$ms2$: A Molecular Simulation Tool for Thermodynamic Properties

Stephan Deublein[a], Bernhard Eckl[b], Jürgen Stoll[b], Sergey V. Lishchuk[c], Gabriela Guevara-Carrion[a], Colin W. Glass[d], Thorsten Merker[a], Martin Bernreuther[d], Hans Hasse[a], Jadran Vrabec[e,*]

[a]*Lehrstuhl für Thermodynamik, Universität Kaiserslautern, 67653 Kaiserslautern, Germany*
[b]*Institut für Technische Thermodynamik und Thermische Verfahrenstechnik, Universität Stuttgart, 70550 Stuttgart, Germany*
[c]*Department of Mathematics, University of Leicester, Leicester LE1 7RH, United Kingdom*
[d]*Höchstleistungsrechenzentrum Universität Stuttgart (HLRS), 70550 Stuttgart, Germany*
[e]*Lehrstuhl für Thermodynamik und Energietechnik, Universität Paderborn, 33098 Paderborn, Germany*

Abstract

This work presents the molecular simulation program $ms2$ that is designed for the calculation of thermodynamic properties of bulk fluids in equilibrium consisting of small electro-neutral molecules. $ms2$ features the two main molecular simulation techniques, molecular dynamics (MD) and Monte-Carlo. It supports the calculation of vapor-liquid equilibria of pure fluids and multi-component mixtures described by rigid molecular models on the basis of the grand equilibrium method. Furthermore, it is capable of sampling various classical ensembles and yields numerous thermodynamic properties. To evaluate the chemical potential, Widom's test molecule method and gradual insertion are implemented. Transport properties are determined by equilibrium MD simulations following the Green-Kubo formalism. $ms2$ is designed to meet the requirements of academia and industry, particularly achieving short response times and straightforward handling. It is written in Fortran90 and optimized for a fast execution on a broad range of computer architectures, spanning from single processor PCs over PC-clusters and vector computers to high-end parallel machines. The standard Message Passing Interface (MPI) is used for parallelization and $ms2$ is therefore easily portable onto a broad range of computing platforms. Auxiliary feature tools facilitate the interaction with the code and the interpretation of input and output files. The accuracy and reliability of $ms2$ has been shown for a large variety of fluids in preceding work.

Keywords: Molecular simulation, molecular dynamics, Monte-Carlo, grand equilibrium method, vapor-liquid equilibrium, transport properties

1. Program summary

Manuscript: $ms2$: A Molecular Simulation Program for Thermodynamic Properties

Authors: Stephan Deublein, Bernhard Eckl, Jürgen Stoll, Sergey V. Lishchuk, Gabriela Guevara-Carrion, Colin W. Glass, Thorsten Merker, Martin Bernreuther, Hans Hasse, Jadran Vrabec (jadran.vrabec@upb.de)

Title of program: $ms2$

Operating system: Unix/Linux, Windows

*Corresponding author: Jadran Vrabec, Warburger Str. 100, 33098 Paderborn, Germany, Tel.: +49-5251/60-2421, Fax: +49-5251/60-3522, Email: jadran.vrabec@upb.de

Computer: The simulation program $ms2$ is usable on a wide variety of platforms, from single processor machines over PC-clusters and vector computers to vector-parallel architectures. (Tested with Fortran compilers, gfortran, Intel compiler, PathScale compiler, Portland Group compiler and Sun Studio compiler)

Memory: $ms2$ runs on single processors with 512 MB RAM. The memory demand rises with increasing number of processors used per node and increasing number of molecules.

Distribution format: tar.gz

Keywords: Molecular simulation, molecular dynamics, Monte-Carlo, grand equilibrium method, vapor-liquid equilibrium, transport properties, parallel algorithms

Programming language used: Fortran90

External: Message passing interface (MPI).

Classification: 7.7, 7.9, 12

vectorised: Yes. Message Passing Interface (MPI) protocol

Scalability: Excellent scalability up to 16 processors for molecular dynamics and $>$ 512 processors for Monte-Carlo simulations.

Nature of problem: Calculation of application oriented thermodynamic properties for rigid electro-neutral molecules: vapor-liquid equilibria of pure fluids and multi-component mixtures, thermal and caloric data as well as transport properties.

Method of solution: Molecular dynamics, Monte-Carlo, various classical ensembles, grand equilibrium method, Green-Kubo formalism

Restrictions: None. The system size is user-defined. Typical problems addressed by $ms2$ can be solved by simulating systems containing typically 2000 molecules or less.

Unusual Features: Auxiliary feature tools are available for creating input files, analyzing simulation results and visualizing molecular trajectories.

Additional comments: Sample makefiles for multiple operation platforms are provided.

Documentation: Documentation is provided with the installation package and is available at `http://www.ms-2.de`.

Typical running time: The running time of $ms2$ depends on the specified problem, the system size and the number of processes used in the simulation. Running four processes on a ”Nehalem” processor, simulations calculating vapor-liquid equilibrium data take between two and 12 hours, calculating transport properties between six and 24 hours.

2. Introduction

Due to the advances in computing power, methodological efficiency and the development of accurate force fields, it is understood that ”molecular modeling and simulation will become a breakthrough technology that is widely accepted in the chemical industry and applied in conjunction with other predictive methods to meet the industry’s evolving fluid property data needs” [1]. At the core of this prospect lies the sound physical basis of molecular modeling and simulation. It allows to adequately describe structure, energetics and dynamics on the microscopic level, which govern material properties on the macroscopic level. Therefore it provides convenient access to thermodynamic properties, particularly if they are difficult or expensive to obtain by experiment, e.g. at high temperatures and pressures, or may not be measured at all, e.g. when new substances are not available in sufficient quantity. Furthermore, for toxic, explosive or in any other way hazardous substances, measuring thermodynamic properties experimentally can be unfeasible even at moderate thermodynamic conditions, while molecular modeling and simulation offers a reliable route [2].

Presently, the chemical industry extends the scope of thermodynamic models that are regularly used from phenomenological equations of state (EOS) or excess Gibbs energy models to more advanced types, such as statistical EOS like SAFT [3, 4] or continuum solvation models like COSMO-RS [5, 6]. It should be noted that all these advanced models are developed using a bottom-up approach, i.e. bridging molecular properties to the macroscopic level. These methods mainly offer access to aggregated properties, such as the Helmholtz energy of bulk fluids and its derivatives. Regarding transport properties, they are not useful.

Molecular modeling and simulation is applicable with very few constraints as both static and dynamic thermodynamic data may be calculated, be it in bulk fluids or in confinements. Both, equilibrium or non-equilibrium conditions, may be studied. Additionally, detailed insight into the mechanisms on the microscopic level is provided. This versatility, however, is associated with a substantial computational effort which is orders of magnitude larger than needed for the methods mentioned above. Traditionally, molecular simulations were carried out in computing centers of national institutions or universities on powerful computing equipment, which is usually unavailable even in large chemical companies.

However, with the increase in computer power, molecular simulations become feasible even off the shelf, if suitable simulations programs are available. The scope of data accessible within reasonable times even for industrial workflows is increasing rapidly. Using molecular modeling and simulation can therefore contribute for a reduction of process development time and production costs [7].

Molecular modeling and simulation is rewarding, because it provides reliable predictions of essentially all thermodynamic properties in a consistent manner [8]. Molecular models yield predictions for any property at any condition. This is of particular interest for industrial applications, where a wide variety of properties needs to be known.

To further stimulate this issue, the Industrial Fluid Properties Simulation Collective [1] has organized six Simulation Challenges to date [8, 9, 10, 11, 12, 1]. The goal of these Simulation Challenges is to assess the capabilities of molecular methods regarding typical industrial tasks, where classical methods are insufficient.

The present paper presents a molecular simulation program, named $ms2$, which successfully competed in two Simulation Challenges, finishing second place 2006 [11] and first place the following year [8]. $ms2$ is aimed at the calculation of thermodynamic properties that are needed for applications in the chemical industry, without requiring expert knowledge by the user. It was developed for rigid molecular models with a focus on condensed phases, covering mixtures containing an arbitrary number of small molecular species.

The simulation program $ms2$ is optimized for a fast execution on a broad range of computer architectures, spanning from single processor PCs over PC-clusters and vector computers to high end parallel machines. It is a standalone Fortran90 code that does not require any libraries or have any other software prerequisites. $ms2$ is provided as a source code. It only needs to be compiled. The structure of $ms2$ is modular and object-oriented, except at the very core of energy and force calculations, where some structural compromises were made to achieve a better performance.

$ms2$ is aimed at homogeneous bulk systems in equilibrium. For VLE calculations, the coexisting phases are simulated independently and subsequently. Thus, it is usually sufficient to sample molecular systems containing in the order of 10^3 molecules to obtain statistically reliable results [13]. Applying standard cut-off radii, the intermolecular interactions are evaluated roughly up to the maximal distance, which is given by the edge length of the cubic simulation volume. Therefore, spatial decomposition schemes for parallelization are not rewarding. Instead, Plimpton's method [14] was implemented for parallel execution of molecular dynamics (MD), which scales reasonably well up to 16 or 32 processors, depending on the particular architecture. For Monte-Carlo (MC), a optimal parallelization scheme was introduced, taking advantage of the stochastic nature of such simulations. The computing resources are splitted into single runs each on one node, sampling the phase space independently. Both parallelization methods allow for short response times[1]. A typical molecular simulation of a vapor liquid equilibrium state point takes roughly six hours on a current workstation. The main expenses for determining thermodynamic data $in\ silico$ are installation and maintenance of computing equipment and/or computing time, which is presently below US$ 0.35/CPU-hour [15].

The simulation program $ms2$ presented here was developed in a long-time cooperation of engineers and computer scientists. With the release of $ms2$, it is intended to facilitate the transfer of a state of the art and user-friendly molecular modeling and simulation package to academic institutions and the chemical industry. The release package consists of the simulation program $ms2$ itself and auxiliary feature tools for setup and analysis, including a simple 3D visualization tool to monitor the molecular trajectories. Furthermore, accurate molecular models for more than 100 small molecules are provided.

In sections 3 and 4 of this paper, $ms2$ and the implemented thermodynamic properties are outlined. In section 5, the structure of the code is explained, while section 6 compares its performance to similar simulation programs and section 7 describes the auxiliary feature tools. Section 8 of this paper draws a conclusion and offers a brief outlook on future developments and improvements.

[1] Under "response time" we understand the real time difference between submission and termination of a simulation run.

3. Simulation program $ms2$

The simulation program $ms2$ is capable of sampling the phase space for rigid electro-neutral molecules by applying the two most fundamental molecular simulation techniques, i.e. MD and MC. MD simulations rely on the numerical solution of Newton's equations of motion: for a point in time, the intermolecular interactions, in particular the resulting forces and torques, are evaluated and treated as constant for a specified time step. They are the driving forces of the molecular motion. The displacements for the time step are calculated on that basis, resulting in a new configuration. This process is repeated in a loop. The chronologically ordered configurations are a time discretized approximation of a molecular process. Both static and dynamic thermodynamic properties are determined via time averages. MC simulation explores the phase space stochastically for a given molecular system. Molecules in the simulation volume are displaced randomly. The probability of accepting the displacement is chosen such that a representative set of configurations is obtained. The Markov chain of configurations generated in this way allows for a rigorous calculation of static thermodynamic properties via ensemble averages.

3.1. Overview

The simulation program $ms2$ allows for the determination of static and dynamic thermodynamic properties in equilibrium. The implemented static properties are:

- Thermal and caloric properties
- Chemical potential
- Vapor-liquid equilibria
- Henry's law constant
- Second virial coefficient

The transport properties can be calculated on the fly during a MD simulation with a reasonable additional computational effort using the Green-Kubo formalism [16, 17]. The implemented dynamic properties are:

- Self-diffusion coefficient
- Maxwell-Stefan diffusion coefficient
- Shear viscosity
- Bulk viscosity

The model class that is supported by $ms2$ covers rigid multi-center Lennard-Jones (LJ) 12-6 interaction sites with an arbitrary number of superimposed electrostatic sites [18, 19, 20, 21]. The supported electrostatic models are point charges, point dipoles and point quadrupoles, which can be positioned anywhere within the molecule. Currently, $ms2$ is designed for electro-neutral species.

The quality of thermodynamic properties calculated by molecular simulation is basically determined by two factors: first, the employed molecular model, i.e. the force field, which fully defines the thermodynamic properties

and deviates to some degree from the behavior of the real fluid; second, the sampling of the phase space during simulation, which is associated with statistical uncertainties.

Molecular models have been investigated with $ms2$ in numerous cases in the past [22, 23, 24, 25, 26, 27, 28]. For many of these models, the geometric and electrostatic interaction parameters were passed on from ab-initio calculations. The remaining parameters were adjusted to reproduce the vapor pressure and the saturated liquid density of the regarded pure substance. These molecular models, combined with $ms2$ and its analysis methods, allow for time-efficient high quality simulations.

