Distributed Monte Carlo Simulations of Liquid Crystalline Phases

Roberto Berardi, Sigismondo Boschi and Claudio Zannoni
Dipartimento di Chimica Fisica ed Inorganica
Università degli Studi
viale Risorgimento 4, 40136 Bologna, Italy

Computer simulations of liquid crystals, although not nearly as abundant as simulations of ordinary liquids have started to become an important way of studying this fascinating category of fluids [1]. The main characteristic of liquid crystals and what makes them particularly difficult to study and simulate is the existence of orientational and possibly positional order and the presence of transitions between phases where some of these order parameters vanish. Our group has studied two [2] and three dimensional [3] systems of uniaxial molecules with orientational and translational degrees of freedom interacting via the Gay–Berne potential (GB) [4] using canonical Monte Carlo (MC) simulations. In the last few years this model has become very popular since it is able to give rise to a number of anisotropic phases, namely nematic, smectic A and B, observed in real liquid crystals. The GB model describes the anisotropic interaction between two uniaxial particles i and j, which we can think as ellipsoids of revolution, with orientations given by unit vectors $\mathbf{u}_i$ and $\mathbf{u}_j$ and with centers separated by the intermolecular vector $\mathbf{r}$. The interaction energy is given by

$$U_{GB}(\mathbf{u}_i,\mathbf{u}_j,\mathbf{r}) = 4\epsilon(\mathbf{u}_i,\mathbf{u}_j,\hat{\mathbf{r}})\left[\left\{\frac{\sigma_s}{r-\sigma(\mathbf{u}_i,\mathbf{u}_j,\hat{\mathbf{r}})+\sigma_s}\right\}^{12} - \left\{\frac{\sigma_s}{r-\sigma(\mathbf{u}_i,\mathbf{u}_j,\hat{\mathbf{r}})+\sigma_s}\right\}^{6}\right]$$

where $\hat{\mathbf{r}}$ is a unit vector along the intermolecular vector. This potential has an attractive and repulsive part decreasing with separation r as r^{-6} and r^{-12} but with a rather complex form that depends on the particle orientations as well as the intermolecular vector [4, 2]. The parameterization of the potential is rather complex but basically it can be related to shape and attractive energy anisotropy [4]. The aim of simulating thermotropic liquid crystalline phases is usually to determine the temperature dependence of the thermodynamic observables, compute configurational properties such as pair correlations, order parameters, graphical renderings of the molecular organization etc. and to study transitions between the ordered phases formed by the molecules.

We have developed a FORTRAN code based on the *"Parallel Virtual Machine"* (PVM) [5, 6] software for the MC simulation of a fluid phase formed by particles interacting with the Gay–Berne anisotropic potential and we present here some results obtained from simulations of both two– [2] and three–dimensional [3] samples of GB particles. Although the specific results are reported for these liquid crystal models, the strategy and findings should be of general applicability in many MC studies.

The general problem in MC simulations of systems presenting phase transitions is related to the requirement of using a large number of particles. This, together with the need of extremely long equilibration runs near a phase transition, causes heavy computer time requirements. Furthermore the computation of the physical observables needed to compare the results of the simulation with experimental data can become the heaviest task of the whole computation (often more than 50%).

