Accelerated Algorithms 2

B. Dünweg
Max Planck Institute for Polymer Research
Ackermannweg 10
D–55128 Mainz
Germany

Abstract. The problem of critical slowing down is the appearance of a very long (diverging) correlation time associated with a corresponding appearance of a diverging correlation length. This is elucidated by a trivially solvable model, the one–dimensional Gaussian model. The general strategy to fight this effect, in order to sample phase space more efficiently, is to update large length scales with artificially high rates. This approach is exemplified by the methods of Fourier acceleration, and multigrid simulations. For polymer chains, the long correlations arise directly from the molecular connectivity. Some Monte Carlo algorithms which attack this problem again by non–local moves (dimerization, pivot, slithering snake, connectivity–altering methods) are presented.

1. The Problem: Critical Slowing Down

Critical slowing down always occurs when a physical system exhibits objects which, in some sense, can be called "critical clusters". This terminology stems from the theory of critical phenomena; however, the applicability of the concepts and algorithms to be discussed in this chapter goes beyond standard second–order phase transitions. A critical cluster is a large correlated object of typical size ξ (the correlation length), which can be made arbitrarily large by means of some control parameter. Typical examples are magnetic clusters in a spin model (where the control parameter is the temperature distance from the critical point), percolation clusters (whose size is controlled by the occupation fraction), polymer chains and tethered membranes (controlled by the degree of polymerization). Furthermore, the objects typically do not have a well–defined shape, but are rather ramified fractals, and exhibit very many configurations, which are all easily accessible as the typical energy to change (or create / delete) the object is, at most, of order of the thermal excitation energy $k_B T$.

Such inherently soft objects are also inherently slow: The physical dynamics is usually *local*, which means that the object can only be re–arranged if the information about such changes has spread throughout it, and this takes the more time the larger ξ is. For the "classical" case of diffusive dynamics, this implies the scaling law $\tau \propto \xi^2$, where τ is the correlation time which is needed to completely re–arrange the object.

Taking non–diffusive dynamics and the fractal geometry into account, one rather has the more general law $\tau \propto \xi^z$ where z is the dynamical exponent. A power law should hold since the system exhibits only two relevant length scales (the correlation length ξ and the lower length scale cutoff a, like the lattice spacing or the particle size), and similarly only two relevant time scales (τ and the microscopic time related to a), while the behavior in between should be scale–invariant. An overview over the various cases of dynamic critical phenomena at second–order phase transitions can be found in the review article by Halperin and Hohenberg [7].

The large value of z of course poses a problem to computer simulations, since the correlation time determines the statistical accuracy of a simulation (see contribution by A. Milchev). If one is only interested in static properties, one would therefore like to move the system in an *unphysical* way through its configuration space, and to "beat" critical slowing down by deliberately *violating* the condition of locality such that information may spread more quickly. The present lecture intends to outline a few of such strategies for some systems, without attempting to cover the topic thoroughly or comprehensively. It should be noted that critical slowing down is only one mechanism of slowing down which plagues simulations; others are equally important and are discussed in other parts of the school (for "hydrodynamic slowing down" and its elimination by choice of a suitable ensemble, see contributions by K. Binder and N. Wilding; for dealing with low–temperature excitations in discrete systems, see contribution by M. Novotny; for dealing with activation barriers, see contributions by N. Wilding and W. Janke).

In order to understand the strategies to devise algorithms against critical slowing down, let us first discuss the simplest model which exhibits this phenomenon, and which is actually exactly solvable.

2. The Gaussian Model

We start out from the well–known Landau–Ginzburg–Wilson (LGW) Hamiltonian for the Ising universality class:

$$\mathcal{H} = \int d^d\vec{x} \, \left[\frac{1}{2} \left| \nabla \phi \right|^2 + \frac{r}{2} \phi^2 + \frac{u}{4!} \phi^4 \right] . \qquad (1)$$

This is the simplest model for a system with a scalar order parameter ϕ exhibiting a second–order transition. All three terms are essential: The terms ϕ^2 and ϕ^4 describe the competition between order ($\phi \neq 0$) and disorder ($\phi = 0$). In the Mean Field picture, only these terms are considered. While u is constant, r is varied and corresponds to the distance from the critical point: For $r > 0$, $\phi = 0$ is stabilized, while for $r < 0$ the ordered phase prevails. This picture, however, lacks any spatial structure, i. e. the tendency of near neighbors to order in the same way. This last property is built in via the first term, which makes interfaces energetically unfavorable.

