avoiding the cumbersome iterative solution of coupled equations [49].

4.2. Data flow and data distribution

Partitioning of the workload is only one part of parallelization, usually the easier one. Mostly, the other part, data management, is more complicated. Referring again to the symbolic flow chart of the CMD-cycle, one realizes a specific data flow: Each substep can start if and only if the data set computed in the previous one is complete. This complete input–complete output scheme is preconditioned by physics and not brought about by the algorithmic design: We have to know the complete list of interacting pairs in order to partition the workload in pair force computation. Due to the principle of actio and reactio, which states that each pair force contributes with opposite sign to the net force on the interacting atoms, integration can proceed if and only if all pair forces have been computed, because even the last portion of the pairlist entries may still affect net forces in an unpredictable manner. The constraint substep in its turn needs all integrated, i.e. promoted coordinates, corrects them and handles them over to the pairlist or force step.

According to this physically pre-conditioned data flow we have adopted replicated data model, where each processor has access to the complete data set during every substep. For a shared-memory architecture, this can be easily realized by common areas in memory. Strictly speaking, one has to care for semaphores that manage read and write conflicts to the global arrays.

For distributed architectures, however, a more evolved data management is necessary: At the end of each substep partial results have to be collected from the individual processors and subsequently the net result has to be broadcasted. In a way this causes synchronization of all processors being symbolized in Fig. 1 by synchronization check points (SCP). With respect to the four substeps the following is meant by *collecting the net result*:

(1) Concatenation, i.e. joining the partial pairlists, created on each processor, in order to have a complete, contiguous pairlist.

(2) Global summation of the pair forces, generated on individual processors, to find the net force on each atom.

(3) Concatenation of atomic coordinates promoted by the integration algorithm.

(4) In addition to the concatenation of atomic coordinates information concerning the linkage of building blocks has to be exchanged.

As a result each synchronization check point (SCP) consists of two communication substeps:

(1) global summation or concatenation;

(2) subsequent broadcasting.

4.3. Portability

Our group has now a eight years tradition in *parallel computation*. During the pilot phase, covering the first two years, we concentrated on the parallelization of the most time consuming substeps creation of the pairlist and computation of the pair forces (cf. steps 1 and 2 in the flow-chart). This work was started out of the simulation package GROMOS being wide-spread in the academic world. In this

way we produced a mixed code consisting of traditional sequential parts of the program PROMD out of the GROMOS package [50] and self-written parallel routines, running on a low-cost transputer network. Although there existed severe limitations in the system's hard- and software, we finally succeeded in simulating hydrated peptides. This work is documented in a series of publications [41,51–53].

With the appearance of more professional architectures we decided to develop a completely new program package, called SPHINX (simulation of proteins by Hamiltonian integration using networks of cross-linked (x) computers). It was essential to this new programming concept to combine inherently parallel coding with maximum portability. On the level of program code we have realized this concept by uniting data partitioning and communication structure in a program core, which in its turn is surrounded by a shell of subprograms of traditional, sequential design. In designing this shell routines optimum single-node performance was a major goal. When calling this shell routines from the program core, loop bounds are handed over that ensure optimal balance of the workload.

As far as the pure computational tasks are concerned, the program architecture described so far guarantees a portable code running on most parallel architectures. For shared-memory systems this is more or less sufficient. For distributed-memory architectures, however, communication has to be kept portable, too. Therefore each synchronization check point (SCP) was realized by the two communication paradigms *global summation or concatenation and broadcasting*. When changing the hardware platform these two communication routines have to be changed only.

As far as efficiency is concerned, there is no doubt that generic communication routine is to be preferred. If one is interested in maximum portability, however, one may use the public domain software *parallel virtual machine* PVM [54], which is based on the *UNIX sockets*, and treats a collection of computing nodes as distributed-memory parallel computer. However, PVM provides only *communication primitives*, i.e. SEND and RECEIVE commands between processors. Therefore we have assembled communication routines for global summation and broadcasting using this primitives as constituting elements.

In the mean time PVM has become a quasi-standard on distributed-memory machines. (Release 3.x supports even shared-memory architectures.) Thus our simulation package SPHINX has now achieved maximum portability.

As opposed to the pipeline architecture of a workstation cluster our communication scheme is ideally supported by a hypercube architecture. Here, global summation and broadcasting merge, because summation steps can be performed in parallel, i.e. simultaneously, such that each processor holds the net sum thereby avoiding explicit broadcasting. For this reason the hypercube implementation was based on the generic communication routines for global summation [55] in order to gain maximum efficiency.