The second factor can be influenced for a given simulation method by the number of molecules and the number of sampled configurations. The more data, the lower the statistical uncertainties.

3.2. Methods

3.2.1. Ensembles

Following ensembles are currently supported by $ms2$:

- canonical ensemble (NVT) - MD and MC
- micro-canonical ensemble (NVE) - MD
- isobaric-isothermal ensemble (NpT) - MD and MC
- grand equilibrium method (pseudo-μVT) - MC

Most of these ensembles are well known and widely in use [29]. Therefore, the discussion is restricted here to the grand equilibrium method, since it was developed recently [30] and is probably new to many readers.

The grand equilibrium method is a technique to determine the VLE of pure substances or mixtures. It is a two-step procedure, where the coexisting phases are simulated independently and subsequently. The specified thermodynamic variables for the VLE are the temperature T and the composition $\boldsymbol{x}$ of the liquid. In the first step, one NpT simulation of the liquid phase is performed at T, $\boldsymbol{x}$ and some pressure p_0 to determine the chemical potentials μ_i^l and the partial molar volumes v_i^l of all components, which corresponds to the molar volume in case of a pure substance. If the entropic properties are determined by Widom's test molecule method [31], both MD or MC can be used to sample the phase space. A more advanced technique for these properties is implemented in combination with MC, i.e. gradual insertion [32, 33] (see below). On the basis of the chemical potentials and partial molar volumes at p_0, first order Taylor expansions can be made for the pressure dependence

$$\mu_i^{\mathrm{l}}\big(T, \boldsymbol{x}, p\big) \approx \mu_i^{\mathrm{l}}\big(T, \boldsymbol{x}, p_0\big) + v_i^{\mathrm{l}}(T, \boldsymbol{x}, p_0) \cdot \big(p - p_0\big) \,. \tag{1}$$

Note that $ms2$ yields $\mu_i\big(T, \boldsymbol{x}, p\big) - \mu_i^{\mathrm{id}}(T)$, where $\mu_i^{\mathrm{id}}(T)$ is the solely temperature dependent ideal contribution. $\mu_i^{\mathrm{l,id}}(T)$ does not need to be determined for VLE calculations, because it cancels out when Eq. (1) is equated to the corresponding expression for the vapor.

In the second step, one pseudo-μVT simulation [30] is performed for the vapor phase on the basis of Eq. (1) that yields the saturated vapor state point of the VLE. This simulation takes place in a pseudo ensemble in the

sense that the specified chemical potentials are not constant, but dependent on the actual pressure of the vapor phase. However, experience based on hundreds of systems [25, 28] shows that the vapor phase simulation rapidly converges to the saturated vapor state point during equilibration so that effectively the equilibrium chemical potentials are specified via the attained vapor pressure. The number of molecules varies during the vapor phase simulation of the grand equilibrium method. Starting from a specified number of molecules in an arbitrarily chosen gaseous state, $ms2$ adjusts the extensive volume V after an equilibration period so that the vapor density does not have to be known in advance.

The grand equilibrium method has been widely used and compares favorably to other methods used for vapor liquid equilibrium simulations, as shown in section 6.

3.2.2. Integrators, thermostat and barostat

In $ms2$, two integrators are implemented to solve Newton's equations of motion during MD simulation: Leapfrog and Gear predictor-corrector. These integrators are well known and described in literature [29]. The leapfrog integrator is a second order integrator that requires little computational effort, while being robust for many applications. The error of the method scales quadratically with the length of the chosen time step. The Gear predictor-corrector integrator is implemented with fifth order and is more accurate for small time steps compared to the Leapfrog integration [29]. The computational demand for both integration schemes is similar as implemented in $ms2$. The Gear integration, though being of higher order, is only 0.3% slower than the Leapfrog algorithm for the total calculation of one MD time step with a molecular model composed of three LJ sites, having three rotational degrees of freedom.

For MC, the Markov chain is generated by repeatedly disturbing the system by translational or rotational motion of a molecule and evaluating the resulting configurations with respect to energy using Metropolis acceptance criterion. The thermostat incorporated in $ms2$ is velocity scaling. Here, the velocities are scaled such that the actual kinetic energy matches the specified temperature. The scaling is applied equally over all molecular degrees of freedom. The pressure is kept constant using Andersen's barostat in MD, and random volume changes evaluated according to Metropolis acceptance criterion in MC, respectively.

3.2.3. Basics on simulations with $ms2$

The basic molecular simulation techniques employed in $ms2$ are well described in the literature [29, 34, 35] and are thus not repeated here. All non-standard methods that are implemented in $ms2$ are introduced in section 4 or the Supplementary Material. The following paragraph briefly describes the most significant assumptions and techniques employed in $ms2$.

Simulations with $ms2$ are performed in a quasi-infinite Cartesian space, using periodic boundary conditions [36] and the minimum image convention [37]. The molecular interactions are considered to be pairwise additive, like in most other simulation packages. Including three- and many-body interactions leads to an increase in computational effort while the benefits are questionable. As usual, the computational effort in $ms2$ is reduced by the introduction of a cut-off radius, up to which the intermolecular interactions are explicitly evaluated. The contributions of interactions with molecules beyond the cut-off radius are accounted for by correction schemes.

For the electrostatic models, the reaction field method is included, while for the dispersive and repulsive interactions, a homogeneous fluid beyond the cut-off radius is assumed. The long range contributions are added to all thermodynamic data that are calculated in $ms2$.

Many thermodynamic properties calculated with $ms2$ are residual quantities, indicated by the superscript "res". To compare them to data that are measured on the macroscopic level, the solely temperature dependent contributions of the ideal gas have to be added. These ideal properties are accessible e.g. by quantum chemical methods and can often be found in data bases [38].

3.3. Thermodynamic properties

The simulation program $ms2$ calculates the thermodynamic properties from the trajectories (MD) or Markov chains (MC) on the fly. The results are written to file with a specified frequency during the course of the simulation. The statistical uncertainties of all results are estimated using the block averaging method according to Flyvbjerg and Petersen [39] and the error propagation law.

3.3.1. Density, pressure, internal energy and enthalpy

The static thermodynamic properties accessible via $ms2$ depend on the chosen ensemble. At constant temperature and volume (NVT ensemble), e.g. the pressure is determined by the virial expression W. Other important thermodynamic properties are accessible at constant temperature and pressure. In the NpT ensemble, the volume is a fluctuating parameter that on average yields the volume corresponding to the specified temperature and pressure. In both ensembles, the residual internal energy is the sum of all pairwise interaction energies u_{ij} and the appropriate long range correction. The residual enthalpy is linked to the residual internal energy, pressure and volume of the system using the thermodynamic definition.

3.3.2. Second derivatives

The implemented second derivatives vary with the employed ensemble. In the NVT ensemble, the residual isochoric heat capacity c_v^{res} is determined by fluctuations of the residual potential energy U^{res}. The partial derivative of the potential energy with respect to the volume at constant temperature $\left(\partial U^{\mathrm{res}}/\partial V\right)_T$ is determined by fluctuations of the residual potential energy U^{res} and the virial W.

In the NpT ensemble, the residual isobaric heat capacity c_p^{res}, the isothermal compressibility β_T and the volume expansivity α_p are functions of ensemble fluctuations [29]. The residual isobaric heat capacity c_p^{res} is related to fluctuations of the residual enthalpy H^{res} and the isothermal compressibility β_T is calculated from the volume fluctuations. The partial derivative of the residual enthalpy with respect to the pressure at constant temperature $\left(\partial H^{\mathrm{res}}/\partial p\right)_T$ is determined by fluctuations of volume and residual internal energy and the volume expansivity α_p again to volume and residual enthalpy fluctuations.

The speed of sound c is the velocity of a sound wave traveling through an elastic medium. It is defined by the isothermal compressibility β_T, the volume expansivity α_p, the isobaric heat capacity c_p and the temperature T. In $ms2$, the speed of sound is calculated both for pure components and mixtures in the NpT ensemble.

3.3.3. Chemical potential

The chemical potential of a given component can be separated into a solely temperature dependent contribution $\mu_i^{\mathrm{id}}(T)$ of the ideal gas and the remaining contribution $\mu_i\big(T, \boldsymbol{x}, p\big) - \mu_i^{\mathrm{id}}(T)$. The solely temperature dependent ideal gas contribution cancels out in the calculation of phase equilibria and is therefore not implemented in $ms2$. The chemical potential depends on the real substance behavior due to the molecular interactions and can be determined with $ms2$ using two different techniques: Widom's test molecule method and gradual insertion.

Widom. A conceptually straightforward approach to calculate the chemical potential of a component i was presented by Widom [31]. It allows for a determination of the chemical potential with low computational cost, both for pure substances and mixtures. For a three-center LJ fluid, the computational demand is shown in Table 1. The execution time for the determination of the chemical potential increases roughly linearly with the specified number of test molecules that are sampled.

The accuracy of the calculation varies with the number of test molecules and the density of the investigated fluid. For very dense fluids, the results are subject to poor statistics and the method may even fail. Within limits, lower statistical uncertainties of the chemical potential can be achieved by inserting a large number of test molecules into the simulation volume.

Gradual insertion. A more advanced method to determine the chemical potential, which is reliable also at very high densities, is gradual insertion. It is briefly described here, further details including all parameters are discussed in literature [32, 33, 40, 41].

Instead of inserting complete test molecules, a fluctuating molecule is introduced into the simulation, which can appear in different states of coupling with the other molecules. In the decoupled state, the fluctuating molecule does not interact at all with the other molecules, while in the fully coupled state, it acts like a real molecule of the specified component i. Between these states, a set of partially coupled states has to be defined, each with a larger fraction of the real molecule interaction, cf. Figure 1.

The N-1 real molecules plus the fluctuating molecule π_l in the state l form a set of sub-ensembles, which can be depicted by the following scheme

$$[N + \pi_0] \leftrightarrow [N + \pi_1] \leftrightarrow ... \leftrightarrow [N + \pi_l] \leftrightarrow ... \leftrightarrow [N + \pi_{k-1}] \leftrightarrow [N + \pi_k] \,. \tag{2}$$

To switch between neighboring sub-ensembles, an additional move is introduced in a standard MC simulation. The probability of accepting a change of the fluctuating molecule from a state of coupling l to a state of coupling m is given by

$$P_{\mathrm{acc}}(l \to m) = \min\big(1, \frac{\omega_m}{\omega_l} \exp\big(-\frac{\psi_m - \psi_l}{k_{\mathrm{B}}T}\big)\big) \,, \tag{3}$$

where ψ_l denotes the interaction energy of the fluctuating molecule in the state l with all other $N-1$ real molecules and $k_{\mathrm{B}}T$ is the Boltzmann constant multiplied by the temperature. The states of coupling are weighted by the weighting factors ω_l to avoid an unbalanced sampling of the different states. If specified, the weighting factors are adjusted during simulation, depending on the number of times N_s the fluctuating molecule appeared

in state l, according to

$$\omega_l^{new} = \omega_l^{old} \frac{N_s(k)}{N_s(l)} \,. \tag{4}$$

The fully coupled state k serves as a reference state for the weighting factors. Local relaxation around the fluctuating molecule is enhanced by biased translational and rotational moves in the vicinity of the fluctuating molecule throughout the simulation [33]. The chemical potential $\mu_i\big(T, \boldsymbol{x}, p\big) - \mu_i^{\mathrm{id}}(T)$ of component i is then determined by

$$\mu_i\big(T, \boldsymbol{x}, p\big) - \mu_i^{\mathrm{id}}(T) = k_{\mathrm{B}} T \ln \big\langle \frac{N_i}{V} \frac{\omega_k}{\omega_0} \frac{\mathrm{Prob}[N + \pi_0]}{\mathrm{Prob}[N + \pi_k]} \big\rangle \,, \tag{5}$$

where Prob$[N+\pi_0]$ and Prob$[N+\pi_k]$ are the probabilities to observe an ensemble with the fluctuating molecule in the fully decoupled and fully coupled state, respectively.

The gradual insertion method yields good results for the chemical potential even in cases where Widom's test molecule method fails. Disadvantages of the method are the extended simulation time and the additional effort needed to define the fluctuating states.

3.3.4. Henry's law constant

The solubility of solutes in solvents is characterized by the Henry's law constant. In solvents of a given composition, it is solely a function of temperature. The Henry's law constant H_i is related to the residual chemical potential of the solute i at infinite dilution ${\mu_i}^{\mathrm{res},\infty}$ as defined in e.g. [42] by

$$H_i = \rho k_{\mathrm{B}} T \exp\big(\frac{{\mu_i}^{\mathrm{res},\infty}}{k_{\mathrm{B}} T}\big) \,, \tag{6}$$

where ρ denotes the solvent density. The residual chemical potential of a component at infinite dilution can often be calculated using Widom's test molecule method. In this case, a simulation of the solvent is performed in the NpT ensemble and the saturated vapor pressure of the solvent, while the solute is introduced as an additional component of the mixture with a molar fraction of zero. Then, the solute is only added in form of test molecules to determine its chemical potential at infinite dilution. For dense phases, the chemical potential and thus the Henry's law constant can be calculated with the gradual insertion method, if Widom's method fails. The number of solute molecules in the simulation is reduced to one, representing the infinitely diluted molecule.

