SIMULATION OF PHASE COEXISTENCE FOR COMPLEX MOLECULES

Daan Frenkel and Berend Smit

Department Editors:
Harvey Gould
hgould@clarku.edu
Jan Tobochnik
jant@kzoo.edu

The goal of molecular simulation is to predict the macroscopic properties of molecular substances on the basis of our knowledge of the interactions between the molecules. The two computational techniques used in molecular simulation, molecular dynamics and Monte Carlo, have been discussed extensively in this column. In this article we focus on some recent advances in Monte Carlo methods that have greatly increased the predictive power of molecular simulation. We then apply these methods to the simulation of polymers.

Consider a chemist who is planning to make a novel substance. Because synthesis is expensive and simulation is increasingly inexpensive, it is worthwhile to predict the properties of the new substance. Our chemist might wish to know whether the substance will be a vapor, liquid, or solid. Also of interest are the mechanical properties (compressibility of fluids, strength of solids) and the transport properties (viscosity, heat or electrical conductivity). Finally, how will this substance interact (or even react) with other substances?

Daan Frenkel is head of the Condensed Matter Division of the FOM Institute for Atomic and Molecular Physics in Amsterdam, The Netherlands, and professor of computational chemistry at Utrecht University. Mailing address: FOM Institute, Kruislaan 407, 1098 SJ Amsterdam, The Netherlands. E-mail: frenkel@amolf.amolf.nl

Berend Smit is professor of chemical engineering at the University of Amsterdam, Department of Chemical Engineering, Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands. E-mail: smit1@cpcoup5.tn.tudelft.nl

Surprisingly, the question about the phase behavior is by no means the easiest to answer. In the following, we briefly explain why it is difficult to predict phase behavior by simulation and discuss the recent progress that has been made. Rather than giving a broad overview, we focus on one specific method, namely *configurational-bias Monte Carlo*. Our goal is to explain how this method works and why it is useful.

Simulation is in many ways similar to experiment. We must prepare a sample, that is, generate a well equilibrated configuration of the system under the desired conditions. Next, we perform a measurement during a certain time interval. As in an experiment, the resulting numerical data are subject to statistical and systematic errors. The analogy between experiment and simulation suggests that to determine a phase diagram in a simulation, we should find the temperatures and pressures where one phase spontaneously transforms into another. However, for first-order phase transitions, this approach usually fails because hysteresis (superheating and supercooling) is a serious problem in simulations. To resolve this problem, we consider what coexistence means. We say that two phases coexist if they can be simultaneously present in equilibrium. Before equilibrium has been established, there can be a net flow of heat or mass from one phase to the other, and the volume of one phase can increase at the expense of the other. In equilibrium, all such transient effects have ceased.

The second law of thermodynamics allows us to deduce the conditions for phase coexistence from the absence of nonequilibrium fluxes. The absence of heat flow implies that the temperature T of the two phases must be equal. The absence of volume changes implies that the pressure P of the two phases is the same. The absence of mass exchange implies equality of the chemical potential μ in the coexisting phases. These conditions help us to determine the values of μ where two phases coexist as a function of P and T without bringing the phases in physical contact. The set of points where the chemical potentials are equal define the coexistence curve.

The problem is that, whereas it is easy to measure (or even impose) P and T in a simulation, special techniques are required to measure (or impose) μ.[1] However, for phase coexistence, we are not interested in the absolute value of μ, and we require only that μ is the same in both phases. This idea was first exploited by Thanasis Panagiotopoulos,[2] who proposed the Gibbs-ensemble technique to study liquid–vapor coexistence. In such simulations, we consider two systems with periodic boundary conditions (thus mim-

 S0894-1866(97)00503-8

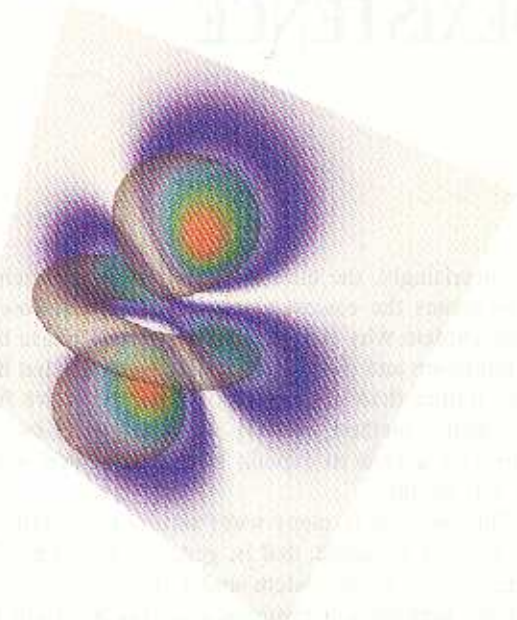

Figure 6. Probability density for the 3,2,0 state of the hydrogen atom.