From there, we go to the *Gaussian model* by omitting the ϕ^4 term. We thus obtain a quadratic Hamiltonian which is exactly solvable. Of course, the model only makes sense in the disordered phase $r > 0$; the stabilization of the ordered phase is not described. For simplicity, we specialize on the one–dimensional case, and study the lattice discretized version:

$$\beta \mathcal{H} = \frac{1}{2} \sum_{n=0}^{L-1} \left[(\phi_{n+1} - \phi_n)^2 + r \phi_n^2 \right] , \tag{2}$$

where L is the number of sites, $\beta = 1/(k_B T)$, with T the absolute temperature and k_B Boltzmann's constant. An appropriate re–definition of the units of ϕ is implied, and periodic boundary conditions are assumed. The statics is trivially solved by the introduction of Fourier modes via a unitary transformation which diagonalizes $\mathcal{H}$:

$$\tilde{\phi}_p = \frac{1}{\sqrt{L}} \sum_{n=0}^{L-1} \phi_n \exp\left(\frac{2\pi i}{L} pn \right) \tag{3}$$

$$\phi_n = \frac{1}{\sqrt{L}} \sum_{p=0}^{L-1} \tilde{\phi}_p \exp\left(-\frac{2\pi i}{L} pn \right) \tag{4}$$

$$\beta \mathcal{H} = \frac{1}{2} \sum_{p=0}^{L-1} \left[4 \sin^2 \left(\frac{p\pi}{L} \right) + r \right] \left| \tilde{\phi}_p \right|^2 . \tag{5}$$

The equipartition theorem then tells us that

$$\left\langle \left| \tilde{\phi}_p \right|^2 \right\rangle = \frac{1}{4 \sin^2 \left(\frac{p\pi}{L} \right) + r} \approx \frac{1}{4 \left(\frac{p\pi}{L} \right)^2 + r}, \tag{6}$$

which is the well–known Ornstein–Zernike correlation function in Fourier space. The last approximation has been made for long–wavelength modes $p \to 0$. From this, we can directly read off $\xi^{-2} \propto r$, implying the Mean Field value 1/2 for the critical exponent ν. Furthermore, the overall mean square fluctuation of the order parameter, which is proportional to the susceptibility, is obtained as the $p = 0$ mode, for which we can read off $\left\langle \left| \tilde{\phi}_0 \right|^2 \right\rangle = r^{-1}$ or $\gamma = 1$ for the Mean Field susceptibility exponent.

Turning to the dynamics, we assume the Langevin equation of motion (see also the other contribution by the author)

$$\frac{d}{dt}\phi_i = -\frac{1}{\tau_m} \frac{\partial \beta \mathcal{H}}{\partial \phi_i} + f_i \tag{7}$$

with

$$\langle f_i \rangle = 0 \qquad \langle f_i(t) f_j(t') \rangle = 2 \frac{1}{\tau_m} \delta_{ij} \delta(t - t'), \tag{8}$$

and τ_m denoting an elementary local ("microscopic") time scale. This is the generic equation of motion for overdamped local dynamics with non–conserved order parameter (so–called "model A" dynamics [7]), as it corresponds, for example, to a simple single–spin flip Monte Carlo algorithm on a somewhat coarse–grained length and time scale. As the thermodynamic force is just linear in the order parameter, the equation of motion is also diagonalized by introduction of the Fourier modes:

$$\frac{d}{dt}\tilde{\phi}_p = -\frac{1}{\tau_m} \frac{\partial \beta \mathcal{H}}{\partial \tilde{\phi}_p^\star} + f_p \tag{9}$$

where the noise acting on mode p does not only vanish on average, $\langle f_p \rangle = 0$, but is even uncorrelated to the noise acting on other modes:

$$\left\langle f_p(t) f_q^\star(t') \right\rangle = 2 \frac{1}{\tau_m} \delta_{pq} \delta(t - t'). \tag{10}$$