3.3.5. Second virial coefficient

The second virial coefficient is related to the intermolecular potential by [43]

$$B = -2\pi \int_0^\infty \mathrm{d}r_{ij} \big\langle \exp\left(-\frac{u_{ij}(r_{ij}, \boldsymbol{\omega}_i, \boldsymbol{\omega}_j)}{k_{\mathrm{B}} T}\right) - 1 \big\rangle_{\boldsymbol{\omega}_i, \boldsymbol{\omega}_j} r_{ij}^2, \tag{7}$$

where the $\langle ... \rangle$ brackets indicate the average over the orientations $\boldsymbol{\omega}_i$ and $\boldsymbol{\omega}_j$ of two molecules i and j separated by a center of mass distance r_{ij}. The integrand, called Mayer's f-function, is evaluated at numerous distances in an appropriate range. $ms2$ evaluates Mayer's f-function by averaging over numerous random orientations at each radius.

3.3.6. Transport properties

In $ms2$, transport properties are calculated using equilibrium MD. Here, the fluctuations of a system around its equilibrium state are evaluated as a function of time. The Green-Kubo formalism relates these microscopic fluctuations to the respective transport properties.

Diffusion coefficients. The self-diffusion coefficient D_i is related to the mass flux of single molecules within a fluid. Therefore, the relevant Green-Kubo expression is based on the individual molecule velocity autocorrelation function [44]. Since all molecules contribute to the self-diffusion coefficient, the autocorrelation function is averaged over all N_i molecules of component i in the ensemble to achieve better statistics. In binary mixtures, the Maxwell-Stefan diffusion coefficient $Đ_{ij}$ is defined by [45]

$$Đ_{ij} = \frac{x_j}{x_i}\Lambda_{ii} + \frac{x_i}{x_j}\Lambda_{jj} - \Lambda_{ij} - \Lambda_{ji} \,, \tag{8}$$

where $x_i = N_i/N$ and Λ_{ij} can be written in terms of the center of mass velocity

$$\Lambda_{ij} = \frac{1}{3N}\int_0^\infty \mathrm{d}t \, \langle \sum_{k=1}^{N_i} \mathbf{v}_{i,k}(0) \cdot \sum_{l=1}^{N_j} \mathbf{v}_{j,l}(t) \rangle \,. \tag{9}$$

From the expressions above, the collective character of the Maxwell-Stefan diffusion coefficient is evident. This leads to significantly less data for a given system size and time step and therefore to larger statistical uncertainties than in case of the self-diffusion coefficient. Note that equations for the Maxwell-Stefan diffusion coefficient are implemented in $ms2$ both for binary and ternary mixtures [45].

Shear viscosity. The shear viscosity η, as defined by Newton's "law" of viscosity, is a measure of the resistance of a fluid to a shearing force [46]. It is associated with the momentum transport under the influence of velocity gradients. Hence, the shear viscosity can be related to the time autocorrelation function of the off-diagonal elements of the stress tensor $\mathbf{J}_p$ [44]

$$\eta = \frac{1}{V k_\mathrm{B} T}\int_0^\infty \mathrm{d}t \, \left\langle J_p^{xy}(t) \cdot J_p^{xy}(0) \right\rangle \,. \tag{10}$$

Averaging over all three independent elements of the stress tensor, i.e. J_p^{xy}, J_p^{xz} and J_p^{yz}, improves the statistics. The component J_p^{xy} of the microscopic stress tensor $\mathbf{J}_p$ is given in terms of the molecular positions and velocities by [46]

$$J_p^{xy} = \sum_{i=1}^{N-1} m v_i^x v_i^y - \sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\sum_{k=1}^{n}\sum_{l=1}^{n} r_{ij}^x \frac{\partial u_{ij}}{\partial r_{kl}^y}. \tag{11}$$

Here, the lower indices l and k indicate the n interaction sites of a molecule and the upper indices x and y denote the spatial vector components, e.g. for velocity v_i^x or site-site distance r_{ij}^x. The first term of Eq. (11) is the kinetic energy contribution and the second term is the potential energy contribution to the shear viscosity. Consequently, the Green-Kubo integral (10) can be decomposed into three parts, i.e. the solely kinetic energy contribution, the solely potential energy contribution and the mixed kinetic-potential energy contribution [46].

Bulk viscosity. The bulk viscosity η_v refers to the resistance to dilatation of an infinitesimal volume element at constant shape [47]. The bulk viscosity can be calculated by integration of the time-autocorrelation function of the diagonal elements of the stress tensor and an additional term that involves the product of pressure p and volume V, which does not appear in the shear viscosity, cf. Eq. (10). In the NVE ensemble, the bulk viscosity is given by [44, 48]

$$\eta_v = \frac{1}{Vk_{\mathrm{B}}T} \int_0^\infty \mathrm{d}t \left\langle (J_p^{xx}(t) - pV(t)) \cdot (J_p^{xx}(0) - pV(0)) \right\rangle. \tag{12}$$

The diagonal component J_p^{xx} of the microscopic stress tensor $\mathbf{J}_p$ is defined analogous to Eq. (11). The statistics of the ensemble average in Eq. (12) can be improved using all three independent diagonal elements of the stress tensor J_p^{xx}, J_p^{yy} and J_p^{zz}. Eqs. (11) and (12) can directly be applied for mixtures.

4. Simulation program $ms2$: Detailed description

This section describes $ms2$ in detail and important options for its application are introduced. It is intended to allow for a better understanding of the simulation program and to facilitate access to the program features.

4.1. Input and output

$ms2$ was designed to be an easily applicable simulation program. Therefore, the input files are restricted to one file for the definition of simulation scenario and one file for each of the molecular species that are used in the simulation. The output files contain structured information on the simulation. All calculated thermodynamic properties are summarized in one output file, which is straight forwardly readable and self-explaining. For a more detailed evaluation of the simulation, the instantaneous and running averages of the most important thermodynamic properties are written to other files. In total, the current status of the simulation and many more details are written to six files and can be accessed during execution.

4.2. Reduced quantities

The simulation program $ms2$ internally uses reduced quantities for its calculations. All quantities are reduced by a reference length σ_{R}, a reference energy ϵ_{R} and a reference mass m_{R}, respectively. These reference values are input variables and may, in principle, be chosen arbitrarily. However, it is recommended to use a reference length σ_{R} in the order of 3 Å, a reference energy $\epsilon_{\mathrm{R}}/k_{\mathrm{B}}$ in the order of 100 K and a reference mass m_{R} in the order of 50 atomic units. The reduction scheme for the most important physical quantities is listed in the Supplementary Material. From these properties, the reduced form of all other quantities can be derived. An exception in the reduction scheme is the chemical potential, which is normalized in $ms2$ by $k_{\mathrm{B}}T$ instead of ϵ_{R}.

4.3. Molecular positions and orientations

The simulation program $ms2$ updates only the positions of the centers of mass and the orientations of the molecules. From these values, the coordinates of the sites are derived on the fly. The center of mass of each molecule is stored in Cartesian coordinates. The absolute positions are reduced by the reference length σ_{R} and scaled by the reduced edge length of the cubic simulation volume, leading to position values in the range of

$-0.5 < x,\, y,\, z < 0.5$. The advantage of this scaling scheme lies mainly in the efficient application of the periodic boundary condition [36] and the minimum image convention [37].

The molecular orientations are stored using normalized quaternions [29]. Quaternions are a biunique representation that avoid divergence problems at low angles. Their use allows for an efficient calculation of site positions, while being less demanding concerning execution time and memory than updating and storing all site positions.

4.4. Initial configuration

It is important to define a stable and physically reasonable starting configuration for a simulation in a reliable way. In $ms2$, the molecules are initially placed on a face-centered cubic lattice in order to avoid overlaps between molecules. For mixtures, the positions of the different molecule species are distributed randomly on the lattice, ensuring a homogeneous distribution of all components in the simulation volume.

After the initial molecule placement, the configuration can be relaxed by translational and rotational MC moves. This step is part of the initialization process and can therefore be performed regardless of the simulation technique used. The number of relaxation moves is user defined and should be chosen large enough to achieve a physically reasonable starting configuration. For MD, the molecules are subsequently assigned with initial velocities such that the temperature is specified and no net translational and rotational moment is present. It is recommended to continue with a MD equilibration until a physically reasonable configuration and distribution of velocities is achieved. The MC loops relax possible overlaps in the initial configuration, whereas the MD equilibration drives the system into a physically reasonable dynamic microstate. Note that the equilibration process does not contribute to the calculation of the thermodynamic properties.

4.5. Intermolecular interactions

4.5.1. Dispersive and repulsive interactions

In $ms2$, the dispersive and repulsive interactions between molecules are reduced to pairwise interactions u_{ij}^{LJ} of the different molecule sites i and j, which are modeled by the 12-6-LJ potential [49]

$$u_{ij}^{\mathrm{LJ}} = 4\varepsilon\big(\big(\frac{\sigma}{r_{ij}}\big)^{12} - \big(\frac{\sigma}{r_{ij}}\big)^{6}\big)\,. \tag{13}$$

Here, the site-site distance between two interacting LJ sites i and j is denoted by r_{ij}, while σ and ε are the LJ size and energy parameters, respectively. The LJ potential is widely used and allows for a fast computation of these basic interactions. It has only two parameters, which facilitates the parameterization of molecular models.

For pure components, the interactions between two different LJ sites are described by the Lorentz-Berthelot combination rules [50, 51]. For mixtures, the combination rules are extended to the modified Lorentz-Berthelot rules, which include two additional parameters η and ξ to describe the interactions between LJ sites of unlike molecules [52]

$$\sigma_{ij} = \eta\frac{\sigma_i + \sigma_j}{2}\,, \tag{14}$$

$$\varepsilon_{ij} = \xi\sqrt{\varepsilon_i\varepsilon_j}\,. \tag{15}$$

The two parameters scale all LJ interactions between molecules of different components equally. Arbitrary modifications of the combination rules are possible. $ms2$ allows the specification of η and ξ for every molecule pair independently and thus the free parameterization of the combination rules.

4.5.2. Electrostatic interactions

$ms2$ considers electrostatic interactions between electro-neutral molecules. Three different electrostatic site type models are available: point charge, point dipole and linear point quadrupole. The two higher order electrostatic interaction sites integrate characteristic arrangements of several single partial charges on a molecule. The simulative advantage of higher order polarities is faster execution and a better description of the electrostatic potential of the molecule for a given number of electrostatic sites [53]. Combining two partial charges to one point dipole reduces the computational effort with $ms2$ by about 30%, combining three point charges to one point quadrupole reduces the computational demand by even 60%. An additional advantage is the simplification of the molecular model, which reduces the number of molecular model parameters. In the following, the implemented electrostatic interactions between sites of the same type are briefly described. The interaction potentials between unlike electrostatic site types is presented in the Supplementary Material.

Point charge. Point charges are first order electrostatic interaction sites. The electrostatic interaction between two point charges q_i and q_j is given by Coulomb's law [29]

$$u_{ij}^{qq}(r_{ij}, q_i, q_j) = \frac{1}{4\pi\varepsilon_0}\frac{q_i q_j}{r_{ij}} \; . \tag{16}$$

This interaction decays with the inverse of r_{ij} and is therefore significant up to very large distances. For computational efficiency, the Coulombic contributions to the potential energy are explicitly evaluated up to a specified cut-off radius r_c. The long range interactions with charges beyond r_c are corrected for by the reaction field method. Its application to point charges is discussed below.

Point dipole. A point dipole describes the electrostatic field of two point charges with equal magnitude, but opposite sign at a mutual distance $a \to 0$. Its moment μ is defined by $\mu = qa$. The electrostatic interaction between two point dipoles with the moments μ_i and μ_j at a distance r_{ij} is given by [29, 43]

$$u_{ij}^{DD}(r_{ij}, \theta_i, \theta_j, \phi_{ij}, \mu_i, \mu_j) = \frac{1}{4\pi\epsilon_0}\frac{\mu_i\mu_j}{r_{ij}^3}\big(\sin\theta_i \sin\theta_j \cos\phi_{ij} - 2\cos\theta_i \cos\theta_j\big) \; , \tag{17}$$

with θ_i being the angle between the dipole direction and the distance vector of the two interacting dipoles and ϕ_{ij} being the azimuthal angle of the two dipole directions, cf. Figure 2. In $ms2$, the interaction between two dipoles is explicitly evaluated up to the specified cut-off radius r_c. The long range contributions are considered in $ms2$ by the reaction field method [29].