4.4. Architectures

While developing the program package SPHINX we had permanent access to three parcel architectures

(1) An *Intel iPSC*860 Hypercube
(2) A workstation cluster of 4 × HP-720 and 2 × SGI-Indy machines
(3) A workstation cluster of 4 × DEC-alpha machines.

4.4.1. Hypercube architecture

This parallel computer is based on the Intel i860-processor. In our implementation it consists of 16 nodes equipped with 8MB memory each. The network true hypercube architecture and there exists a library of generic communication routines. All nodes share a concurrent file system. A specific ethernet node also part of the system and enables an efficient access to data via FTP-protocol.

4.4.2. HP-SGI workstation cluster

This cluster consists of 6 workstations, 4 HP Apollo 720 GRX and 2 Silicon Graphics (SGI) machines of the type INDIGO or INDY. For our SPHINX code the partitioning of the computational workload was uniform, because there was only a marginal difference in the CPU speed of the SGI R4000 processor and the HP-720.

While PVM provides the software basis for communication, conventional ethernet links are used in hardware. In order to protect the cluster from the usual network traffic, we installed a dedicated subnet. A SUN-IPC workstation equipped with two ethernet cards served as a gateway. Outside connection managed by a gate daemon. This protection or screening was of great merit in running parallel jobs.

4.4.3. DEC-alpha workstation cluster

As discussed in the subsequent section on benchmarks ethernet links are somewhat prohibitive to efficient parallelization on workstation clusters. Cooperation with a research group at Salzburg University, however, enabled access to a cluster consisting of four DEC-alpha workstations connected by a glass fibre network. In this way we were able to learn more about the communication time behavior of our SPHINX code.

4.5. Current work

We are currently augmenting our SPHINX package by a quantum mechanics module in order to enable an adequate treatment of co-factors in proteins. Inevitably this will change our programming paradigm from a homogeneous to a heterogeneous one, i.e. some processors will work on the quantum mechanical part SCF-MD, while the others do classical work CMD. Nevertheless, the algorithms for parallelization will be the same for both parts due to the complete analogy between SCF-MD and CMD, as discussed in the algorithms section. Therefore one can expect that the final gain from parallelization will be very similar in both cases. In other words the following section on benchmarks, which is restricted to CMD, may give some impression on the performance of quantum mechanical SCF-MD simulations as well.

5. Application and benchmarks

Although our quantum mechanical SCF-MD module will be finished during the next months, we are not yet in a position to give precise benchmark results. Therefore this section is entirely devoted to classical CMD simulations. Nevertheless, the complete analogy between SCF-MD and CMD should already give some impression of the quantum benchmark results for SCF-MD.

When studying the dynamical structure of proteins the realistic, i.e. fully atomistic, description of the hydration shell is essential for our investigations. Even if the dimensions of this shell are kept at minimum size, typically four to five water diameters, one has to simulate explicitly several thousand water molecules. On the other hand, the numerical effort to compute the pair forces, consuming typically 85% of the total computer time per CMD-cycle, is a linear algorithm, which can be coped with almost perfectly by parallelization.

As a test system we have chosen the protein ubiquitin already mentioned in the preamble. The corresponding hydration shell consisted of 4200 water molecules in accordance with the technical limitations of the hypercube and the workstation clusters. For the computation of pair forces we applied a twin-range cut-off principle. Interacting pairs separated by a distance closer than 0.9 nm were collected in a pairlist, which was kept constant for 10 time steps and then updated. In addition to this core-list force contributions from interacting neighbors in the spherical shell 0.9 nm to 1.2 nm were evaluated and also kept constant for 10 time steps. The k-space sum was limited to 40 terms.

Although the integration of the equations of motion (3rd substep in CMD-cycle) and constraints (4th substep) are open to parallelization, we have computed them sequentially. As the corresponding computer time — denoted by S in the following — is only a small fraction of the totally elapsed time, the gain in parallelization would be certainly covered by the additional communication overhead.

As shown by the benchmark Table 1, the second substep, pair force computation, can be parallelized with high efficiency. Denoting the sequential, i.e. single-node, performance time with $F(1)$ and the corresponding parallel time on N nodes by $F(N)$, it can be seen that $F(N)$ comes very close to the ideal value $F(N)=F(1)/N$. Upon adding the sequential part S, the actual *speed-up*

$$\mathrm{SPU(exp.)}=\frac{F(1)+S}{F(N)+S+C(N)} \tag{73}$$

is reduced according to Amdahl's law. The major source of deterioration is communication $C(N)$, which exhibits a strong N-dependence as opposed to the constant sequential part.