Hence, the modes decay *independently*,

$$\frac{d}{dt}\tilde{\phi}_p = -\frac{1}{\tau_p} \tilde{\phi}_p + f_p \tag{11}$$

such that the correlation function $\left\langle \tilde{\phi}_p^\star(0) \tilde{\phi}_p(t) \right\rangle$ is just an exponential with correlation time

$$\tau_p = \frac{\tau_m}{4 \sin^2\left(\frac{p\pi}{L}\right) + r}. \tag{12}$$

From this, we can read off $z = 2$ in the long–wavelength limit $p \to 0$ via two equivalent considerations: Either we study the relaxation of the overall order parameter $p = 0$ off critically, resulting in $\tau_0 \propto r^{-1} \propto \xi^2$, or we study the nonzero modes at criticality $r = 0$, resulting in $\tau_p \propto p^{-2} \propto \lambda_p^2$, where λ_p is the wavelength associated with mode p.

The important point about this exercise is that it gives us a rough understanding how critical slowing down comes about, even when we consider non–trivial models, which require simulation. The system becomes increasingly soft, the larger the length scale under consideration is. This is expressed by the decreasing thermodynamic force as $p \to 0$. Correspondingly, it also becomes increasingly sluggish, as one sees from the increase of τ for $p \to 0$.

3. Fourier Acceleration

The idea of Fourier acceleration [1] is easy to understand, based on the considerations of the previous section. We start from a system which is described by a field–theoretic LGW Hamiltonian, and simulate this directly on a lattice, by solving the Langevin equation of motion, Eq. 7, for its non–trivial Hamiltonian, using the methods outlined in the other contribution by the author. Suppose we would instead run this in Fourier space. Then we would have to solve Eq. 9 instead. However, for a non–trivial Hamiltonian the interaction in Fourier space is long–ranged, and cumbersome to evaluate. It is therefore much easier to rather evaluate the forces in real space, and Fourier transform these, to give the forces on the modes. We thus arrive at a scheme where we start from a field configuration in real space, evaluate the forces there, then Fourier transform both the fields and the forces, then update the fields in Fourier space, and afterwards Fourier transform the new fields back into real space. For a non–critical system, there would however be no point in doing so; one would only introduce superfluous operations. But for a critical system we can exploit the previously explained sluggishness for $p \to 0$ by noting that the necessary *time step* of a Langevin update is governed by the intrinsic time scale associated with that kind of motion. We hence introduce a *multiplicity of time steps* and update those modes which have a long wavelength with a large time step h_p, and ideally adjust h_p in such a way that $h_p \propto \tau_p$. For the Gaussian model, this procedure eliminates critical slowing down completely. For a non–trivial model, there remains some significant residual slowing down. Here the modes are not completely decoupled, and the concept of a mode relaxation time is only approximately valid. It is often useful to combine these Langevin updates with a Metropolis acceptance criterion in order to correct for discretization errors ("force biased Monte Carlo", see also the other contribution by the author).

An important ingredient of such simulations is to run lattice sizes which are a power of two. In this case, the Fourier transformation can be done in a very efficient way using the so–called Fast Fourier Transform (FFT), whose effort scales as $L \log L$ for system size L. This is a recursive divide–and–conquer method, by which one (large) Fourier transformation is mapped onto two transformations of half the size, etc. For further details, see textbooks on numerical analysis [12].

4. Multigrid Monte Carlo

This approach is rather similar in spirit. We again study a Hamiltonian in terms of continuous variables (a lattice–discretized field theory), and again confine ourselves to a rough sketch of the method, as also explained in the original literature [6]. For simplicity, let us again look at the one–dimensional LGW Hamiltonian,

$$\beta\mathcal{H} = \sum_{n=0}^{L-1} \left[\frac{1}{2} \left(\phi_{n+1} - \phi_n \right)^2 + \frac{1}{2} r \phi_n^2 + \frac{1}{4!} u \phi_n^4 \right] . \tag{13}$$