Linear point quadrupole. A linear point quadrupole describes the electrostatic field induced either by two collinear point dipoles with the same moment, but opposite orientation at a distance $a \to 0$, or by at least three collinear point charges with alternating sign (q, $-2q$, q). The resulting quadrupole moment Q is defined by

$Q = 2aq$. The electrostatic interaction between two linear point quadrupoles with the moments Q_i and Q_j at a distance r_{ij} is given by [43, 54]

$$\begin{aligned} u_{ij}^{QQ}(r_{ij}, \theta_i, \theta_j, \phi_{ij}, Q_i, Q_j) = & \frac{1}{4\pi\varepsilon_0}\frac{3}{4}\frac{Q_i Q_j}{r_{ij}^5} \\ & \big[1 - 5\big((\cos\theta_i)^2 + (\cos\theta_j)^2\big) - 15(\cos\theta_i)^2(\cos\theta_j)^2 + \\ & 2\big(\sin\theta_i \sin\theta_j \cos\phi_{ij} - 4\cos\theta_i \cos\theta_j\big)^2\big], \end{aligned} \tag{18}$$

where the angles θ_i, θ_j and ϕ_{ij} indicate the relative angular orientation of the two point quadrupoles, as discussed above. Note that no long range correction is necessary for this interaction type if the fluid is isotropic.

Reaction field method. The truncation of interactions between first and second order electrostatic sites leads to errors that need to be corrected. The reaction field method [34, 55] is implemented in $ms2$ for this task, being widely used and well accepted [56, 57]. Its advantages are accuracy and stability, while requiring little computational effort compared to other techniques like Ewald summation [58]. However, it is limited to electro-neutral systems, thus $ms2$ is currently restricted to electro-neutral molecules. The basic assumption of the reaction field method implemented in $ms2$ is that the system is sufficiently large so that tinfoil boundary conditions ($\varepsilon_s \to \infty$) are applicable without a loss of accuracy. This is the case for $N \geq 500$ [59, 60, 61, 62].

4.6. Cut-off modes

$ms2$ supports two different cut-off modes: the site-site cut-off and the center of mass cut-off. The site-site cut-off mode explicitly considers the interactions between all sites that are within a distance of r_c. Beyond r_c, the long-range contributions to the energy and pressure are estimated by analytical functions assuming a homogeneous fluid [29]. A disadvantage of the site-site cut-off arises, if molecular models contain point charges. In many cases close to r_c, molecules are only partially considered in the explicit calculation, cf. Figure 3. The point charges within the cut-off radius may be unbalanced so that an overall charge within the cut-off sphere might occur. For this condition, the reaction field is not valid.

A more robust alternative is the center of mass cut-off mode. It considers all interaction sites of different molecules explicitly, if their molecular centers of mass are within the cut-off radius. The long-range contributions beyond r_c due to LJ interactions are approximated by the formulations of Lustig [63]. Note that the computational advantage of the center of mass cut-off mode increases with the number of sites per molecule.

4.7. Monte-Carlo algorithm

Thermodynamic properties are determined by MC simulation via a Markov chain of molecular configurations. In $ms2$, this Markov chain is generated by executing a loop of N_MC moves per configuration, where N_MC is defined by

$$N_\mathrm{MC} = \frac{1}{3}\sum_{i=1}^{N} N_{i,\mathrm{DGF}} \; . \tag{19}$$

Here, N represents the number of molecules in the system and $N_{i,\mathrm{DGF}}$ is the number of degrees of freedom of molecule i. For MC simulations, three different moves are implemented in $ms2$, translational and rotational

displacement of a single molecule and fluctuation of the simulation volume. All three moves are well known and widely in use [29] and thus not further discussed here.

The acceptance of MC moves is associated with energy differences before and after that perturbation. To speed up execution, it is avoided in $ms2$ to calculate the old energy for each attempted move. Instead, the pairwise interactions of all molecules with all the remaining $N-1$ molecules in the system are stored in a $N \times N$ matrix. The sum of each column and row of that matrix, respectively, equals the potential energy of one particular molecule. If a molecule i is assigned to be moved, its old energy is determined by simply summing up all contributions in column i of the energy matrix. After the move, its new energy is determined by calculating the pairwise interactions with the remaining molecules and the individual contributions are stored in a vector of dimensions $N \times 1$. If the move is accepted, the vector replaces column i as well as row i of the energy matrix. If the move is rejected, the vector is discarded. The same technique is used for the virial contributions.

5. Implementation

In this section, the code design and implementation issues of $ms2$ are discussed. These explanations of the source code are intended to help understanding the program and to encourage the reader to further develop and improve the code. New developers should get a smooth entry and the possibility to make fast adaptations to new problems. Due to the experience of the core developers and the suitability for an efficient numerical code, Fortran90 was chosen as programming language. The program makes extensive use of the concepts introduced with Fortran90, notably modular programming. A wide variety of different simulation setups is possible, each requiring different computations. This variety leads to a need for a highly modular structure, where modules have clear-cut interfaces. Given such modularity, code parts that are not needed for the user defined simulation setup, can be simply skipped within the calculation. Besides avoiding if-statements at computationally demanding points and therefore leading to a better data flow, this allows for an efficient implementation of new functionalities, as existing modules can be used whenever feasible. In $ms2$, a stringent philosophy regarding modularization was followed, which is discussed in section 5.1. The runs need to be fast and scale well with increasing number of processors in order to reduce response times. $ms2$ is parallelized with the Message Passing Interface (MPI) [64]. While MD needs synchronization after every time step due to its deterministic nature, MC can be executed embarrassingly parallel, requiring synchronization only at the very end of a run. Furthermore, the by far most expensive part of the code, calculating forces and potential energies, is highly vectorized and optimized.

5.1. Modular structure

Goals.

The simulation program $ms2$ is intended to be flexible, expandable and easy to get familiar with. Still, the program has to be efficient and effective in its calculations, such that the employed computer resources are efficiently used and therefore, the response times are minimized. The introduction of Abstract Data Types (ADT) is a popular approach to reduce the complexity of a software system, decomposing it into smaller subunits, which are easier to maintain. The ADT model, with its data and code abstraction, forms the basis of Object-Oriented

Programming (OOP). It should be mentioned that OOP holds challenges regarding efficiency. Distributing the data between objects can lead to data fragmentation and indirections can occur, leading to a negative impact on the performance. One objective was to introduce OOP concepts, while still maintaining efficiency, cf. section 6.

Fortran issues.
Fortran90 modules are a further development of Fortran77 common blocks. A module may not only contain data, but also define subroutines and user derived data types. Due to the fact that module data and associated module subroutines implement singletons from the start, grouping module-defined user derived data types and associated functions also allows an OOP-like programming style. A small example will demonstrate this. The module ms_simulation, defined in the file ms_simulation.F90, contains the type TSimulation as well as the subroutines

- TSimulation_Construct (this)
- TSimulation_Destruct (this)
- TSimulation_RunSteps (this , StepStart , StepEnd)

among others. The strict naming convention helps avoiding clashes in name space. All the subroutines above receive a reference to an instance of TSimulation through the first parameter, named $this$, and they serve as (default) constructor, destructor or ordinary member function, respectively. In the following, constructs like $this$ will be referred to as a class. $ms2$ does not strictly follow an OOP approach, because all data elements can be accessed directly, therefore there is no information hiding through data encapsulation.

Class structure.
In order to keep the class hierarchy flat, $ms2$ does not make use of inheritance and hence polymorphism. E.g. the module ms2_site contains the classes TSiteLJ126, TSiteCharge, TSiteDipole and TSiteQuadrupole, which have data and functionalities in common, but are not derived from a common class. The class structure and the relations between the classes are organized in six levels, as depicted in Figure 4. Every class is located on a certain level and hence has data and modules relevant to its level. The six levels are intuitive:

1. Global – all data and functions of global use are handled. Scope: whole code
2. Simulation – setup and control of the simulation. Scope: simulation framework
3. Ensemble – the required ensemble is set up and initialized. Scope: molecular ensemble
4. Component – data and functions dealing with all molecules of a given molecule type. Scope: one molecule species
5. Molecule – data regarding the basic structure of a molecule. Scope: one molecule
6. Site – position of sites with respect to a molecule. Scope: one site of one molecule

All subroutines on every level are task specific and implemented as modules. Depending on the simulation setup, e.g. MD/MC or ensemble, the appropriate modules are engaged individually during simulation. The modules of $ms2$ can coarsely be divided into following tasks:

- initialization
- simulation
- accumulation
- output

If a task spans more than one level, the corresponding module on the topmost level is called, which in turn calls modules on lower hierarchical levels and so on.

First level - global. The first level contains subroutines and variables that are needed globally in all parts of the simulation. The variables are mainly natural constants like the Boltzmann constant k_{B}, the Avogadro constant N_A etc. Additionally, variables defining the simulation setup for the given run are stored, e.g. the simulation technique and the ensemble type. These variables determine the type of calculation and are assigned according to user specifications to the levels below in $ms2$. Subroutines that are stored on this level are also of global use in $ms2$. These are basic routines for input into the simulation program and output into files, as well as more advanced routines for automated communication between hardware and program via signals. The latter routines are of particular interest for running $ms2$ on high performance computers, where possible data loss can be avoided by automated communication between cluster nodes and the program.

Second level - simulation. On the second level, the simulation flow is controlled. The simulation is initiated, started and sustained. This includes controlling the length of the simulation and its output. User specifications concerning the simulation setup are read and the corresponding global variables are assigned accordingly. Furthermore, all simulation quantities and averages are analyzed and written to file.

Third level - ensemble. On the third level, the simulation is organized according to the specified ensemble. The respective ensemble is initialized, by characterizing the ensemble setup and defining e.g. the volume and the composition of the molecular species in the simulation volume. In addition, the initial positions and velocities of all molecules are distributed. Furthermore, the routines for the subsequent simulation are assigned on this level. The time integration or the Markov chain, respectively, is performed and the ensemble averages of thermodynamic properties are calculated during the simulation run.

Fourth level - component. On this level, the molecule species (component) and their interactions are addressed. The class structure has three distinct branches. While the first branch deals with issues regarding structure and properties of one component at a time, the second branch deals with interactions between molecules of one or two components. The third branch contains all routines for the accumulation of data, determining the average value of the data as well as their statistical uncertainties. The first two branches will be discussed in more detail. The first branch deals with the structure of one individual component. An instance of the class component stores center of mass position, orientation, velocity etc. of every molecule of one species, e.g. H_2O. The contained modules deal with calculations of all molecules of this species, e.g. modules for solving Newton's equations of

motion, kinetic energy calculation, atom to molecule transformation and vice versa. Furthermore, this branch is responsible for the initialization. It reads the molecular model for every component, calculates positions, introduces the reduced quantities etc.

The second branch deals with interactions between molecules of the same or different type. Therefore, it contains all force and potential energy calculations, which are the computationally most expensive parts by far. This branch is highly vectorized to achieve a fast calculation, in particular if executed in vector-parallel. The modules in this branch are simulation technique specific, i.e. MD or MC. In a MD simulation, first, the interaction partners of every molecule are determined, i.e. other molecules or sites within the cut-off radius. Then the interaction energies and virial contributions as well as the resulting forces and torques are calculated and summed up for every molecule.

In a MC simulation, the modules are designed to calculate only the energy and the virial. Forces are not evaluated and the corresponding algorithms are therefore entirely omitted in these modules. For MD, all interactions of one component-component pair are calculated in one call of the module. For MC, the modules are finer-grained, calculating all interactions of one component-component pair by determining every molecule-molecule interaction individually. The corresponding modules are thus called more frequently than in case of MD.

Fifth level - molecule. The basic information on a molecular model is stored on this level, such as mass, moment of inertia tensor, rotational degrees of freedom etc. Higher level modules access these data via the class molecule.

Sixth level - site. Here, the assembly of a molecule is stored. The data are used by higher level modules to determine the site positions from the molecular positions and orientations. This allows for the calculation of site-site interactions. It should be noted that $ms2$ only integrates (MD) or accepts (MC) center of mass positions and orientations. Site specific data are calculated for every simulation step on the fly.