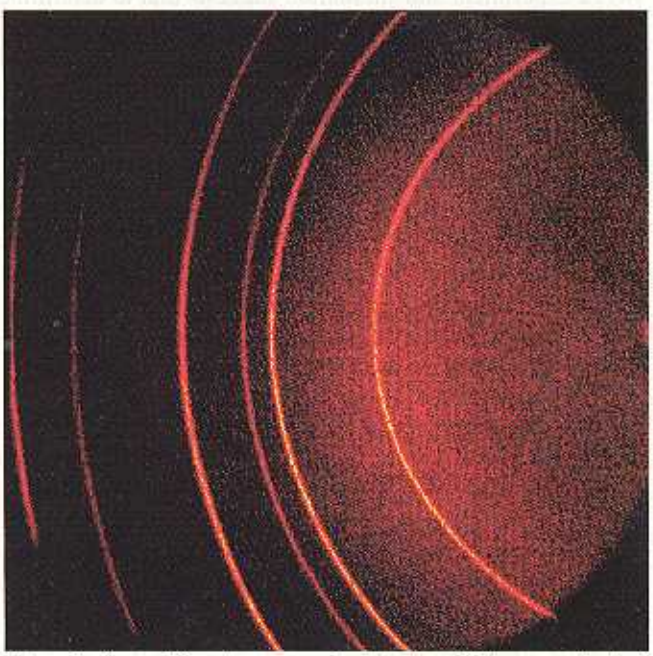

Figure 7. X-ray diffraction pattern. Production of this pattern involved transferring data from the x-ray diffractometer to an SGI workstation.

Finally, in Advanced Laboratory taken either in the winter or the spring, students continue to do data analyses, fit curves (including nonlinear functions), and graph experimental data by techniques already learned. In addition, they do more sophisticated online gathering of data, transfer data from the computer-based apparatus on which it is gathered to other machines for further analysis, and work with techniques for image processing. This course lays heavy expectation on the students' preparation of oral reports and formal written reports, so the course provides yet another motivation for students to develop their skills at computer-based preparation of technical documents.

Example 9: As part of his work with Professor Jeffrey Collett during the summer of 1996, Michael Stenner, LU '97, fathomed the structure of the file produced by a recently acquired Siemens x-ray machine, successfully read the file into IDL, and produced the x-ray image shown in Fig. 7 on the screen of an SGI workstation. Images obtained on the Siemens machine can now be massaged with the processing capabilities available in the CPL with IDL. In the future, students will be asked to use these capabilities both in the advanced laboratory course and in various independent senior research projects.

The conclusion of this article in the next issue of *Computers in Physics* will describe the impact of computers on the remainder of the Lawrence physics curriculum and offer reflections on the overall project.

From the Department Editor: *Please send your comments, suggestions, and manuscripts for submission to this column to Dr. Denis Donnelly, Department of Physics, Siena College, Loudonville, NY 12211. E-mail: donnelly@siena.edu.*

Available Electives

Students at Lawrence University can choose from a substantial number of electives in physics. Physics majors must take three of the following courses: Thermal Physics, **Advanced Electricity and Magnetism**, **Mathematical Methods**, Optics, Condensed Matter, **Advanced Mechanics**, Advanced Modern Physics, Tutorial in Physics, and Laser Physics. Two additional electives—**Computational Tools in Physics** and Independent Study in Physics—are offered, bringing the total available to 11. Courses shown in boldface have explicit computer content.

References

1. Comput. Phys. **4**, 197–201, 308–313 (1990); Proceedings of the Lawrence/Sloan Conference on Computing in Advanced Undergraduate Physics (1990); invited paper given at the Davidson Conference on Computational Physics in the Undergraduate Curriculum (1990); invited paper given at the summer American Association of Physics Teachers (AAPT) meeting in Boise, ID (1993); contributed paper given at the summer AAPT meeting in Spokane, WA (1995); invited paper given at the summer AAPT meeting in College Park, MD (1996).
2. Scientific Workstations in Undergraduate Physics, NSF-ILI, 1988; Discovery Approach to Introductory Physics Experiments, NSF-ILI, 1990; Partial Differential Equations in Advanced Undergraduate Physics, NSF-ILI, 1993.
3. Scientific Workstations in Undergraduate Physics, W. M. Keck Foundation, 1988; Preparing Physics Majors for a Capstone Experience, W. M. Keck Foundation, 1993.
4. J. Pauschenwein and B. Thaller, Comput. Phys. **10**, 558–566 (1996).