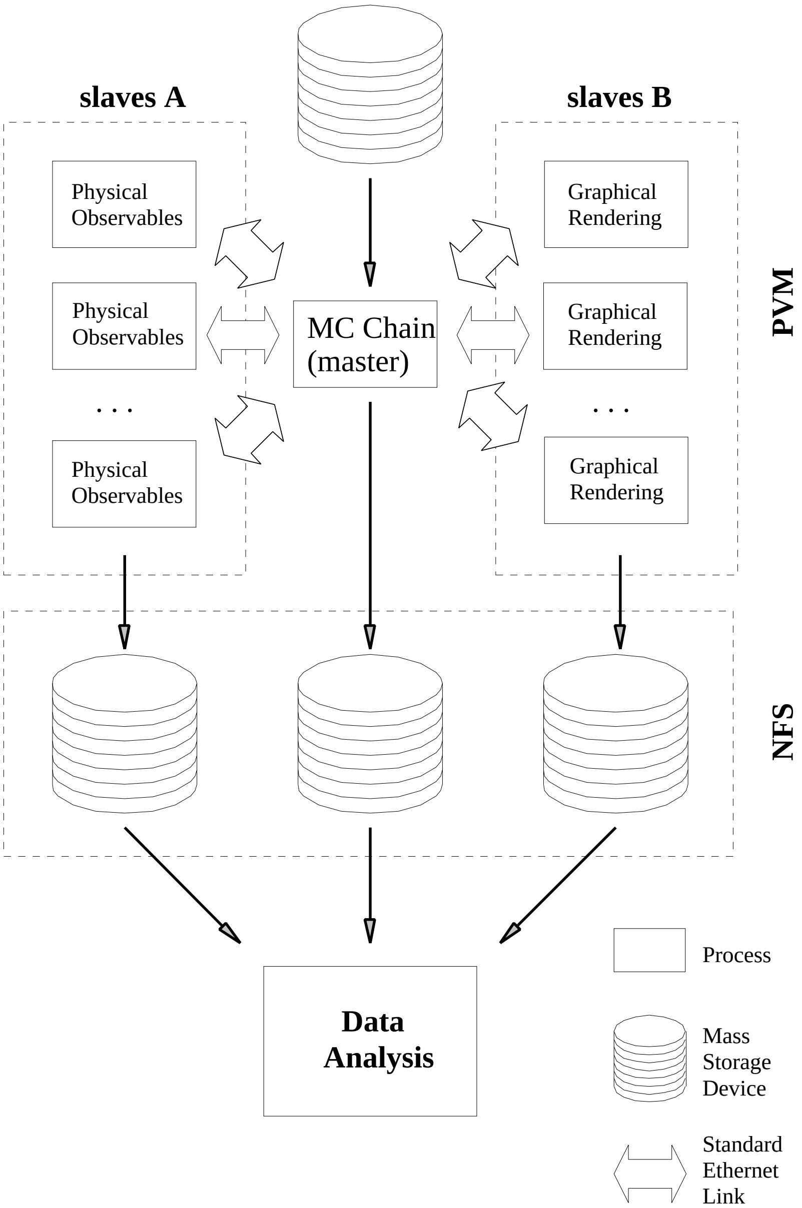

Chart showing the simple master–slave PVM architecture with networked workstations and NFS shared disk space used in this work.

MC programs are intrinsically difficult to parallelize and the first problem in converting a standard MC code for parallel computing is dividing the job in different tasks while minimizing the communication overhead. For instance using neighbour lists [7] the CPU time of the MC evolution for a GB sample (without added electrostatic interactions requiring Ewald sums) grows almost linearly with the number N of particles while the CPU time for the computation of the physical observables and pair properties increases with N^2. Thus we have chosen to split the simulations in two tasks: the evolution of the MC chain and the computation of the thermodynamic and structural properties. Since this latter part of the simulation mainly reduces to determining frequency histograms of pair properties it is possible to distribute this computation on more than one process, each one analyzing a different configuration. The global average of the local histograms can then be performed at the end of the MC run with a very small computational effort. We have adopted a master–slave architecture with the master process executing the core of the MC evolution and a set of slave processes computing the configurational observables (slaves A in the PVM chart). In addition we have introduced a second kind of slave processes (slaves B) as a C written program used for the computation of the graphical rendering of the molecular organization using a shading algorithm.

In a typical MC simulation the physical observables are computed with fixed frequency usually every ten or twenty evolution cycles, depending on the size of the sample, to guarantee statistical independence. When executed under PVM the master MC process sends with this given frequency the sample configuration (i.e. positions and orientations of all the particles) of the MC sample to the first available slave (A for the observables and B for the rendering). Apart from synchronization signals the slave processes do not need any other data to perform their task and in practice this amounts to transfering a few kbytes of data. The overhead due to the network communication is in this case negligible. If all the slaves are still computing the master process skips the current iteration and continues the MC evolution. In practice since the amount of CPU time required to compute the observables is considerably higher than that required to perform a MC cycle, the configuration of the virtual machine can be chosen in order to maximize the frequency of computation of the observables within the available computational resources. If these resources and the number of slaves are sufficient to acknowledge every call from the master process the balancing of the load on the slave processes is automatically controlled since slaves running on slower machines are inevitably called less frequently. The overall result is that the time needed to perform the whole simulation is determined by the process running the MC evolution. The virtual machine can then be configured in order to gain speed in the simulation or to run at "zero cost" part of the simulation on slower machines.

The first example of these PVM applications comes from a series of detailed MC simulations of two dimensional systems with $N = 256 - 4096$ uniaxial particles interacting via the Gay–Berne GB potential. [2] The simulations of the bigger samples with $N = 4096$ run on two HP735/99 entirely devoted to this purpose for almost 3.5×10^6 cycles (more than 5000 CPU hours). The virtual machine was configured with the MC evolution running on one CPU and a process computing of the observables running on the other. The two processes were synchronized in order to completely use both CPU's and avoid idle periods. We could perform very long runs ($1 \times 10^5 - 5 \times 10^5$ cycles) needed for the equilibration and study the temperature dependence of the two dimensional phases formed by these systems.