5.2. Parallelism

The source code is implemented for an efficient parallelism with distributed memory, using the MPI [64] standard for communication. The program uses the following MPI calls:

- MPI_Init, MPI_Finalize, MPI_Comm_rank, MPI_Comm_size to set up basic MPI functionality
- MPI_Abort to stop the program in case of an error
- MPI_Barrier, MPI_Wtime and MPI_Wtick within the stopwatch class
- MPI_Bcast, MPI_Reduce, MPI_Allreduce for simulation data exchange

Exclusively collective communication is used, an approach suitable for molecular simulations, where the cut-off radius is in the order of the simulation volume edge length. For MD simulations with $ms2$, only the force calculation is parallelized using molecule decomposition according to Plimpton [14]. The interaction matrix is rearranged such that the number of interaction partners is almost equally distributed in the matrix. This is achieved by calculating the interactions between molecules 1 and N as interactions between molecules N and

1, cf. Figure 5. The parallelization scheme then distributes the calculation among N_P processors with an almost equal work load, such that every processor executes N/N_P rows of the interaction matrix. The master process reduces then all the force components to sum up the resulting molecular forces on each molecule.
The MC code is parallelized by exploiting the stochastic nature of the simulation method. Starting from an equilibrated state in the simulation volume, the molecular configuration is copied multiple times into different volumes. Then, each copy runs independently in parallel, using a different random number seed, to calculate the thermodynamic properties. At the end, the data from all copies are gathered and averaged.

5.3. Vector-based structure

The program $ms2$ specifically accounts for vector-parallel machines. All the main information needed in the computationally costly inner loops of the calculation, like the positions of the molecules and their sites, are stored in vectors that are accessed sequentially. This allows for an efficient use of memory on vector-parallel machines.
For the evaluation of one configuration in $ms2$, e.g. the position vectors are loaded once and then distributed among all processors. Then, additional information like interaction partners of each individual molecule is computed, tabulated in vectors and communicated. The order of this data follows the order of all other vectors, e.g. the position vector. The appropriate order allows for a serial access to the data in all vectors in the random access memory. Therefore, the calculations can be vectorized, increasing the efficiency.

6. Benchmarks

A subset of the $ms2$ examples, which come along with the code release, was used for profiling and runtime tests that are presented here. The number of time steps (MD) or loops (MC), respectively, considered was lower than for a normal production run, since only comparisons were carried out. The MD and MC test case simulates the equimolar liquid mixture of methanol and ethanol at 298.15 K and 0.1 MPa in the NpT ensemble [65]. Methanol was modeled by two LJ sites and three point charges and ethanol by three LJ sites and three point charges [66]. This test case was chosen, because it is a typical application. Similar results are expected for a wide class of problems. However, note that the actual run times will differ with varying thermodynamic conditions that are simulated. Run times increase with higher density of the system, among others.

6.1. Sequential version

Compiler. $ms2$ is distributed as a source code and can be compiled with a wide variety of compilers. However, the performance is significantly influenced by the compiler and linker as well as the used options. Figure 6 shows the runtime of the test case on different platforms using different compilers. The binaries were generated by

- GNU gfortran[2] with ”-fdefault-real-8 -O3”
- Intel ifortran[3] with ”-r8 -fast”

[2] `http://gcc.gnu.org/fortran/`
[3] `http://software.intel.com/en-us/intel-compilers/`

- PGI pgf95[4] with "-r8 -fastsse"
- Sun Studio sunf90[5] with "-r8const -fast"
- Pathscale pathf90[6] with "-r8 -Ofast"

with options activated to enforce the use of double precision floating point numbers. The chosen optimization flags represent a common choice.
The best combination of compiler and platform was the Intel ifortran compiler and the Intel Xeon E5560 "Nehalem" processor[7]. An Intel ifortran compiled binary of $ms2$ significantly outperforms binaries compiled by GNU gfortran and PGI pgf90, independent on the computing platform.

Profiling. The test case was chosen for profiling studies running the sequential version of $ms2$ with valgrind using the callgrind tool[8]. The binary was built by the Intel ifortran 11.1 compiler and optimized for SSSE3 to work with valgrind $3.5.0$. From the resulting estimated CPU cycles, the computational hot spots are presented for both MD (Figure 7a) and MC (Figure 7b) simulations. The calculation of forces and energies consumes most of the CPU cycles. For the MD run, 54.33% of the CPU time is spent for charge-charge interactions and 36.09% for LJ-LJ interactions, which adds up to 90.42%. Further program routines, e.g. determining the interaction partners (7.21%), add up to a total of 99.70% of the CPU time spent for the entire calculation of potential energies and forces. In MC calculations, 95.68% of the time is spent for energy calculations, including the routine for determining the interaction partners (4.28%).
With valgrind set up to use a 32768 Byte, 4-way associative level 1 data cache with 64 Byte cachelines and prefetch enabled, the cache miss rate for reading (D1r) was below 5%, cf. Table 2. The misses occur mainly in the force calculation for MD (99.19%) and the energy calculation for MC (99.07%).
The ratio of read and write access rates is larger than six, therefore, level 1 reading cache misses (D1mr) have a larger impact on the performance than level 1 writing cache misses (D1mw) and level 2 cache misses. Instruction cache misses are negligible here.

6.2. Vectorization

$ms2$ MD simulations also run efficiently on vector machines like the NEC SX architectures. Table 3a shows profiling data running the MD test case on a NEC SX8R using "ftrace". The force calculation routines reached an average vector length of about 170 and 1892 MFLOPS, which is good, especially considering that the test case is not favorable regarding vectorization. The MC algorithm, however, is not well suited for vectorization, cf. Table 3b. The NEC SX8 shows comparable results, proportional to the lower processor clock speed, whereas results for the NEC SX9 are below expectation with respect to the higher hardware peak performance and memory access speed. For comparison, the same test case was executed on a mainstream Intel Xeon E5440 "Harpertown"

[4] http://www.pgroup.com/products/
[5] http://developers.sun.com/sunstudio/
[6] http://www.pathscale.com/
[7] http://www.hlrs.de/systems/platforms/nec-nehalem-cluster/
[8] http://valgrind.org/info/tools.html#callgrind

2.83 GHz system, reading hardware counters with PAPI. The MD version achieved an average of 1461 MFLOPS and the MC version 1374 MFLOPS on this platform.

6.3. Parallelization

$ms2$ supports parallel systems with distributed memory using the MPI standard for communication. Taking a look at the MPI routines used to exchange data, excluding the ms2_global and ms2_stopwatch modules, solely three different MPI calls are employed: MPI_Bcast (42x), MPI_Reduce (13x) and MPI_Allreduce (12x). As a result of the profiling in section 6.1, only the force calculations were parallelized for MD and therefore a master process has to reduce all force components to sum up the resulting molecular forces. MC calculations use this kind of parallelization only in the equilibration phase. During production, the phase space is sampled with fully independent random walks on each processing unit, generating independent Markov chains, which have to be consolidated only once at the end of the simulation.

The $ms2$ parallelization capabilities were tested on a "Nehalem" dual-Quadcore-CPU cluster with Infiniband, using the Intel ifortran compiler and OpenMPI V1.4 as well as Intel MPI V4.0. The test case runtime results for a range of commonly used numbers of processors are shown in Figure 8. Using more than a single node, i.e. 8 processes here, Intel MPI showed a better scalability than OpenMPI. While the MD scaling behavior is good for a decent number of processors, MC production steps can be characterized as optimally parallel. In contrast to this, the MC equilibration steps, parallelized using a different strategy, show the expected scaling characteristics of MC simulations. A closer look at the time spent for communication shows the differences between equilibration and production for a parallel MC run with 32 processors. After the equilibration, which consumes roughly 23 seconds in this case, no communication takes place, cf. Figure 9a.

For MD, MPI communication is necessary throughout the whole simulation, with communication phases in each time step. Figure 9b shows the communication needed for a single time step, where the barrier induced by the first collective call results in a waiting time for faster processes and particularly for the last process, which might be notably faster than the others, cf. Figure 9c. This imbalance of the last process is because the force calculation for N molecules is divided among the N_p processes, where each processor is assigned with $\lceil N/N_p \rceil$ molecules, except the last one to which the rest is assigned. Regarding the MD test case with 1372 molecules in Figure 9, 31 processes each calculate the forces for 43 molecules, while the last process only deals with 39 molecules. This load imbalance is not a serious issue, since also with a different molecule distribution, where the number of molecules on each processor only differs by one, there is still a processor with 43 molecules, which dictates the overall execution time. The fraction of the time spent in MPI routines increased to about 10% for 32 processes from 5% for 16 processes for the test case. Within the collective communication, process 0 is the master and therefore exhibits a different communication pattern, cf. Figure 9.

6.4. Memory

The memory demand of $ms2$ is low, e.g. for a simulation of 1372 molecules with five interaction sites, roughly 200 MB of RAM is required. For parallel execution, the concept of distributed memory is used. Here, the

memory scales linearly with the number of processors that are used. For most applications of $ms2$ a total RAM of 2 GB is sufficient.

6.5. Comparison to other codes

Monte-Carlo. The performance of $ms2$ was evaluated against the simulation program $MCCCS\ Towhee\ V6.2.7$ [67], which is widely in use and well accepted by the scientific thermodynamic community. In a first step, the comparison was restricted to a simple NVT simulation of the test case. $ms2$ and $MCCCS\ Towhee$, both compiled with the Intel ifortran compiler, executed the MC test case sequentially, i.e. with one process only, on a "Nehalem"/Infiniband cluster with an equal number of MC moves. A sequential run was executed since $MCCCS\ Towhee$ is not yet prepared for parallelization. The performance of $ms2$ was faster than the $MCCCS\ Towhee$ program for this simulation by a factor of around 20.

An important application of $ms2$ is the determination of VLE data. The present comparison was restricted to two other programs: $MCCCS\ Towhee$ [67] and the $Errington$ Gibbs ensemble Monte-Carlo code [68]. Both codes use the Gibbs ensemble approach for the determination of VLE data, a simulation scenario that is not yet supported by $ms2$. Instead, the grand equilibrium method was used in $ms2$ for determining the VLE.

The test case was again a mixture of methanol and ethanol at $T = 393$ K, using N_l = 1372 molecules for the liquid phase and N_v = 500 molecules for the gas phase. These numbers were also used as starting values for the Gibbs ensemble calculations. For $MCCCS\ Towhee$ and the $Errington$ Code, three different types of moves were allowed, translational and rotational displacement, volume exchange and molecule transition between the two simulation volumes. The moves were chosen with probabilities of 79%, 1% and 20% respectively, which are typical numbers for simulations including dense liquid phases. In $ms2$, the number of moves was restricted to two in the liquid phase, translational and rotational displacements (N_l times) and one volume fluctuation each loop. For the gas phase simulation, a move for insertion and deletion of molecules was additionally invoked twice in each loop. The overall execution time was set to 96 hours execution time, which is about the maximal time that is currently provided for a single simulation in high performance computing centers. The runs were performed sequentially, since $MCCCS\ Towhee$ and the $Errington$ code are not suited for parallel execution. For this test case, $ms2$ showed the best performance. Within the 96 h, $ms2$ calculated 52000 loops in the liquid phase simulation, and 50000 loops in the vapor phase simulation. The $Errington$ code performed second best in this test, running 22000 loops, followed by Towhee, which ran 6000 loops.

The performance of the codes influenced the quality of the simulation results drastically. For the VLE test case, the reference phase equilibrium data taken from the literature comprise a vapor pressure of 0.58 MPa, a saturated liquid density of 17.22 mol/l and a saturated vapor density of 0.19 mol/l. $ms2$ reproduced the experimental results well in the given time. The calculated vapor pressure of $ms2$ was 0.53 MPa, while the saturated liquid density of the mixture was predicted to be 17.54 mol/l and the saturated vapor density to be 0.18 mol/l. The statistical uncertainties for the quantities was acceptably low, being 0.01 MPa for the vapor pressure and 0.03 mol/l and 0.003 mol/l for the saturated liquid density and saturated vapor density, respectively. The $Errington$ code calculated in the given time a vapor pressure of 0.19 MPa well as a saturated liquid density of 19.93 mol/l and

a vapor density of 0.443 mol/l. Using $Towhee$, the calculation yielded a vapor pressure of 1.81 MPa as well as densities of 13.62 mol/l and 0.64 mol/l, respectively. Here, the standard deviations of the results were significantly higher. These simulation results showed a drift, e.g. for the saturated liquid density by an average of 1.50 mol/l over 1000 steps. The results for the $Errington$ Code and $MCCCS\ Towhee$ indicate that the simulations with these programs had not reached the phase equilibrium in 96 h. This effect is much more pronounced for $MCCCS\ Towhee$, having executed significantly less loops than the $Errington$ code.

Molecular dynamics. The performance of the MD part of $ms2$ was evaluated against the simulation program $Gromacs\ V4.0.3$ [69], which is designed for simulations of biological systems. The comparison was restricted to a simple NVT simulation of the equimolar liquid mixture of ethanol and methanol at $T = 298$ K and $p = 0.1$ MPa. The runs were executed sequentially on a "Harpertown"/Infiniband cluster with an equal number of time steps and the same interaction cut-off radius of 21 Å. Both programs, $ms2$ and $Gromacs$, were compiled with the Intel ifortran compiler. $Gromacs$ was faster by almost a factor of two than $ms2$ and also the scaling behavior for the parallel version was superior. This shows that there is still an optimization potential in $ms2$, which will be exploited in the future.
A comparison between $Gromacs$ and $ms2$ for the VLE test case was not carried out, since $Gromacs$ does not support VLE simulations.