icking two bulk samples) such that the particles in one system do not interact with those in the other. We keep the two systems at the same temperature, but allow them to exchange volume and particles. The total volume and the total number of particles are kept fixed. After an initial transient, the two systems will come into equilibrium and the two phases coexist, even though they are not in direct physical contact (there is no liquid–vapor interface). This method works well, provided that we can efficiently exchange particles between the systems (exchanging volume is hardly ever a problem). Until recently, this condition limited the use of the Gibbs-ensemble method to moderately dense systems of small molecules, for example, methane, ethane, and propane. In principle, the method is correct for larger molecules (say hexa-decane), but the probability of transferring such a molecule from one system to the other becomes so small that equilibrium cannot be reached in any simulation of reasonable duration. As we discuss below, the configurational-bias Monte Carlo method has made it possible to overcome this problem.

Grand-canonical simulations

To explain how configurational-bias Monte Carlo can be used to study phase coexistence, we first consider a simpler problem, the adsorption of molecules in a porous medium. This example is simple, yet of considerable practical importance—an example is the adsorption of hydrocarbons in zeolite catalysts.[3] The problem of adsorption is simpler, because we need consider only the exchange of particles between the porous medium and the vapor phase. Moreover, we shall assume that the vapor phase is so dilute that it obeys the ideal-gas law. We wish to devise a Monte Carlo scheme that results in the correct Boltzmann distribution of the molecules over the two phases. The following derivation cuts a few corners but is essentially correct.

Let us first consider one molecule. The probability of finding this molecule in an infinitesimal volume element $d\mathbf{r}$ about the point $\mathbf{r}$ is proportional to $\exp[-\beta U(\mathbf{r})]\, d\mathbf{r}$, where $\beta \equiv 1/k_B T$ and $U(\mathbf{r})$ is the potential energy experienced by the molecule at point $\mathbf{r}$. The molecule may either be in the gas phase, which has a (large) volume V_0 or in the porous medium with volume V_1. It is convenient to write the probability of finding the molecule in an infinitesimal fraction $d\mathbf{s}$ of the volume of the porous medium about $\mathbf{r}$ as

$$N(\mathbf{r}) = c V_1 e^{-\beta U(\mathbf{r})} d\mathbf{s}, \tag{1}$$

where c is a normalization constant and $d\mathbf{s} \equiv d\mathbf{r}/V_1$. The probability that the particle is found in an equal infinitesimal fraction of the volume of the gas is

$$N(\mathbf{r}') = c V_0 d\mathbf{s}, \tag{2}$$

where we have used the fact that $U(\mathbf{r}') = 0$ in the ideal gas. The ratio of these probabilities is

$$\frac{N(\mathbf{r})}{N(\mathbf{r}')} = \frac{V_1}{V_0} e^{-\beta U(\mathbf{r})}. \tag{3}$$

If we perform a Monte Carlo trial move in which we attempt to swap a particle from the ideal gas to the porous medium, we should accept this trial move with the Metropolis probability

$$\mathrm{acc}(0 \to 1) = \min[1, (V_1/V_0)\exp\{-\beta U(\mathbf{r})\}].$$

A single particle is not very interesting, so let us consider a system (ideal gas plus porous medium) that contains M (indistinguishable) particles. The probability of finding a realization of this system where N particles are in the porous medium and $M-N$ in the ideal gas is given by

$$N(\mathbf{r}^N) \propto \left(\frac{V_1^N V_0^{M-N}}{N!(M-N)!}\right) e^{-\beta U(\mathbf{r}^N)} \propto \left(\frac{V_1}{V_0}\right)^N \left(\frac{M!}{N!(M-N)!}\right) e^{-\beta U(\mathbf{r}^N)}. \tag{4}$$

We now consider the limit of an infinite ideal-gas reservoir, that is, $M \to \infty$, $V_0 \to \infty$ while the density in the reservoir is fixed, $\rho = M/V_0$. Then we can rewrite Eq. (4) as

$$N(\mathbf{r}^N) \propto \frac{(\rho V_1)^N}{N!} e^{-\beta U(\mathbf{r}^N)}. \tag{5}$$

Now consider the ratio of the probabilities of finding configurations with $N+1$ and N particles in the porous medium. From Eq. (5) it follows that this ratio is

$$\frac{N(\mathbf{r}^{N+1})}{N(\mathbf{r}^N)} = \frac{\rho V_1}{N+1} e^{-\beta[U(\mathbf{r}^{N+1}) - U(\mathbf{r}^N)]}. \tag{6}$$

This ratio determines the acceptance probability of a Monte Carlo move in which we try to add (or remove) a particle to (from) the porous medium. Note that the properties of the reservoir enter only through the density ρ.