Quite similar to the idea of real–space renormalization, we introduce a “blocking” transformation by subdividing the lattice into identical blocks, each of which comprises a certain set of sites. On each of these blocks, we introduce block variables ψ just as the arithmetic mean of the original variables ϕ. In our example, where the blocks are just pairs of consecutive sites, we have $M = L/2$ variables $\psi_n = (\phi_{2n} + \phi_{2n+1})/2$. Furthermore, we introduce fluctuation variables π in each block. The precise form is not particularly important; what we require is that the π variables should be linear combinations of the original ϕ variables in the block, and that they, together with the ψ variable, determine the ϕ variables in the block uniquely (if the block comprises k sites, then we have $k - 1$ π variables per block). In our one–dimensional example, a convenient choice is $\pi_n = (\phi_{2n} - \phi_{2n+1})/2$, and the Hamiltonian in terms of the new variables is

$$\begin{aligned} \beta\mathcal{H} &= \sum_{n=0}^{M-1} \left[\frac{1}{2} \left(\psi_{n+1} - \psi_n \right)^2 + h_n(\{\pi_i\}) \psi_n \right. \qquad (14) \\ &+ \left. \frac{1}{2} r_n(\{\pi_i\}) \psi_n^2 + \frac{1}{3!} g_n(\{\pi_i\}) \psi_n^3 + \frac{1}{4!} (2u) \psi_n^4 \right] \\ &+ \beta\mathcal{H}'(\{\pi_i\}), \end{aligned}$$

where we have focused on the ψ dependence. The dependence on the π variables comes through a purely additive term, and the new coupling parameters h_n, r_n, g_n, whose precise form is of course important for the actual implementation, but not for the spirit of the method. It is however very important that the new Hamiltonian is again a fourth–order polynomial, and that the range of interaction in terms of blocks does not exceed the nearest neighbors. These properties remain true if one goes to higher dimensions, and if one iterates the transformation (as in the renormalization group). In contrast to renormalization–group calculations, however, no attempt is being made to integrate out the fluctuation variables (for any interesting Hamiltonian, this could be done only approximately). The fluctuation variables are rather kept, and the transformation is used to update the ψ variables at an unphysically high rate. One thus arrives at a recursive hierarchy of systems, each of which is decomposed into block variables and fluctuation variables, and where the block variables of a certain level are the "raw" variables at the next level, such that the number of degrees of freedom in each level becomes less and less. It is useful to view these systems as separate, such that each has its own region of memory assigned, in which all of its variables ϕ, ψ, and π, as well as the couplings, are stored. Together, the algorithm hence involves an "up" step (corresponding to "coarse–graining"), and a "down" step (corresponding to "fine–graining"). There are various ways how to jump back and forth between the levels, and we shall not discuss these here in detail. The "up" step works as follows: Suppose that at a certain level the ϕ variables and the couplings are given. From these, one calculates the fluctuation variables, plus the block variables and the new couplings. The latter two variables are stored at the system one level higher. Then some standard Monte Carlo procedure is applied on the latter system, such that, in effect, only the block variables are updated, while the fluctuation variables are kept constant. The corresponding "down" step then consists of the transformation back, such that the ϕ variables at the lower level are obtained from the (old) π variables of the lower level, and the (new) ϕ variables of the higher level, which are the block (ψ) variables of the lower level. This is followed by a standard Monte Carlo procedure applied to the lower level. The speedup is essentially due to the fact that rather large moves through phase space are facilitated by updating a comparatively small number of degrees of freedom.

5. Polymer Algorithms

5.1. POLYMER PHYSICS: SOME BASICS

In this subsection, we wish to briefly summarize some important scaling laws in polymer physics [2, 3], which are of direct importance for the construction of efficient simulation algorithms.

Usually, a polymer chain is described as a self–similar random fractal characterized by the law $R \propto N^{\nu}$, where R is the typical size (e. g. the gyration radius or the end–to–end distance), and N is the number of monomers within the chain, or the degree of polymerization. For a single chain immersed in a good solvent, ν has the value ≈ 0.59 in three dimensions. This is larger than the random–walk (RW) value $1/2$, due to the swelling resulting from the excluded–volume

(EV) interaction. $\nu \approx 0.59$ thus corresponds to the statistics of so–called “self–avoiding walks” (SAWs). In a dense melt, however, the EV interaction is screened; here we rather have RW statistics.