6.6. Computational demand for the calculation of transport properties

The computational demand for transport properties following the Green-Kubo formalism was evaluated. This investigation was performed for the test case equimolar liquid mixture of methanol and ethanol at a temperature $T = 298$ K and $p = 0.1$ MPa with an autocorrelation length of 13.82 ps. The time period between two autocorrelation functions was specified to be 197.4 fs. A total of 100 autocorrelation functions was explicitly written to file in order to check the results. The transport properties were evaluated every four autocorrelation functions. The reference case was a MD run performed with the same technical details, except for the calculation of the autocorrelation functions. Calculating transport properties increased the CPU time of a simulation step by about 78% compared to the reference case. Note that all autocorrelation functions were evaluated each time step, which is an extreme frequency that can be reduced without much loss of accuracy.

7. Features

The feature programs described in the following are intended to facilitate the handling of $ms2$. They should allow for an easy start of molecular simulations with $ms2$ and give access to all output data generated by $ms2$.

7.1. Simulation setup: $ms2par$

The GUI-based feature tool $ms2par$ allows for a convenient generation of input (*.par) files according to user specifications, cf. Figure 10. It assigns the molecular model, ensemble, thermodynamic state point, time step, number of equilibration steps etc. Furthermore, the tool proposes a cut-off radius based on the input data. After the user completes the specifications, the program generates the *.par-file in ASCII format, to be read by $ms2$.

Applying $ms2par$ is simple and should be intuitive even for new users. $ms2par$ is a java application, thus it runs on all operating systems.

7.2. Simulation analysis: $ms2chart$

$ms2chart$ is a GUI-based java applet for evaluating the simulation results from $ms2$, cf. Figure 11. This tool plots the evolution of properties calculated by $ms2$. The properties can be shown for different graph axes that are chosen from a drop down menu and the plot is shown directly in the GUI. For a better analysis, $ms2chart$ allows various features: plotting block averages as well as simulation averages into the same plot, changing the design of the plot, individual labeling of the axes and adjusting the frame detail. All plots can be exported in png format.

The analysis program $ms2chart$ can be executed at any time of the simulation, i.e. also while the simulation is still running. There is no loss of data, if it is run on the fly. The tool $ms2chart$ is easy to understand and can be handled intuitively. It allows for an easy first analysis of the $ms2$ simulation results. Users will employ $ms2chart$ for a quick check of their simulation results, such as whether the equilibration time was appropriate, which is important if extensive series of simulation runs are performed.

7.3. Visualization tool: $ms2molecules$

The program $ms2molecules$ is a visualization tool for $ms2$. The program displays the molecular trajectories that are stored by $ms2$ as a series of configurations in the *.vim file. $ms2molecules$ visualizes molecular sites by colored spheres. The colors are user defined. The size of a sphere is by default proportional to the LJ size parameter σ, but can be changed manually in the first lines of the *.vim file. This feature facilitates monitoring a component of interest by reducing the size of other components, e.g. solvents. Other features like zooming into or out of the simulation volume as well as rotating the simulation volume also facilitate the analysis of simulations. The visualization can be exported via snapshots in the jpg format.

The program is based on OpenGL and written in C. The handling of the program is simple and console based. Requirements for the feature tool are OpenGL in a Windows or Linux environment. Figure 12 shows a snapshot of a ternary mixture, taken with the program $ms2molecules$, which is convenient to gain insight into the trajectory of the system or the state of the system, respectively.

8. Conclusions

The molecular simulation program $ms2$ was designed for the calculation of thermodynamic properties of bulk fluids. Special care was given to a minimization of the response time. The capabilities of $ms2$ are broad, ranging from basic static thermodynamic properties, like thermal and caloric data, over vapor-liquid equilibria to transport properties, like diffusion coefficient, viscosity and thermal conductivity. These data are accessible for pure substances and mixtures. The accuracy of the simulation data generated by $ms2$ is high, while consuming a reasonable computational effort. Molecular models of more than 100 pure fluids are supplied with $ms2$ that accurately describe their thermodynamic properties. Despite the fact that $ms2$ is a sophisticated Fortran90 program, new developers benefit from its modular structure and object-orientation. The application of $ms2$ is

straightforward, because the input files are well structured and auxiliary feature tools help to create input files, analyze simulation results and visualize molecular trajectories. The code was optimized for the current hardware technology and achieves a high efficiency.

Ongoing efforts focus on the implementation of new algorithms that extend the applicability of the program as well as its analysis tools. The two major current developments are the implementation of internal degrees of freedom to allow for the application of $ms2$ to larger molecules and Ewald summation to allow for the application of $ms2$ to charged molecules. The source code is available to the scientific community at `http://www.ms-2.de`.

9. Acknowledgments

The authors gratefully acknowledge financial support by the BMBF ”01H08013A - Innovative HPC-Methoden und Einsatz für hochskalierbare Molekulare Simulation” and computational support by the Steinbruch Centre for Computing under the grant LAMO and the High Performance Computing Center Stuttgart (HLRS) under the grant MMHBF. The present research was conducted under the auspices of the Boltzmann-Zuse Society of Computational Molecular Engineering (BZS).

Table 1: Computational demand for the calculation of the chemical potential for a three-center LJ fluid using Widom's test molecule insertion in MD or MC simulation. The calculations were performed on a single processor. The simulation time per time step (MD) was 0.043 s and 0.300 s per loop (MC), respectively. The simulations were performed with 500 molecules at a density of 0.23 mol/l. A cut-off radius of 60 Å was employed.

# test molecules	Additional CPU time per MD time step or MC loop / s
1	0.0015
500	0.1925
1000	0.3865
1500	0.5810
2000	0.7750

Table 2: $ms2$ data cache behavior for the test case investigated with callgrind (D: data, 1m: level 1 cache misses, r/w: read/write access). The data misses are normalized by the total amount of data access.

	D1mr/Dr [%]	D1mw/Dw [%]	D2mr/Dr [%]	D2mw/Dw [%]	Dr/Dw
MD	2.41	0.34	0.0001	0.0002	6.18
MC	4.98	7.48	0.006	2.04	10.35

Table 3: NEC SX8R ”ftrace” profiling. The five routines with the largest computational demand and their vectorization for the test case are shown for MD (a) and MC (b).

(a)

PROG.UNIT EXCLUSIVE TIME[sec]	(%)	AVER.TIME [msec]	FREQUENCY MOPS	MFLOPS	V.OP RATIO	AVER. V.LEN	VECTOR TIME	I-CACHE MISS	O-CACHE MISS	BANK CONF CPU	 NETWORK
ms2_potential.tpotcc_force			126900								
598.058	(53.4)	4.713	8661.0	2164.7	99.53	169.3	579.369	0.012	3.383	3.487	13.433
ms2_potential.tpotljlj_force			89300								
373.605	(33.4)	4.184	8039.8	1804.6	99.60	170.0	360.055	0.010	2.775	2.317	8.112
ms2_interaction.tinteraction_force			14100								
118.367	(10.6)	8.395	1749.6	429.2	80.68	172.6	36.326	0.025	39.243	0.092	0.961
ms2_interaction.tinteraction_calcpartners			14100								
27.200	(2.4)	1.929	14262.1	3412.8	99.28	191.2	25.795	0.021	0.091	0.169	0.830
ms2_component.tcomponent_restartsave			2								
0.302	(0.0)	151.159	341.9	7.7	0.14	3.3	0.003	0.048	0.010	0.000	0.001
total			648179								
1119.044	(100.0)	1.726	7855.1	1891.6	99.09	170.5	1002.287	0.265	45.647	6.157	23.568

(b)

PROG.UNIT EXCLUSIVE TIME[sec]	(%)	AVER.TIME [msec]	FREQUENCY MOPS	MFLOPS	V.OP RATIO	AVER. V.LEN	VECTOR TIME	I-CACHE MISS	O-CACHE MISS	BANK CONF CPU	 NETWORK
ms2_interaction.tinteraction_energy			10155544								
5055.675	(98.3)	0.498	2388.1	563.9	69.23	162.3	635.244	22.679	827.141	6.305	45.971
ms2_component.tcomponent_mol2atom1			5604792								
25.066	(0.5)	0.004	305.0	41.3	0.00	0.0	0.000	3.765	7.560	0.001	0.453
ms2_ensemble.tensemble_energy1			3430000								
13.135	(0.3)	0.004	1271.2	492.5	87.92	237.8	2.329	1.990	4.316	0.001	0.163
ms2_ensemble.tensemble_move			1714383								
9.818	(0.2)	0.006	144.8	7.1	7.24	3.0	0.395	2.692	2.778	0.000	0.216
ms2_ensemble.tensemble_rotate			1715617								
8.658	(0.2)	0.005	141.9	9.9	0.00	0.0	0.000	2.491	2.313	0.000	0.199
total			44555359								
5142.345	(100.0)	0.115	2366.6	559.1	69.33	162.7	646.942	38.509	849.895	6.401	47.877

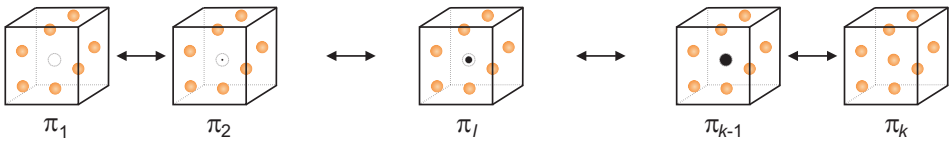


Figure 1: Schematic of the gradual insertion of a molecule. The molecule (dark gray) in state π_l fluctuates between state $l = 0$ (fully decoupled) and state $l = k$ (fully coupled).

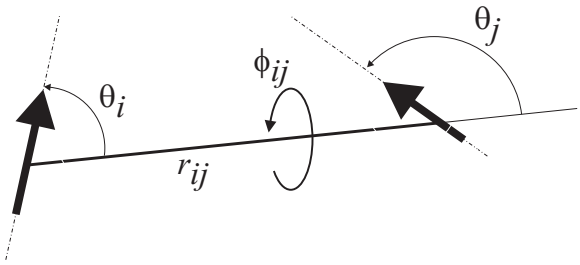


Figure 2: Schematic of the angles between two point dipoles i and j indicated by the arrows, which are situated on different molecules at a distance r_{ij}.

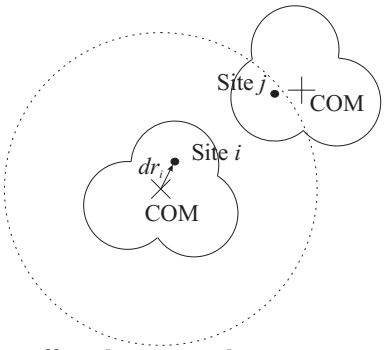


Figure 3: Schematic of the different cut-off modes. Using the site-site cut-off mode, the interaction between the two sites marked by a dot is explicitly evaluated, since they are within the cut-off radius indicated by the dotted line. The distance between the centers of mass (COM), marked by a cross, is irrelevant. Using the center of mass cut-off mode, none of the site-site interactions of the two molecules are explicitly evaluated, since the centers of mass are not within the cut-off radius. The distance between the center of mass and one particular site of molecule i is defined as dr_i.