The experimental, i.e. actually achieved, speed-up SPU(exp.) is influenced by two N-dependent quantities $F(N)$ and $C(N)$ as stated by Eq. (68). While $F(N)$ is given to a very good approximation by the simple relation

$$F(N)=F(1)/N \tag{74}$$

Table 1

Number of nodes	$F(N)$ (s)	$F(N)/F(1)$	$F(N)+S+C(N)$ (s)	SPU(exp.)	SPU(theoret.)
iPSC860					
2	41.4	1.99	45.0	1.88	1.90
4	21.0	3.92	24.75	3.43	3.48
8	10.75	7.66	14.75	5.76	5.80
16	5.5	14.96	10.25	8.3	8.09
4HP+2SGI					
3	13.33	2.94	18.0	2.33	2.43
4	10.1	3.88	15.5	2.71	2.84
5	9.5	4.13	15.25	2.75	3.02
6	7.5	5.22	15.0	2.80	3.05
4DEC-alpha					
3	8.1	3.0	9.5	2.63	2.69
4	6.1	4.0	7.5	3.33	3.34

(cf. second column in the corresponding tables), the N-dependence of $C(N)$ varies considerably among different architectures. In order to facilitate interpretation we have fitted simple analytic formulae to the measured communication times. In this empirical way a linear function is found for the hypercube

$$C(N)=K^*(N-1)+L \tag{75}$$

and a quadratic relation

$$C(N)=K^*N^*(N-1)+L \tag{76}$$

for the two workstation clusters. In some sense the additive term L stands for a *Latency*, which is left when extrapolating $C(N)$ to $N=1$. The pre-factor K or $1/K$ may be interpreted as a measure of the data throughput of the respective network. Both parameters are collected in Table 2 for the various architectures.

Combining Eqs. (73)–(75) an expression for the *theoretical* speed-up

$$\mathrm{SPU(theoret.)}=N\,\frac{F(1)+S}{F(1)+N(S+L)+K^*N^*(N-1)} \tag{77}$$

$$\mathrm{SPU(theoret.)}=N\,\frac{F(1)+S}{F(1)+N(S+L)+K^*N^{2*}(N-1)} \tag{78}$$

can be deduced, where the fraction describes the deviation from ideal behavior. While the sequential part S and the latency L appear in a linear function of the number of nodes N, the effective communicational burden increases quadratically (hypercube) or even with the third power (workstation clusters). In case of the 4HP+2SGI cluster, i.e. for slow ethernet links, the weighing factor K is quite high. As a consequence the achieved speed-up is already saturated for five nodes, in other

Table 2
Four parameter model

	iPSC860 (s)	4HP+2SGI (s)	DEC-alpha (s)
$F(1)$	82.3	39.2	24.4
S	2.5	2.8	0.45
L	1.0	0.7	0.45
K	0.125	0.125	0.04

words the gain in computing power is ruined by the rapid increase of communication time.

Bearing in mind that the data throughput of a glass fibre connection exceeds that of ethernet links by two orders of magnitude, the comparison of corresponding K- and L-values leads to a disappointing result. The modest data rate is certainly caused by the communication software PVM, which sends very small data packages, thus increasing latency, and transfers a lot of unnecessary information as well.

Comparing the last two columns in Table 1 one finds that the theoretical relations Eqs. (77) and (78) predict the experimental ones with an uncertainty of a few percent. As these deviations are within experimental errors, the four parameter model is capable of predicting all benchmark results presented here. These four parameters are: The time for computing the pair forces on a single node $F(1)$ and the sequential part S, which consists of integration and constraints, as well as the communication parameters latency L and throughput K. Actual values are collected in Table 2.

All results discussed so far refer to sequence *forces-integration-constraints* (FIC) performed during each CMD-cycle. Taking into account the updating of the pairlist necessary after ten cycles, the rather different communication effort for hypercube

Table 3

Number of nodes	$P(N)$ (s)	$P(N)/P(1)$	SPU(without NB)	SPU(with NB)
iPSC860				
2	81.2	1.98	1.88	1.89
4	41.7	3.85	3.43	3.47
8	22.0	7.3	5.76	5.9
16	12.5	12.9	8.3	8.9
4HP+2SGI				
3	36.0	1.65	2.33	2.27
4	36.5	1.62	2.71	2.61
5	37.0	1.60	2.75	2.65
6	39.0	1.52	2.80	2.68
4DEC-alpha				
3	17.0	2.24	2.63	2.60
4	16.5	2.3	3.33	3.23

or workstation architectures leads to a dissimilar time behavior $P(N)$ for the updating of the pairlist: On the hypercube the ratio $P(N)/P(1)$ exceeds that of the FIC-sequence, while on workstation clusters it is just the other way round. Although the communicational burden in the pairlist substep is two orders of magnitude higher as compared to the FIC-sequence, the hypercube copes better than workstation architectures. As can be seen from the last column of Table 3, this has a positive or negative impact on the total speed-up.