A simulation of a system in contact with an infinite particle reservoir is called a *grand-canonical Monte Carlo* simulation. Usually the control parameter in such simulations is the fugacity z, rather than the density ρ, and we write

$$N_{zVT}(\mathbf{s}^N) \propto \frac{z^N V^N}{N!} e^{-\beta U(\mathbf{s}^N)}. \tag{7}$$

For an ideal gas, the fugacity is equal to the density. The relation between the fugacity and the chemical potential is

$$z = \frac{e^{\beta\mu}}{\Lambda^3}, \tag{8}$$

where Λ is the de Broglie thermal wavelength.

Before discussing the configurational-bias Monte Carlo method, we recall a basic ingredient of all Monte Carlo simulations: we need to ensure that all points in configuration space are visited with the correct probability. Imagine that we perform a very large number of simulations (say M) in parallel. On average, we find that N_i out of the M systems are in configuration i. Of course, N_i fluctuates. However, because M is very large, the noise in the fraction

N_i/M is negligible. In equilibrium, N_i/M is proportional to the Boltzmann weight of state i.

In a Monte Carlo simulation, we allow a system to make jumps from its original state (denoted by o) to a new state n. In equilibrium, the rate at which a system leaves state o should be equal to the rate at which other states jump to state o. In practice, we usually impose a stronger condition, namely that the average "flow" from state o to any other state n, must be equal to the flow from n to o. This condition is called detailed balance and is written as

$$K(o\rightarrow n)=K(n\rightarrow o), \tag{9}$$

where $K(o\rightarrow n)$ is the flow of state o to n. This flow is given by the product of the probability of being in state o, the probability of generating a trial state n, and the probability of accepting this trial move,

$$K(o\rightarrow n)=N(o)\times\alpha(o\rightarrow n)\times \mathrm{acc}(o\rightarrow n). \tag{10}$$

In the conventional Metropolis Monte Carlo scheme, the probability of generating a particular state is equal for forward and reverse trial moves, that is, $\alpha(o\rightarrow n) = \alpha(n\rightarrow o)$. As a consequence, we can eliminate α in Eq. (10). If we use the fact that $N(i) \sim \exp[-\beta U(i)]$ in equilibrium, we obtain the following relation between the forward and reverse acceptance probabilities:

$$\frac{\mathrm{acc}(o\rightarrow n)}{\mathrm{acc}(n\rightarrow o)}=e^{-\beta[U(n)-U(o)]}. \tag{11}$$

The Metropolis acceptance probability (see the acceptance rule used in Box 2) satisfies this criterion.

In a grand-canonical simulation, acceptable trial moves are as follows.

(1) Displacement of particles. A particle is selected at random and moved to a new position. This trial move is accepted with a probability

$$\mathrm{acc}(s\rightarrow s')=\min[1,\ \exp\{-\beta[U(\mathbf{s}'^N)-U(\mathbf{s}^N)]\}]. \tag{12}$$

(2) Insertion and removal of particles. A particle is inserted at a random position or, with equal probability, a randomly selected particle is removed. The insertion of a particle is accepted with probability

$$\mathrm{acc}(N\rightarrow N+1) = \min\left[1, \frac{zV}{(N+1)}\exp\{-\beta[U(N+1)-U(N)]\}\right], \tag{13}$$

and the removal of a particle is accepted with probability

$$\mathrm{acc}(N\rightarrow N-1) = \min\left[1, \frac{N}{zV}\exp\{-\beta[U(N-1)-U(N)]\}\right]. \tag{14}$$

Boxes 1–3 show the basic structure of a simulation in the grand-canonical ensemble.

Grand-canonical Monte Carlo will fail if the acceptance of particle insertions or removals becomes too low. For atomic fluids, this condition effectively limits the maximum density at which the method can be used to about twice the critical density. However, for chain molecules, the method breaks down at a much lower density.