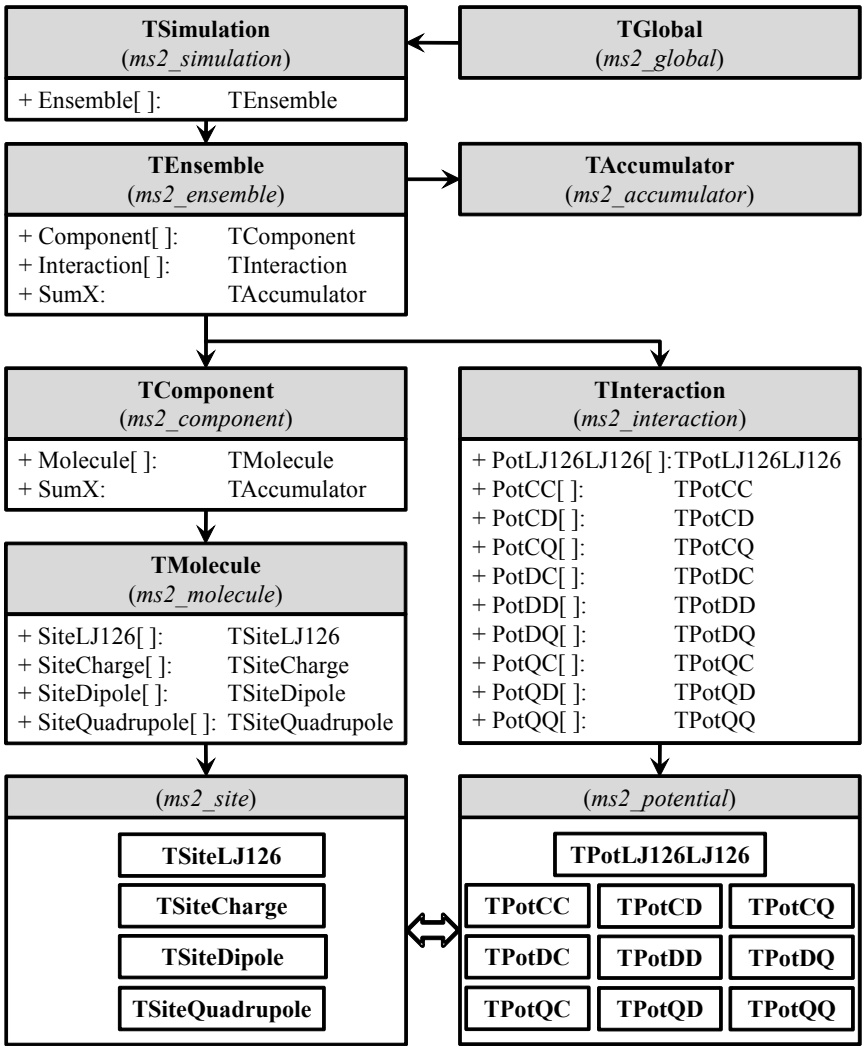


Figure 4: UML class diagram of $ms2$, showing object attributes. The bold headers specify the class name, while the italic names correspond to the program files, the class is implemented. The arrows indicate the command structure, while the bold arrows indicate information transfer. For a better understanding of the structure, following abbreviations are introduced into the diagram: CC: ChargeCharge; CD: ChargeDipole; CQ: ChargeQuadrupole; DC: DipoleCharge; DD: DipoleDipole; DQ: DipoleQuadrupole; QC: QuadrupoleCharge; QD: QuadrupoleDipole; QQ: QuadrupoleQuadrupole

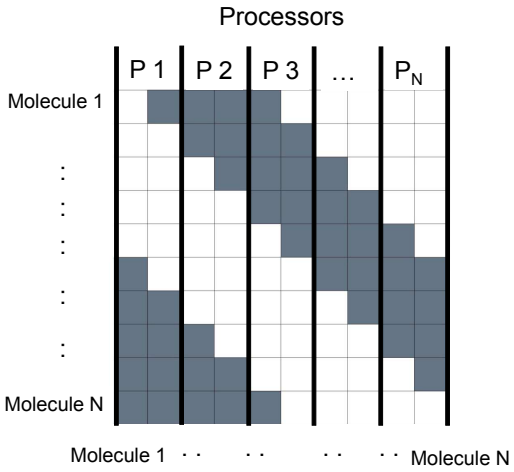


Figure 5: Schematic of the molecule decomposition parallelization method by Plimpton [14]. The colored boxes signify the pair interactions that need to be determined in order to proceed the simulation. The black lines define the range of molecules each processor is assigned to calculate their interactions.

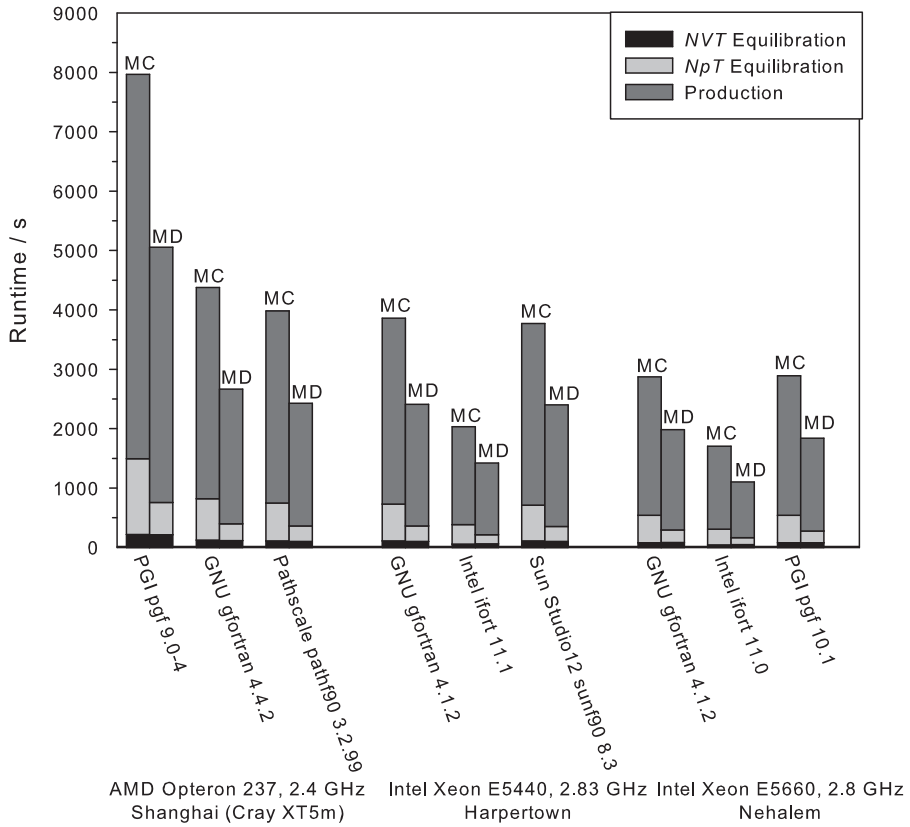
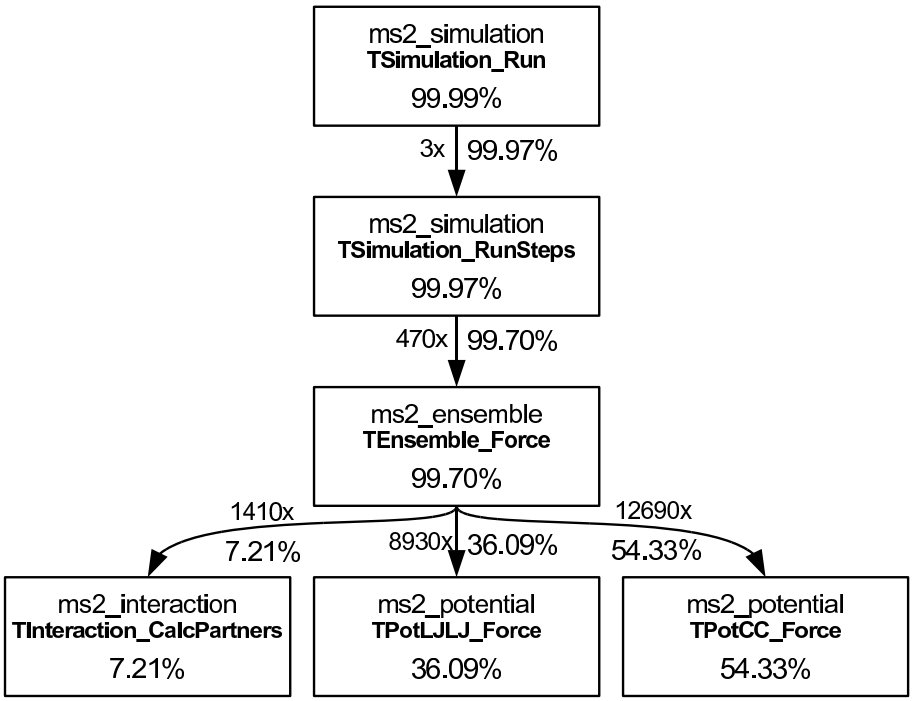
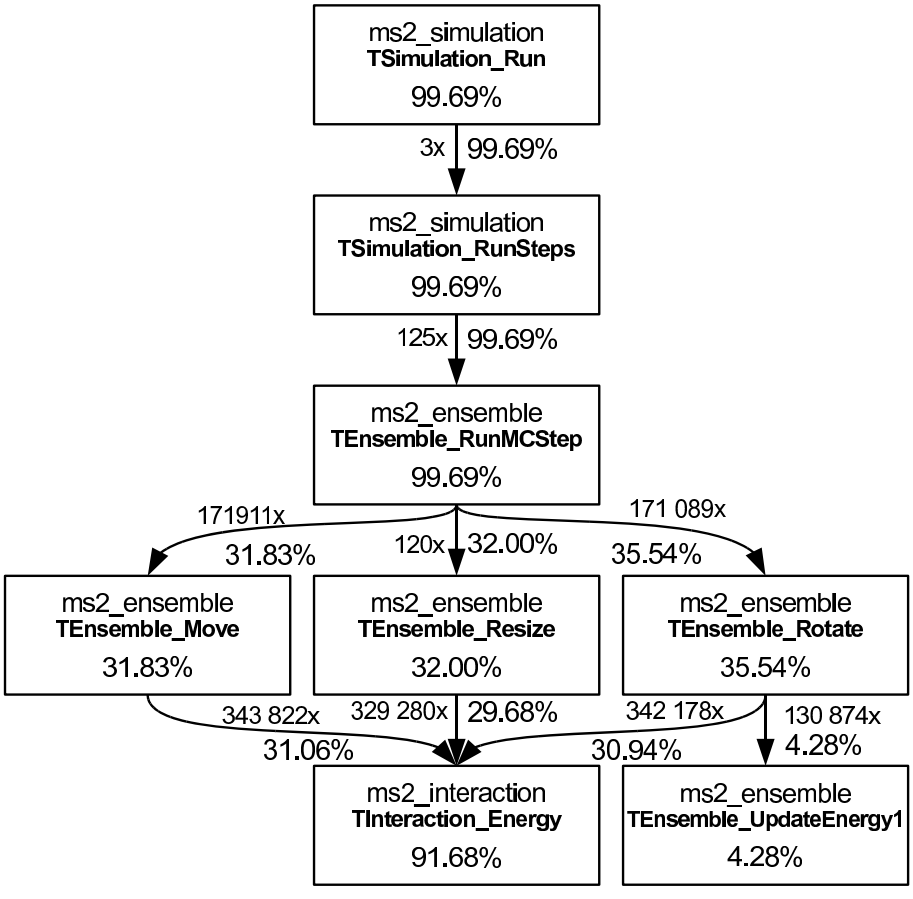


Figure 6: Runtime for the equimolar liquid mixture of methanol and ethanol at 298.15 K and 0.1 MPa in the NpT ensemble using different compilers and processors. The simulation was executed with 1372 molecules with MD for 200 NVT equilibration time steps, 500 NpT equilibration time steps and 4000 production time steps. For MC, 50 NVT equilibration loops, 200 NpT equilibration loops and 1000 production loops.

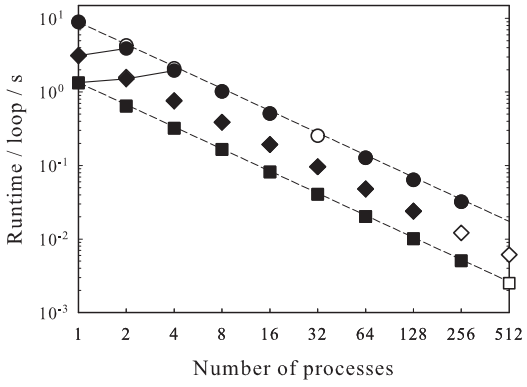


(a)

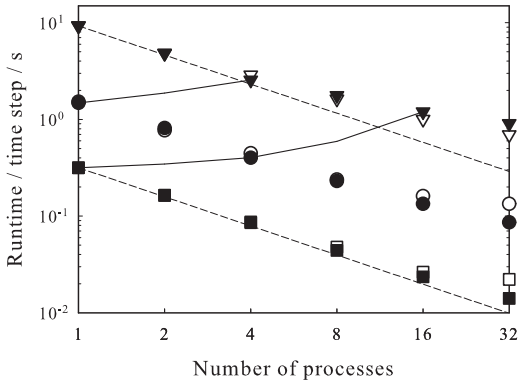


(b)

Figure 7: CPU cycles for one MD step (a) and one MC loop (b) of the test case estimated by valgrind. All functions with less than 1% CPU time share are not shown.