Furthermore, it is interesting to ask how many conformations are available to the chain. For RW statistics, this number is just $Z = \mu^N$, where μ can be considered as an effective coordination number of a lattice chain. For SAW statistics, it can be shown that [2]

$$Z_{SAW} = A \mu'^N N^{\gamma - 1}, \tag{15}$$

where (if considering the same lattice) μ' is *smaller* than μ, and the exponent γ has a universal value which depends only on the spatial dimension ($\gamma \approx 1.16$ in three dimensions).

Turning to the dynamics, it is instructive to first discuss the RW Rouse model. Firstly, a RW can be described as a *Gaussian chain*. The distribution of the end–to–end vector is Gaussian, as a result of the central limit theorem. Identifying the Gaussian distribution with the Boltzmann factor, this corresponds to an effective harmonic Hamiltonian. The Gaussian distribution must also hold for subchains, for reasons of self–similarity, and even down to the scale of the so–called Kuhn segments (the shortest subchain for which such a description makes sense). If b is the root mean square length of the segments, and the chain is composed of N segments, then the effective Hamiltonian in d dimensions is

$$\beta \mathcal{H} = \frac{d}{2b^2} \sum_{n=0}^{N-1} \left(\vec{r}_{n+1} - \vec{r}_n \right)^2 . \tag{16}$$

The spatial dimensions are thus statistically independent, and the Hamiltonian of each dimension is obviously mathematically identical to the one–dimensional Gaussian model at criticality. The Rouse model also assumes an overdamped dynamics, and hence the equation of motion is identical to Eq. 7. For this reason (see above), the longest relaxation time scales as $\tau \propto N^2 \propto R^4$.

Starting from the RW Rouse model, one can introduce various modifications, which all give different dynamic universality classes: (i) The inclusion of hydrodynamic interactions (see contribution by A. Ladd) yields the so–called Zimm model, which shall not be discussed further at this point (see Ref. [3]); (ii) taking into account the EV effect, one arrives at the SAW Rouse model; (iii) taking into account the temporary topological constraints (“entanglements”) for dense systems, one obtains the reptation model [2, 3].

For the SAW Rouse model we observe that the chain conformations should relax on the same time scale as is needed for translational motion on the scale of the coil size. Thus, if D is the translational diffusion coefficient of the chain, then $D\tau \propto R^2 \propto N^{2\nu}$. However, in the Rouse model the friction coefficients of the segments just add up, and hence $D \propto 1/N$. For these reasons, $\tau \propto N^{1+2\nu}$.

In the reptation model, it is assumed that the topological constraints enforce a motion which is essentially *curvilinear* on time scales below τ, with a curvilinear diffusion coefficient $D_{curv} \propto 1/N$ (as in the Rouse model). The path which must be followed until a new conformation is attained is proportional to N, and hence $D_{curv}\tau \propto N^2$. Combining these results, we find $\tau \propto N^3$.

Thus, in all cases we find an unfavorable power law dependence of the relaxation time vs. N, *if* the slow physical dynamics is followed. Polymers can therefore be considered as prototypical examples of critical slowing down. Hence, the general approaches outlined in the previous sections should work for polymers, too. Indeed, for 2d SAWs the author has shown that the strategy of Fourier acceleration is applicable [4]. There is however quite a number of algorithms which specifically exploit the chain structure and are more efficient. A few of these shall be presented below; the reader should note that this is just a small selection which by far does not cover the field exhaustively.

5.2. ALGORITHMS FOR SINGLE CHAINS

The simplest approach to sample the conformation statistics of a SAW would be to take a chain and move it through its conformation space by local updates. This, however, corresponds to slow Rouse dynamics ($\tau \propto N^{1+2\nu}$), and hence the range of accessible chain lengths is limited. Alternatives are to either generate new conformations completely "from scratch" (static methods), or to move the chain in a non–local fashion. We will only give a brief sketch; for a much more comprehensive overview the reader is referred to the review by Sokal [13]. Let us first discuss the static methods.

It is immediately obvious that "simple sampling" (i. e. "blind" generation of a RW, which is discarded whenever there is an overlap) cannot work. Indeed, the acceptance probability is given by $p_{acc} = Z_{SAW}/Z_{RW} \propto N^{\gamma-1}\exp(-\text{const.}N)$ and hence exceedingly small for all but very short chains. In contrast, the *dimerization* algorithm [13, 14] is based on a recursive buildup: A SAW of length $N = 2M$ is generated by concatenating two SAWs of length M, i. e. chains for which non–overlap has been checked already before. In case of an overlap, both subchains are discarded, and the procedure is attempted again. The two subchains, in turn, are generated by precisely the same procedure out of even shorter chains, and so forth.