Application to polymers

Polymers are long chain molecules, which are typically modeled as a string of monomers connected together by a

Box 1. Basic grand-canonical-ensemble simulation.
The algorithm ensures that detailed balance is obeyed after each Monte Carlo step. Per cycle we perform on average npart attempts to displace particles and nexc attempts to exchange particles with the reservoir. Subroutine mcmove attempts to displace a particle (see Box 2), subroutine mcexc attempts to exchange a particle with a reservoir (see Box 3), and subroutine sample determines quantities (such as the potential energy or the pressure) every nsamp cycle.

```
PROGRAM mc_gc                          simulation of basic μVT-ensemble
do icycl=1, ncycl                      perform ncycl MC cycles
  ran=int(ranf()*(npart+nexc))+1
  if (ran.le.npart) then               displace a particle
    call mcmove
  else                                 exchange a particle
    call mcexc                         with the reservoir
  endif
  if (mod(icycl, nsamp).eq.0)          sample averages
&   call sample
enddo
end
```

Box 2. Attempt to displace a particle.
Subroutine ener calculates the energy of a particle at a given position. The ranf() function generates a random number uniform in [0,1].

```
SUBROUTINE mcmove
  o=int (ranf( )*npart)+1
  call ener (x(o), eno)
  xn=x(o)+(ranf( )-0.5)*delx
  call ener(xn, enn)
if (ranf( ).lt.exp(-beta
&    *(enn-eno)) x(o)=xn
return
end
```

attempt to displace a particle
select a particle at random
energy of old configuration
give particle random displacement
energy of new configuration
Metropolis acceptance rule
accepted: replace x(o) by xn

Box 3. Attempt to exchange a particle with a reservoir.
In this algorithm, trial removals and insertions are attempted with equal probability.

```
SUBROUTINE mcexc

if (ranf().lt.0.5) then
  if (npart.eq.0) return
  o=int(npart*ranf())+1
  call ener(x(o),eno)
  arg=npart*exp(beta*eno)
&       /(zz*vol)
if  (ranf().lt.arg) then
  x(o)=x(npart)
  npart=npart-1
endif
else
  xn=ranf()*box
  call ener(xn, enn)
  arg=zz*vol*exp(-beta*enn)
&       /(npart+1)
  if  (ranf().lt.arg) then
    x(npart+1)=xn
    npart=npart+1
  endif
endif
return
end
```

attempt to exchange a particle with reservoir
decide to remove or add a particle
test whether there is a particle
select a particle to be removed
energy particle o
acceptance rule (14)

accepted: remove particle o

new particle at random position
energy of new particle
acceptance rule (13)

accepted: add new particle

fixed spacing between adjacent monomers. One approach to simulating polymers is called reptation. In this approach a trial move consists of removing a monomer at one end of the polymer and adding a monomer at a random orientation at the other end. However, we would like to move the entire polymer to a new shape in one trial move. An arbitrary attempt to do this move or to add a polymer to the system will almost always be rejected. To overcome this low acceptance rate, we need to use a smarter Monte Carlo scheme. We shall use trial moves that are no longer completely random; the moves are biased in such a way that the trial position and shape of the polymer to be moved have an enhanced probability to fit into the existing configuration. A similar approach also will be used for polymer insertion and removal. Biasing a Monte Carlo trial move means that the probability to generate forward trial moves is no longer equal to the probability to generate reverse trial moves, that is, $\alpha(o \to n) \neq \alpha(n \to o)$. To satisfy detailed balance (10), we also must change the acceptance rule (11).

The starting point for the configurational-bias Monte Carlo technique is the static Monte Carlo scheme introduced by Rosenbluth and Rosenbluth in 1955 to sample