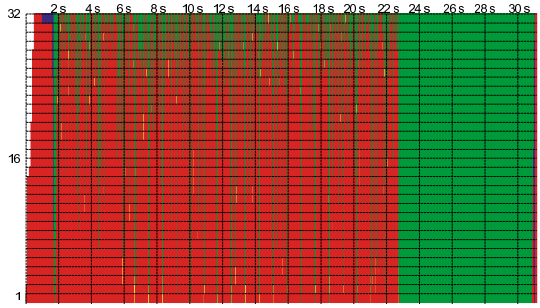


(a)

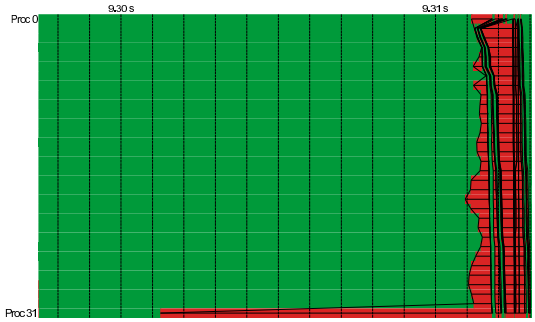


(b)

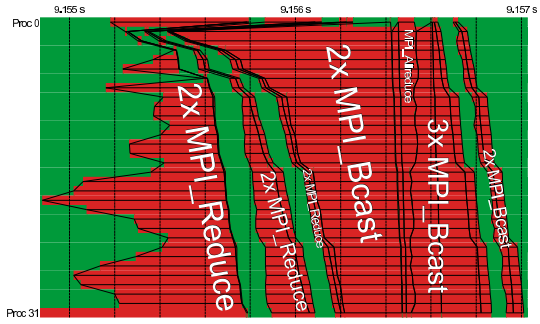
Figure 8: $ms2$ runtime of the test case with MC (a) and MD (b) on a "Nehalem"/Infiniband cluster for Amdahl and Gustafson scaling. Full symbols: Intel MPI V4.0, empty symbols: OpenMPI V1.4.1, □: N=1372, ◇: N=2744, ○: N=5488, ▽: N=21952. The dashed lines indicate perfect strong scaling, while the full lines indicate the achieved weak scaling.



(a)



(b)



(c)

Figure 9: Communication between 32 processes executing the test case. The green color indicates calculations, the red color indicates communication. Figure (a) shows the summary time line for MC simulation, while the lower figures show the summary time line for MD simulation. Figure (b) exhibits the communication for one entire MD step, while (c) gives a magnified view on the communication pattern at the end of the time step.

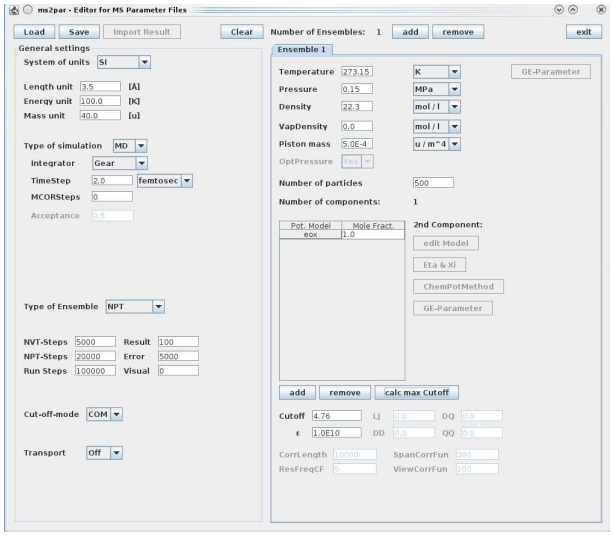


Figure 10: Snapshot of $ms2par$ generating an input (*.par) file for a MD simulation of pure liquid ethylene oxide.

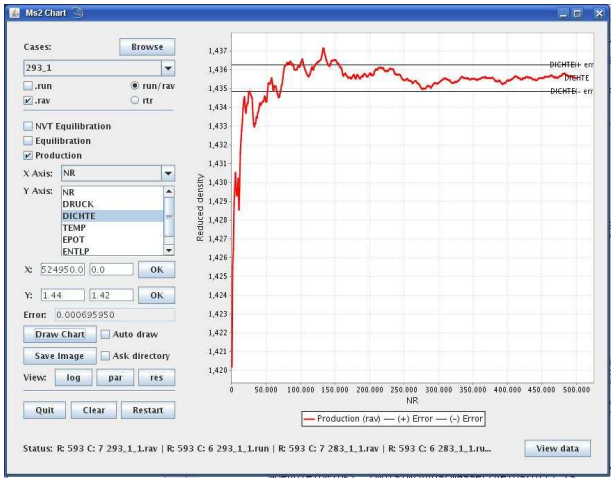


Figure 11: Snapshot of the tool $ms2chart$. The picture shows the analysis of a simulation of liquid cyclohexane.

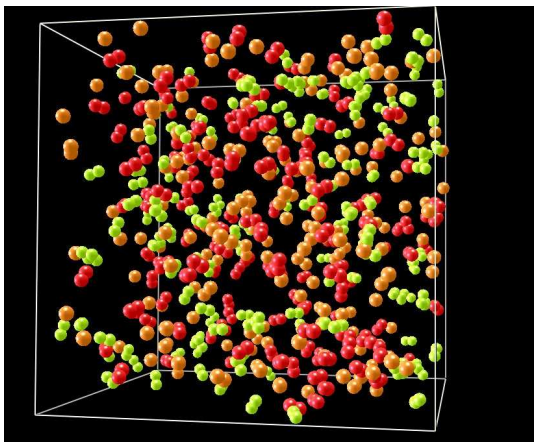


Figure 12: Snapshot of a ternary mixture taken with the visualization tool $ms2molecules$. The mixture consists of ethane (red), methane (orange), and carbon dioxide (green). The white frame illustrates the size of the simulation volume.

References

[1] http://www.fluidproperties.org, Industrial fluid properties simulation collective, 2010.

[2] Gubbins, K. E. and Moore, J. D., The Journal of Chemical Physics **49** (2010) 3026.

[3] Tan, S. P., Adidharma, H., and Radosz, M., Industrial & Engineering Chemistry Research **47** (2008) 8063.

[4] Kleiner, M., Turnakaka, F., and Sadowski, G., Molecular Therodynamics of Complex Systems **131** (2009) 75.

[5] Klamt, A., The Journal of Physical Chemistry **99** (1995) 2224.

[6] Klamt, A. and Eckert, F., Fluid Phase Equilibria **172** (2000) 43.

[7] Gupta, S. and Olson, J., Industrial & Engineering Chemistry Research **42** (2003) 6359.

[8] Case, F. H. et al., Fluid Phase Equilibria **274** (2008) 2.

[9] Case, F. et al., Fluid Phase Equilibria **217** (2004) 1.

[10] Case, F. et al., Fluid Phase Equilibria **236** (2005) 1.

[11] Case, F. H. et al., Fluid Phase Equilibria **260** (2007) 153.

[12] Case, F. H. et al., Fluid Phase Equilibria **285** (2009) 1.

[13] Lyubartsev, A. P. and Laaksonen, A., Computer Physics Communications **128** (2000) 565.

[14] Plimpton, S., The Journal of Computational Physics **117** (1993) 1.

[15] The Economist **March 13th** (2010) 68.

[16] Green, M., The Journal of Chemical Physics **22** (1954) 398.

[17] Kubo, R., The Journal of The Physical Society of Japan **12** (1957) 570.

[18] Poncela, A., Rubio, A. M., and Freire, J. J., Molecular Physics **91** (1997) 189.

[19] Kristof, T., Vorholz, J., Liszi, J., Rumpf, B., and Maurer, G., Molecular Physics **97** (1999) 1129.

[20] Ketko, M. H., Rafferty, J., Siepmann, J. I., and Potooff, J. J., Fluid Phase Equilibria **274** (2008) 44.

[21] Bourasseau, E., Haboudou, M., Boutin, A., Fuchs, A. H., and Ungerer, P., The Journal of Chemical Physics **118** (2003) 3020.

[22] Eckl, B., Vrabec, J., and Hasse, H., Chemie Ingenieur Technik **80** (2008) 25.

[23] Eckl, B., Vrabec, J., and Hasse, H., Molecular Physics **106** (2008) 1039.

[24] Eckl, B., Vrabec, J., and Hasse, H., Fluid Phase Equilibria **274** (2008) 16.

[25] Huang, Y.-L., Vrabec, J., and Hasse, H., Fluid Phase Equilibria **287** (2009) 62.

[26] Eckl, B., Horsch, M., Vrabec, J., and Hasse, H., High Performance Computing In Science And Engineering '08 (2009) 119.

[27] Merker, T., Guevara-Carrion, G., Vrabec, J., and Hasse, H., High Performance Computing In Science And Engineering '08 (2009) 529.

[28] Vrabec, J., Huang, Y.-L., and Hasse, H., Fluid Phase Equilibria **279** (2009) 120.

[29] Allen, M. and Tildesley, D., *Computer Simulation of Liquids*, Clarendon Press, Oxford, 1987.

[30] Vrabec, J. and Hasse, H., Molecular Physics **100** (2002) 3375.

[31] Widom, B., The Journal of Chemical Physics **39** (1963) 2808.

[32] Lyubartsev, A. P., Martinovski, A. A., Shevkunov, S. V., and Vorontsov-Velyaminov, P. N., The Journal of Chemical Physics **96** (1992) 1776.

[33] Nezbeda, I. and Kolafa, J., Molecular Simulation **5** (1991) 391.

[34] Frenkel, D. and Smith, B., *Understanding Molecular Simulation*, Academic Press, Elsevier, San Diego, 1993.

[35] Rappaport, D., *The Art of Molecular Dynamics Simulation*, Cambridge University Press, Cambridge, 2004.

[36] Born, M. and von Karman, T., Physikalische Zeitschrift **13** (1912) 297.

[37] Metropolis, N., Rosenbluth, A., Rosenbluth, M. N., Teller, A. H., and Teller, E., The Journal of Chemical Physics **21** (1953) 1087.

[38] Rowley, R. et al., *DIPPR Data Compilation of Pure Compound Properties*, Design Institute for Physical Properties, AIChE, 2003.

[39] Flyvberg, H. and Petersen, H., The Journal of Chemical Physics **91** (1989) 461.

[40] Lyubartsev, A., Laaksonen, A., and Vorontsov-Velyaminov, P., Molecular Physics **82** (1994) 455.

[41] Vrabec, J., Kettler, M., and Hasse, H., Chemical Physics Letters **356** (2002) 431.

[42] Shing, K., Gubbins, K., and Lucas, K., Molecular Physics **65** (1988) 1235.

[43] Gray, C. and Gubbins, K., *Theory of molecular fluids, Volume 1: Fundamentals*, Clarendon Press, Oxford, 1984.

[44] Gubbins, K., *Statistical Mechanics*, volume 1, The Chemical Society Burlington House, London, 1972.

[45] Krishna, R. and van Baten, J. M., Industrial & Engineering Chemistry Research **44** (2005) 6939.

[46] Hoheisel, C., Physics Reports **245** (1994) 111.

[47] Barton, A., *The dynamic liquid state*, Longman, London, 1974.

[48] Steele, W., *Transport Phenomena in fluids*, Marcel Dekker, New York, 1969.

[49] Lennard-Jones, J., Proceedings of the Physical Society **43** (1931) 461.

[50] Berthelot, D., Comptes Rendues Hebdomadaires des Seances de l'Academie des Sciences **126** (1898) 1703.

[51] Lorentz, H., Annalen der Physik (1881) 127.

[52] Fischer, J., Möller, D., Chialvo, A., and Halle, J. M., Fluid Phase Equilibria **48** (1989) 161.

[53] Stone, A. J., Science **321** (2008) 787.

[54] Hirschfelder, J., Curtiss, C., and Bird, R., *Molecular Theory of Gases and Liquids*, J. Wiley & Sons, 1954.

[55] Nymand, T. and Linse, P., The Journal of Chemical Physics **112** (2000) 6152.

[56] Steinhauser, O., Molecular Physics **45** (1982) 335.

[57] van der Spoel, D., van Maaren, P., and Berendsen, H., The Journal of Chemical Physics **108** (1998) 10220.

[58] Ewald, P., Annalen der Physik **64** (1921) 253.

[59] Lisal, M., Budinsky, R., and Vacek, V., Fluid Phase Equilibria **135** (1997) 193.

[60] Garzon, B., Lago, S., Vega, C., and Rull, L., The Journal of Chemical Physics **102** (1995) 7204.

[61] Lisal, M., William, W., and Nezbeda, I., Fluid Phase Equilibria **181** (2001) 127.

[62] Jedlovszky, P. and Mezei, M., The Journal of Chemical Physics **110** (1999) 2991.

[63] Lustig, R., Molecular Physics **65** (1988) 175.

[64] MPI-Forum, *MPI: A Message-Passing Interface Standard, Version 2.2*, High Performance Computing Center Stuttgart (HLRS), 2009.

[65] Guevara-Carrion, G., Nieto-Draghi, C., Vrabec, J., and Hasse, H., The Journal of Physical Chemistry B **112** (2008) 16664.

[66] Schnabel, T., Srivastava, A., Vrabec, J., and Hasse, H., The Journal of Physical Chemistry B **111** (2007) 9871.

[67] Towhee, http://www.towhee.sourceforge.org, 2008.

[68] Errington, J. and Panagiotopoulos, A. Z., http://kea.princeton.edu/jerring/gcmc/index.html.

[69] Hess, B., Kutzner, C., van der Spoel, D., and Lindahl, E., The Journal of Chemical Theory and Computation **4** (2008) 435.