In order to estimate the efficiency of this scheme [13], we note that the acceptance probability of the dimerization step is given by

$$p_{acc} = \frac{Z_{SAW}(N)}{[Z_{SAW}(N/2)]^2} = \text{const.}\, \frac{\mu'^N N^{\gamma-1}}{\mu'^N N^{2\gamma-2}} = \text{const.}\, N^{-(\gamma-1)}, \qquad (17)$$

i. e. a weak power–law decay. From this, we can conclude the following recursion formula for the CPU time to generate one walk:

$$T_{CPU}(N) = 2 T_{CPU}(N/2) \ \text{const.}\, N^{\gamma-1}. \qquad (18)$$

From this, one shows by induction

$$\begin{aligned} T_{CPU}(N) &= \text{const.}\, N^{f(N)} \qquad (19) \\ f(N) &= \frac{\gamma-1}{2} \log_2 N + \text{const.}. \end{aligned}$$

This is a power law modified with a $N^{c \ln N}$ behavior, i. e., for very long chains, more unfavorable than any of the dynamic methods, which scale with pure power laws. However, the power law itself is only a moderate increase in N. Furthermore, the prefactor of the $\log_2 N$ term in $f(N)$ is quite small, such that the logarithmic modification is felt only for quite long chains. For these reasons, dimerization is a quite competitive method for moderate chain lengths up to a few thousand.

Turning to the non–local dynamic methods, let us first consider the simple "slithering–snake" algorithm [13, 15]. The trial move is to randomly select an end monomer, to cut it off, and to attach it to the other end with a random bond orientation. The EV interactions control if this move is successful. In order to compare the scaling of this algorithm with Rouse dynamics, we can estimate the relaxation time τ via $D_{curv} \tau \propto N^2$, as in reptation theory. However, the scaling of D_{curv} is now much more favorable: Firstly, one elementary move shifts the chain by one unit, and secondly, the unit time is given by performing N such elementary moves (note that in Rouse dynamics the computer has to work on N monomers in order to move the chain one time step ahead). Therefore, the mean square curvilinear displacement in one time unit is of order N, such that $D_{curv} \propto N^{+1}$! For these reasons, $\tau \propto N$, i. e. the algorithm is by a factor of $N^{2\nu}$ more efficient than Rouse dynamics. While this is less efficient than the pivot algorithm (see below) for single chains, the slithering–snake algorithm is an excellent choice for (moderately) dense systems, where exactly the same scaling considerations apply. However, one has to take into account that in a dense matrix the acceptance rate is rather small, such that the scaling law is hampered by a rather unfavorable prefactor.

In the so–called "pivot algorithm" [13, 9, 10] a monomer (the "pivot" monomer) and a rotation axis going through that monomer are selected at random, and either the “left” or the “right” tail attempts to rotate by a random angle. This move is successful if permitted by the EV interactions. In another variant, the rotation axis is defined as the line going through two randomly selected monomers.

Let us try to analyze the efficiency of this method [13]. Neglecting the correlations between the two tails of length L and $N - L$, we can estimate the acceptance rate by assuming a concatenation:

$$p_{acc}(L) = \frac{Z_{SAW}(N)}{Z_{SAW}(L) Z_{SAW}(N-L)} = \text{const.} \left[N x (1-x) \right]^{-(\gamma - 1)} \quad (20)$$

with $x = L/N$. Averaging over x yields $p_{acc} \propto N^{-(\gamma-1)}$, as in dimerization. Empirically [13], one rather finds an $N^{-p} \approx N^{-0.11}$ law, but this may perhaps not be the true asymptotic behavior.