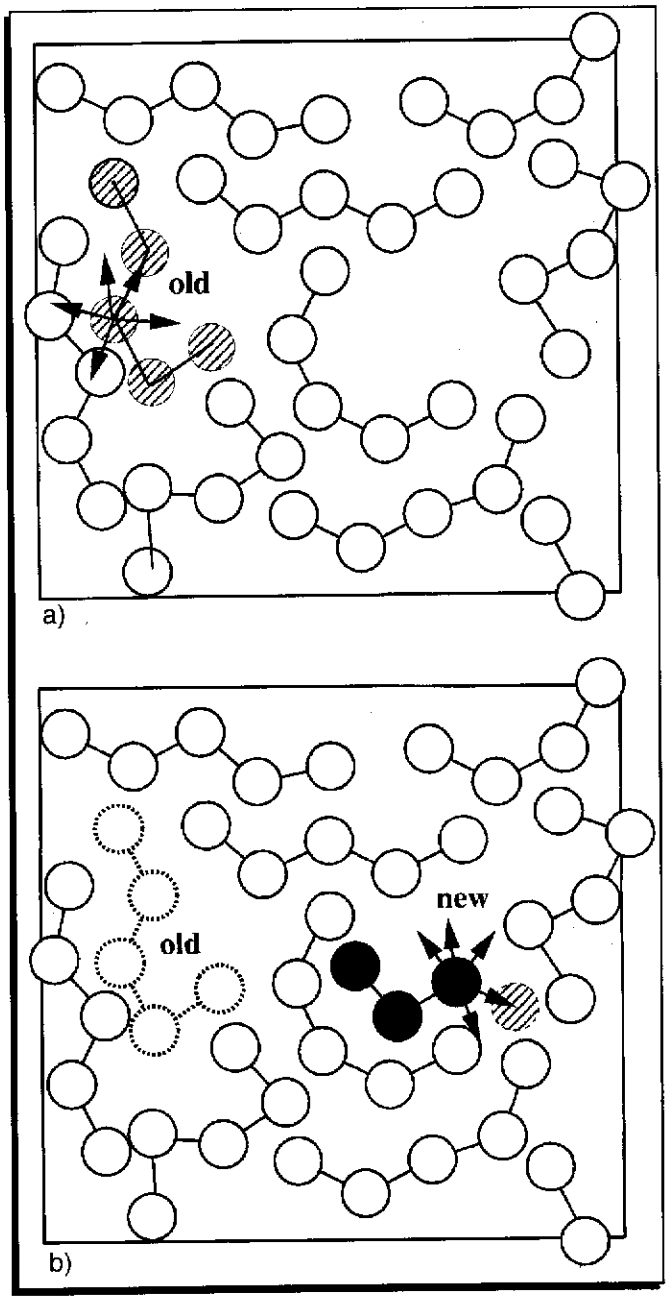

Figure 1. Sketch of the configurational-bias Monte Carlo scheme. The sketch shows the retracing of the old conformation (a) and the generation of a new configuration (b). The arrows indicate a set of trial positions.

polymer conformations.[4] In a static Monte Carlo scheme, each new configuration is generated from scratch. The problem with the Rosenbluth scheme is that the probability of generating a polymer configuration is *not* proportional to its Boltzmann weight. To correct for this bias, we have to weight every generated configuration by the "Rosenbluth weight" (see below). In the configurational-bias Monte Carlo method, we use the Rosenbluth scheme to generate trial moves, and we use the Rosenbluth weight to bias the *acceptance* of these trial moves. As we shall show, this procedure guarantees that all polymer conformations are generated with the correct Boltzmann weight. Note that the configurational-bias Monte Carlo algorithm is a dynamic rather than a static method, because new configurations are generated by applying a trial move to an existing configuration.

To illustrate the basic idea, we show how the configurational-bias Monte Carlo algorithm works for a lattice model consisting of N polymers each consisting of l monomers. The algorithm consists of the following steps:

(1) Choose a polymer at random, retrace its conformation [see Fig. 1(a)], and determine its Rosenbluth weight $W(o)$. In the following we discuss how to determine $W(o)$.

(2) Generate a trial conformation at a new position using the Rosenbluth scheme [see Fig. 1(b)] to grow the entire polymer and compute its Rosenbluth weight $W(n)$.

(3) Accept the trial move with a probability

$$\mathrm{acc}(o\rightarrow n)=\min[1,W(n)/W(o)]. \tag{15}$$

In practice, step (2) is carried out before step (1). The reason is that the generation of a new trial configuration often is unsuccessful. In that case, the trial move can be rejected right away, and there is no need to compute the Rosenbluth weight of the old configuration.

To determine the Rosenbluth weight of the old configuration denoted by o, we use the following steps [see Fig. 1(a)].

(1) Measure the energy of interaction $u_1(o)$ of the first monomer with other polymers and compute $w_1(o) = k \exp[-\beta u_1(o)]$, where k is the coordination number of the lattice.

(2) To compute the Rosenbluth weight for the remainder of the chain, we determine the interaction energy $u_i(j)$ of monomer i at its actual position, and the energy it would have had if it had been placed in any of the other $k-1$ sites (labeled by j) neighboring the actual position of monomer $i-1$ [see Fig. 1(a)]. The energy $u_i(j)$ includes all interactions of monomer i with other polymers in the system and with monomers 1 through $i-1$ of the same polymer. These energies are used to calculate the weights $w_i(o)$,

$$w_i(o)=e^{-\beta u_i(o)}+\sum_{j=2}^{k} e^{-\beta u_i(j)}. \tag{16}$$

(3) Once the entire chain has been retraced, we determine its Rosenbluth weight as

$$W(o)=\prod_{i=1}^{l} w_i(o). \tag{17}$$

The implementation of this scheme is shown schematically in Boxes 4 and 5.