Apart from all parallelism it is interesting to compare the single-node performance. For the three CPU's i860, HP-720 and DEC-alpha, one finds a ratio of 2:1:0.6. In other words, eight nodes of the hypercube should equal four of HP-720 on pure computational reasons. This is confirmed by the respective values of 10.75 and 10.1 s for $F(N)$. More surprising, however, is the equality of the totally elapsed times $F(N)+S+C(N)$ amounting to 14.75 and 15.5 s, respectively. This means that the hypercube architecture manages the communicational burden of eight nodes at the same rate as the workstation architecture for four nodes. Beyond this equality point the workstation network runs into saturation, while the hypercube still provides some gain. This demonstrates once more the crucial balance of single-node performance and throughput of the network.

Acknowledgements

We are greatly indepted to the research group of Prof. Zinterhof at Salzburg University for both, having access to their DEC-alpha cluster and being supported strongly.

References

[1] T.E. Creighton, Proteins: Structures and Molecular Properties, Freeman, New York, 1984.
[2] P.G. Wolynes, J.N. Onuchic, D. Thirumalai, Navigating the folding routes, Science 267 (1995) 1619.
[3] S. Vijay-Kumar, C.E. Bugg, W.J. Cook, Structure of ubiquitin refined at 1.8 Å resolution, J. Mol. Biol. 194 (1987) 531.
[4] D.L. DiStefano, A.J. Wand, Two-dimensional NMR study of human ubiquitin, Biochemistry 26 (1987) 7272.
[5] P.L. Weber, S.C. Brown, L. Mueller, Sequential NMR assignments and secondary structure identification of human ubiquitin, Biochemistry 26 (1987) 7282.
[6] D.M. Schneider, M.J. Dellwo, A.J. Wand, Fast internal main-chain dynamics of human ubiquitin, Biochemistry 31 (1992) 3645.
[7] H.J.C. Berendsen, J. Mavri, Quantum simulation of reaction dynamics by density matrix evolution, J. Phys. Chem. 97 (1993) 13464.
[8] R. Car, M. Parinello, Phys. Rev. Letters 55 (1985) 2471.
[9] K. Laasonen, M. Sprik, M. Parinello, R. Car, Ab-initio liquid water, J. Chem. Phys. 99 (1993) 9080.
[10] M.C. Payne, M.P. Teter, D.C. Allen, T.A. Arias, J.D. Joannopoulos, Iterative minimization techniques for ab-initio total energy calculations, Rev. Mod. Phys. 64 (1992) 1045.
[11] D.K. Remler, P.A. Madden, Mol. Phys. 70 (1990) 921.
[12] L. Verlet, Phys. Rev. 159 (1967) 98.
[13] P. Hohenberg, W. Kohn, Inhomogeneous electron gas, Phys. Rev. 136 (1964) B864.