The efficiency of the pivot algorithm is due to the fact that the large–scale variables are decorrelated after a small number of steps, which (at least in good approximation) does not depend on N. Unless special tricks are applied [13], one step takes a CPU time of order N, and hence, for comparison with Rouse dynamics, we should identify the unit time with one (attempted) pivot step. Combining this consideration with the acceptance rate, we find $\tau \propto N^{p}$ for the correlation time, which is very efficient. There is, however, a severe caveat: The consideration applies only to large–scale variables, while a *full* decorrelation will only have taken place if the conformations are also relaxed on small scales, i. e. if, on average, each monomer has been visited once as a pivot. This deteriorates the efficiency by another power of N, resulting in $\tau \propto N^{1+p}$. For these reasons, it is often advisable to combine the pivot algorithm with a local Rouse–type dynamics in a hybrid scheme.

5.3. ALGORITHMS FOR DENSE SYSTEMS

In the previous subsection, we have already mentioned the slithering–snake algorithm as an efficient method for dense systems, which is however hampered by a small prefactor. Another (in principle rather old) strategy is to alter the connectivity, i. e. to cut the chains into pieces, and re–connect them in a different way. The most recent versions of these algorithms, with special emphasis on applicability to models with detailed atomistic structure, have been developed by the group of Theodorou and co–workers under the names “end bridging” [11] and “double bridging” [8]. In what follows, we shall attempt to briefly discuss these methods, in a simplified version which disregards the atomistic structure and would be applicable to highly flexible chains.

The end–bridging algorithm randomly selects a monomer at the end of a chain, plus two other monomers (usually not at some chain end) which are (i) directly connected with each other, (ii) close to the end monomer, and (iii) not directly connected with the end monomer. Usually, this pair of monomers will reside on a different chain, but this condition is not necessary. As an attempted Monte Carlo move, this pair is broken, and the end monomer is connected to one of the now free monomers. Care must be taken that the acceptance criterion satisfies the condition of detailed balance (the reaction paths in both directions must be considered explicitly), but this can be done in a straightforward way with some book–keeping. These updates must be combined with some conventional Molecular Dynamics or Monte Carlo moves, since otherwise the monomers would always keep their positions.

The biggest disadvantage of this algorithm is that it does not conserve the chain length distribution, which is rather a result of the algorithm itself. Usually, the procedure is adjusted in order to keep the distribution reasonably narrow. However, it is impossible to simulate a strictly monodisperse system where only one chain length occurs.

Empirically it has been found that this method is very efficient, but this question has not yet been studied systematically in terms of scaling laws. In what follows, we will attempt this analysis; however, the presented results should be viewed as tentative. We assume that the main relaxation mechanism is the diffusion of the "hot spots", which are moved by roughly one monomer unit within one elementary update. Such an update can be viewed as a complete Monte Carlo or Molecular Dynamics cycle throughout the system (to move the monomers), followed by end–bridging attempts wherever they are possible. As the density of "hot spots" is of order $1/N$, the CPU effort of the latter updates can be completely neglected in the long–chain limit. We now envision the "hot spots" as random walkers which scan the system, and assume that the system is relaxed if every monomer has been visited once. Now, three–dimensional random walks have only few overlaps with each other, and also only few self–overlaps. It is therefore reasonable to assume that the whole system has been visited after a time of order N. We thus conclude $\tau \propto N$, as for the slithering–snake algorithm.

In the double–bridging algorithm, the same idea is applied, but the disadvantage of a polydisperse system is removed. Again, the algorithm selects a pair of connected monomers (say, $i - j$). Instead of a reaction with a nearby end monomer k as in end–bridging, it here attempts a reaction with another connected pair $(k - l)$ located nearby. This other pair must not be directly connected to $i - j$. Now, a new connectivity is attempted (either $i - k$ and $j - l$ or $i - l$ and $j - k$). This is always rejected whenever a chain with length $\neq N$ would be created; otherwise the condition of detailed balance must be taken into account.

By construction, monodispersity is conserved. This results either in a rearrangement within one chain, or in an exchange of the tails of two chains.