The Rosenbluth scheme to generate a new trial conformation (denoted by n) of a polymer proceeds as follows [see Fig. 1(b)].

(1) The first monomer is inserted at a random position. Its energy is denoted by $u_1(n)$, and we define $w_1(n) \equiv k \exp[-\beta u_1(n)]$. (The factor k in the definition of the Rosenbluth weight of the first monomer, strictly speaking, is unnecessary. We introduce it here only to make the subsequent notation more compact.) For example, $k=6$ for a simple cubic lattice.

(2) For the next monomer, with index i, there are k possible trial directions. The energy of trial direction j is denoted by $u_i(j)$. From the k possible directions, we select one, say n, with a probability

$$p_i(n)=\frac{e^{-\beta u_i(n)}}{w_i(n)}, \tag{18}$$

where the partial Rosenbluth weight $w_i(n)$ is defined as

$$w_i(n)\equiv\sum_{j=1}^{k} e^{-\beta u_i(j)}. \tag{19}$$

Equation (18) biases the growing of the polymers in such a way that conformations with the lowest energy are selected with the highest probability. It does not include the interactions with monomers $i+1$ to l. Hence, the total energy of the polymer chain is given by $\mathcal{U}(n)=\sum_{i=1}^{l} u_i(n)$.

(3) Step (2) is repeated until the entire polymer chain is grown. The total Rosenbluth weight of configuration n is given by

$$W(n)=\prod_{i=1}^{l} w_i(n). \tag{20}$$

We need to demonstrate that the acceptance rule (15) removes the bias introduced by the use of the Rosenbluth scheme to generate trial conformations. The probability of generating a particular conformation n follows from the repeated use of Eq. (18),

$$\alpha(o\to n)=\prod_{i=1}^{l} \frac{e^{-\beta u_i(n)}}{w_i(n)}=\frac{e^{-\beta \mathcal{U}(n)}}{W(n)}. \tag{21}$$

Similarly, for the reverse move,

$$\alpha(n\to o)=\frac{e^{-\beta \mathcal{U}(o)}}{W(o)}. \tag{22}$$

Using Eqs. (21) and (22) in the requirement of detailed balance (9) imposes the following condition on the acceptance criterion:

$$\frac{\mathrm{acc}(o\to n)}{\mathrm{acc}(n\to o)}=\frac{W(n)}{W(o)}. \tag{23}$$

Clearly, the acceptance criterion given in Eq. (15) satisfies this condition.

Next we consider a configurational-bias Monte Carlo method for off-lattice systems. If the orientation of a monomer relative to a neighboring monomer is described by a continuous variable, then there is an essential difference with the lattice model. In the latter case all the possible orientations can be considered explicitly, and the corresponding Rosenbluth weight can be calculated exactly. For the continuum case, we cannot sample all possible orientations, and it is impossible to determine the exact Rosenbluth weight, because an infinite number of orientations are possible. Hence, the scheme for lattice models, in which the Rosenbluth weight for all orientations is calculated, cannot be used for a continuum model. A possible solution would be to use a large but finite number of trial directions. Surprisingly, this approximation is not necessary. It is possible to devise a *rigorous* algorithm using an arbitrary subset of all possible trial directions. As a consequence, an off-lattice trial move looks very much like its lattice equivalent.

(1) For the first monomer, a trial position $\mathbf{r}_n$ is selected at random, and the energy of this monomer is calculated. This energy is denoted by $u_1(n)$, and, as before, we define $w_1(n)=k \exp[-\beta u_1(n)]$.

Box 4. Basic configurational-bias Monte Carlo.
The details of the model are considered in subroutine grow (see Box 5 for a polymer on a lattice). Subroutine accept takes care of the bookkeeping of the new configuration.

```
      PROGRAM CBMC
      new_conf=.true.
      call grow(new_conf, wn)

      new_conf=.false.
      call grow(new_conf, wo)

      if (ranf().lt.wn/wo)
     &  call accept
      end
```

```
configurational bias Monte Carlo
first consider the new configuration
grow (part of) chain and calculate
Rosenbluth weight of new config.

next retrace old configuration to calculate
its Rosenbluth weight
acceptance test (15)
accept and do bookkeeping
```

Box 5. The growing of a polymer on a lattice.