[14] R.G. Parr, W. Yang, Density-Functional Theory of Atoms and Molecules, Clarendon Press, Oxford, 1989.
[15] P.A.M. Dirac, Note on exchange phenomena in the Thomas atom, Proc. Cambridge Phil. Soc. 26 (1930) 376.
[16] J.C. Slater, A simplification of the Hartree–Fock method, Phys. Rev. 81 (1951) 385.
[17] A.D. Becke, Phys. Rev. A 38 (1988) 3098.
[18] C. Lee, W. Yang, R.G. Parr, Phys. Rev. B 37 (1988) 385.
[19] M. Born, R. Oppenheimer, Ann. Physik 84 (1927) 457.
[20] P. Pulay, Mol. Phys. 17 (1969) 197.
[21] R. Feynman, Forces in molecules, Phys. Rev. 56 (1939) 340.
[22] J. Harris, R.O. Jones, J.E. Miller, Force calculation in the density functional formalism, J. Chem. Phys. 75 (1981) 3904.
[23] W. Kohn, L.J. Sham, Self-consistent equations including exchange and correlation effects, Phys. Rev. 140 (1965) A1133.
[24] A. Warshel, M.J. Levitt, J. Mol. Biol. 103 (1976) 227.
[25] M.J. Fields, P.A. Bash, M.J. Karplus, J. Comput. Chem. 11 (1990) 700.
[26] V. Thery, D. Rinaldi, J.L. Rivail, B. Maigret, G.J. Ferenczy, Comput. Chem. 15 (1994) 269.
[27] J.A. McCammon, B.R. Gelin, M. Karplus, Nature 267 (1977) 585.
[28] Brooks C.L., III, M. Karplus, B.M. Petitt, Advan. Chem. Phys. 71 (1988) 1.
[29] V. Daggett, M. Levitt, Chem. Phys. 158 (1991) 501.
[30] J.A. McCammon, S.C. Harvey, Dynamics of Proteins and Nucleic Acids, Cambridge University Press, London, 1987.
[31] W.F. van Gunsteren, P.K. Weiner, Computer Simulation of Biomolecular Systems, Escom, Leiden, 1989.
[32] W.F. van Gunsteren, H.J.C. Berendsen, in: J. Hermans (Ed.), Molecular Dynamics and Protein Structure, Polycrystal Book Service, Western Springs, IL, 1985.
[33] W.F. van Gunsteren, H.J.C. Berendsen, Mol. Phys. 34 (1977) 1311.
[34] M.A. Thompson, G.K. Schenter, J. Phys. Chem. 99 (1995) 6380.
[35] L. Pauling, Die Natur der chemischen Bindung, Verlag Chemie, Weinheim, 1962.
[36] S. Kirkpatrick, D.C. Gelatt, M.P. Vecchi, Science 220 (1983) 671.
[37] M.L. Glasser, L.J. Zucker, in: D. Henderson (Eds.), Theoretical Chemistry: Advances and Perspectives, vol. 2 (1980) p. 67.
[38] P. Ewald, Ann. Phys. 64 (1921) 253.
[39] M.P. Allen, D.J. Tildesly, Computer Simulation of Liquids, Oxford University Press, Oxford, 1987.
[40] D.J. Adams, G.S. Dubey, J. Comput. Phys. 72 (1987) 156.
[41] H. Schreiber, O. Steinhauser, P. Schuster, Parallel molecular dynamics of biomolecules, Parallel Comput. 18 (1992) 557.
[42] B.I. Dunlap, J.W.D. Conolly, J.R. Sabin, J. Chem. Phys. 71 (1979) 4993.
[43] R. Ahlrichs, Theoret. Chim. Acta 33 (1974) 157.
[44] J.E. Mertz, D.J. Tobias, C.L. Brooks III, U.C. Singh, Vector and Parallel Algorithms for the Molecular Dynamics Simulation of Macromolecules on Shared-Memory Computers, J. Comput. Chem. 12 (1991) 1270.
[45] S. Plimpton, B. Hendrickson, A New Parallel Method for Macromolecular Dynamics Simulation, SANDIA report, August 1994.
[46] W. Smith, T.R. Forester, Parallel macromolecular simulations and the replicated data strategy, Comput. Phys. Comm. 79 (1994) 52.
[47] S.E. DeBolt, P.A. Kollman, AMBERCUBE MD, parallelization of AMBER's macromolecular dynamics module for distributed-memory hypercube computers, J. Comput. Chem. 14 (1993) 312.
[48] H. Schreiber, O. Steinhauser, PSHAKE, a SHAKE alternative for macromolecules, in preparation.
[49] H. Schreiber, O. Steinhauser, G. Löffler, Direct solution of the constraints force problem for water, in preparation.
[50] W.F. van Gunsteren, H.J.C. Berendsen, Groningen Molecular Simulation Library Manual, BIOMOS, Groningen, 1987.

[51] H. Schreiber, O. Steinhauser, Cut-off size does strongly influence molecular dynamics results on solvated polypeptides, Biochemistry 31 (1992) 5856.
[52] H. Schreiber, O. Steinhauser, Molecular dynamics studies of solvated polypeptides: why the cut-off scheme does not work, Chem. Phys. 168 (1992) 75.
[53] H. Schreiber, O. Steinhauser, Taming cut-off induced artifacts in molecular dynamics studies of solvated polypeptides: The reaction field method, J. Mol. Biol. 228 (1992) 909.
[54] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Manchek, V. Sunderam, PVM 3 User's Guide and Reference Manual, Oak Ridge National Laboratory.
[55] IPSC 860 Programmers Reference Manual, Intel Co.