Again, the scaling behavior of the algorithm is not yet known, and we shall tentatively attempt it here. Obviously, the relaxation is again governed by the "hot spots" (i. e. the local regions where the move can be successful) and their density, but here we have to distinguish between intra–chain and inter–chain contributions. The inter–chain density is proportional to $1/N$, as one can see from the following consideration: For a given bond $i - j$, we can move a neighboring chain with a slithering–snake motion until the condition of tail length fits. From this we see that the probability of a fitting tail length is proportional to $1/N$. For the intra–chain density the length condition is automatically satisfied, and hence it is controlled by the probability of self–overlap of a RW, which, in three dimensions, is proportional to $N^{-1/2}$ (the density of a RW is proportional to $N/R^3 \propto N/N^{3/2} = N^{-1/2}$). The intra–chain density of "hot spots" is therefore proportional to $N^{-1/2}$, i. e. much larger than the inter–chain density. Nevertheless, one cannot expect that the intra–chain moves are able to relax the system fully; this is rather accomplished by the slower inter–chain mechanism. Moreover, in the double–bridging case the "hot spots" are neither conserved nor continuously moving in space, and therefore a diffusion picture does not apply. They are rather created and destroyed at a certain constant rate. Again assuming that the system is relaxed when every monomer has been at a "hot spot", we arrive at $\tau \propto N/r$, where r is the creation / destruction rate, and the factor N takes into account the $1/N$ scaling of the "hot spot" density. Assuming that the rate r is independent of chain length N, one would again arrive at $\tau \propto N$.

However, this latter assumption may be overly optimistic [5]. One has to take into account that τ is actually given by the mean time that a given bond $i-j$ needs to find a reaction partner, and this is governed by the monomer motion. In three dimensions the monomers have to travel a typical distance of order $N^{1/3}$ (the mean distance between "hot spots", as one concludes from the density). If we now assume that the changing of the connectivity does eliminate the reptation–like slowing down, but keeps the Rouse–like slowing down (neither of these assumptions is completely obvious), then the time would be given by $\tau \propto N^{4/3}$ (note that in Rouse dynamics, the mean square displacement of a monomer scales as $t^{1/2}$ [3], resulting in $\tau^{1/4} \propto N^{1/3}$). So this scaling argument indeed produces some slowing down compared to the end–bridging case, which is expected to a certain extent, in view of the monodispersity constraint which is absent for end–bridging. Nevertheless, the definitive resolution of the question of the scaling of these algorithms will probably have to wait until accurate data on the performance have been produced.

Acknowledgments

Stimulating discussions with R. Everaers on Sec. 5.3 are gratefully acknowledged.

References

1. Batrouni, G. G., G. R. Katz, A. S. Kronfeld, G. P. Lepage, B. Svetitsky, and K. G. Wilson: 1985. *Phys. Rev. D* **32**, 2736.
2. deGennes, P. G.: 1979, *Scaling Concepts in Polymer Physics.* Cornell University Press.
3. Doi, M. and S. F. Edwards: 1986, *The Theory of Polymer Dynamics.* Oxford: Oxford University Press.
4. Dünweg, B.: 1994. In: D. P. Landau, K. K. Mon, and H.-B. Schüttler (eds.): *Computer Simulation Studies in Condensed Matter Physics*, Vol. 7. Heidelberg: Springer–Verlag, p. 177.
5. Everaers, R., private communication.
6. Goodman, J. and A. D. Sokal: 1986. *Phys. Rev. Lett.* **56**, 1015.
7. Halperin, B. and P. Hohenberg: 1977. *Rev. Mod. Phys.* **49**, 435.
8. Karayiannis, N. C., V. G. Mavrantzas, and D. N. Theodorou: 2002. *Phys. Rev. Lett.* **88**, 105503.
9. Lal, M.: 1969. *Mol. Phys.* **17**, 57.
10. Madras, N. and A. D. Sokal: 1988. *J. Stat. Phys.* **50**, 109.
11. Mavrantzas, V. G., T. D. Boone, E. Zervopoulou, and D. N. Theodorou: 1999. *Macromolecules* **32**, 5072.
12. Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling: 1988, *Numerical Recipes in C.* Cambridge: Cambridge University Press.
13. Sokal, A. D.: 1995. In: K. Binder (ed.): *Monte Carlo and Molecular Dynamics Simulations in Polymer Science.* Oxford: Oxford University Press, p. 47.
14. Suzuki, K.: 1968. *Bull. Chem. Soc. Japan* **41**, 538.
15. Wall, F. T. and F. Mandel: 1975. *J. Chem. Phys.* **63**, 4592.