The configuration of the virtual machine can be changed in order to fulfill the constraints given by the size of the MC sample as we show with this second example. We are developing a generalization of the Gay–Berne potential to biaxial ellipsoids [8] and we are running short test simulations on three dimensional samples with different sizes and densities. In one of these simulations with a sample of $N = 1024$ biaxial particles we have used a virtual machine formed by two workstations where the MC process runs on an HP735/99 (2.2 s of CPU time per cycle) and calls the slave process running on an HP720 (9.3 s of CPU time per cycle) every 20 cycles. Using a sample of $N = 4096$ we have used five workstations with an HP735/99 (10.8 s of CPU per cycle) updating the MC sample and calling one of the four slave processes for the computation of the observables every 10 cycles. Two slave processes run on two HP720 (144.5 and 173.0 s of CPU per cycle each) and the remaining two run on two HP735/99 (65.1 and 64.3 s of CPU per cycle each). On average on every workstation other two extra processes were running at the time these tests were performed. The different CPU times observed for the same slave processes running on identical workstations is a consequence of their different load when these tests were run.

As a final example, in the following table we report some early results [9] we obtained running this MC code on a virtual machine with different configurations.

PVM			MC evol. (Master)		Observables (Slave 1)		Rendering (Slave 2)		max
M	S1	S2	CPU/ks	cycle CPU/s	CPU/ks	call CPU/s	CPU/ks	call CPU/s	CPU/ks
[A]	—	—	1.61	3.22±0.88	0.41	8.16±0.12	0.45	44.91±1.21	$2.47^{(a)}$
[B]	—	—	0.70	1.39±0.42	0.18	3.78±0.05	0.16	16.21±1.01	$1.04^{(a)}$
[A]	[A]	[A]	1.62	3.25±0.88	0.40	8.06±0.06	0.45	44.52±1.31	1.62
[A]	[A]	[B]	1.61	3.22±0.89	0.40	8.08±0.14	0.16	16.00±0.80	1.61
[A]	[B]	[A]	1.62	3.24±0.89	0.18	3.87±0.63	0.45	44.52±1.24	1.62
[B]	[A]	[A]	0.69	1.38±0.40	0.40	8.10±0.21	0.45	45.21±1.17	0.69
[C]	—	—	0.67	1.34±0.15	0.25	4.92±0.01	0.41	40.56±0.21	$1.33^{(a,b)}$
[D]	—	—	0.59	1.19±0.06	0.11	2.28±0.03	—	—	$0.70^{(a)}$

where [A] ≡ HP720, [B] ≡ HP735/99, [C] ≡ CRAY–YMP *e* [D] ≡ DEC ALPHA–500.
the CPU times in ks are for Monte Carlo runs 500 *cycles long.*
The thermodynamic observables were computed every 10 cycles.
The graphic rendering was performed every 50 cycles.
$^{(a)}$ *global CPU time: all processes ran on the same machine.*
$^{(b)}$ *CRAY—YMP single processor.*

For these tests we have used three HP720 and one HP735, a cluster of eight DEC-ALPHA 500 workstation at CASPUR (Rome) and a CRAY-YMP at CINECA (Bologna). Even in this case every workstation on the virtual machine had a typical (two extra processes) CPU loading. The MC trial runs were of 500 cycles. The computation of physical observables was performed every 10 cycles while the graphical rendering every 50. For every process we report the type of CPU running it, the average CPU time for every cycle (call) as well as the turnaround CPU time spent. The last column reports the maximum amount of CPU used by a single processor of the virtual machine and represents the global time needed by the PVM application to be completed.

We thank CNR and MURST for financial support, CASPUR for the use of their DEC–ALPHA cluster and CINECA for an allocation of CPU time on the CRAY–C90.

References

[1] G.R. Luckhurst and C.A. Veracini, Editors, *"The Molecular Dynamics of Liquid Crystals"*, NATO ASI, Series C, Vol. 431, Kluwer Academic Publishers, (1994).

[2] R. Berardi, S. Boschi and C. Zannoni, *to be published*, (1994).

[3] R. Berardi, A.P.J. Emerson and C. Zannoni, *"Monte Carlo Investigations of a Gay–Berne Liquid Crystal"*, *J. Chem. Soc. Faraday Trans.*, **89**, 4069, (1993).

[4] J.G. Gay and B.J. Berne, *J. Chem. Phys.*, **74**, 3316, (1981).

[5] J. Dongarra, G.A. Geist, R. Mancheck and V.S. Sunderam, *Computers in Physics*, **7**, 166, (1993).

[6] A. Geist, A. Beguelin, J. Dongarra, W. Jiang, R. Mancheck and V. Sunderam, *PVM 3.3 User's Guide and Reference Manual*, Oak Ridge National Laboratory, TN (USA), May 1994, *and references therein.*

[7] M.P. Allen and D.J. Tildesley, *"Computer Simulation of Liquids"*, Clarendon Press, Oxford, (1987).

[8] R. Berardi, C. Fava and C. Zannoni, *to be published*, (1994).

[9] R. Berardi, S. Boschi and C. Zannoni, *"Monte Carlo Simulations of Anisotropic Phases using Distributed Network Computing"*, 2^o *Convegno Nazionale di Informatica Chimica*, 16–18 february 1994, CINECA, Casalecchio di Reno, Italy.