The subroutine grows an l-bead polymer on a lattice with coordination number k and calculates its Rosenbluth weight w. If new_conf=.true., a new configuration is generated; if new_conf=.false., an old one is retraced. In a lattice model we consider all possible trial positions, denoted by b(j), and hence the existing configuration is automatically included. Subroutine select chooses one of the trial positions with probability $p(i)=w(i)/\Sigma_j w(j)$. Subroutine ener calculates the energy of the monomer at the given position with the other polymers and the monomers of the chain that have been grown already.

```
SUBROUTINE grow (new_conf,w)
if (new_conf) then                          insert first monomer
  xn (1)=ranf()*box
else
  o=ranf()*npart+1                          select old chain at random
  xn(1)=x(o,1)
endif
call ener(xn(1), en)                        calculate energy
w=k*exp(-beta*en)                           Rosenbluth weight of first monomer
do i=2, ell
  sumw=0                                    consider k trial directions
  do j=1,k
    xt(j)=xn(i-1)+b(j)                      determine trial position
    call ener(xt(j),en)                     determine energy trial position j
    w(j)=exp(-beta*en)
    sumw=sumw+w(j)
  enddo
  if (new_conf) then
    call select (w, sumw, n)                select one of trial positions
    xn(i)=xt(n)                             direction n is selected
  else
    xn(i)=x(o,i)
  endif
  w=w*sumw                                  update Rosenbluth weight
enddo
return
end
```

(2) For the subsequent monomers, k trial bonds of length b are generated, starting from the position of the previous monomer. We denote these trial bonds by $\{\mathbf{b}\}_k=(\mathbf{b}_1,\mathbf{b}_2,\ldots,\mathbf{b}_k)$. The end points of these vectors are distributed randomly on the surface of a sphere with radius b. For each trial position, the energy $u_i(\mathbf{b}_j)$ is calculated, and one of these positions is selected with probability

$$p_i(\mathbf{b}_n)=\frac{e^{-\beta u_i(\mathbf{b}_n)}}{w_i(n)}, \tag{24}$$

where

$$w_i(n)=\sum_{j=1}^{k} e^{-\beta u_i(\mathbf{b}_j)}. \tag{25}$$

In practice, it often is convenient to separate the intramolecular (bending and torsion) potential energy from the nonbonded ("external") potential energy. In that case, the trial positions for the monomers are drawn from a Boltzmann distribution of "internal" energies, whereas the $u_i(\mathbf{b}_j)$ correspond to the external interactions (for details, see Ref. 1).

(3) Step (2) is repeated until the entire polymer of length l has been grown, and the normalized Rosenbluth weight can be calculated as

$$\mathscr{W}(n)\equiv\frac{W(n)}{k^l}=\prod_{i=1}^{l}\frac{w_i(n)}{k}. \tag{26}$$

We can interpret $\mathscr{W}$ as the *ratio* of the Rosenbluth weight of the trial configuration to that of an ideal (non-self-avoiding) chain in the ideal gas reservoir. As in the case of normal configurational-bias Monte Carlo (15), we can derive from the condition of detailed balance that this ratio appears in the expression for the probability of accepting a trial insertion [see Eq. (27) below].

To compute the (off-lattice) Rosenbluth weight of the existing configuration of this polymer, we proceed as for the new configuration, except that we now generate $k-1$

randomly oriented trial monomers around every existing monomer. We compute the partial Rosenbluth weight, $w_i(o)$, for every set consisting of the actual monomer plus $k-1$ trial monomers. The total Rosenbluth weight of the existing chain conformation is $W(o)=\Pi_{i=1}w_i(o)$, and the normalized weight is defined as $\mathscr{W}(o)\equiv W(o)/k^l$.

Trial moves to insert a polymer are implemented as follows. First we randomly select a position for the first monomer of the polymer, and then we grow the rest of the chain and compute the Rosenbluth weight in the same way as we would for a trial move of an existing polymer. The selected polymer is inserted with a probability

$$\mathrm{acc}(N\rightarrow N+1)=\min\left[1,\frac{zV}{N+1}\mathscr{W}(n)\right]. \tag{27}$$

Trial moves to remove a polymer are implemented in a similar way. First we randomly select a polymer, then we compute the Rosenbluth weight $\mathscr{W}(o)$, and then the selected polymer is removed with a probability

$$\mathrm{acc}(N\rightarrow N-1)=\min\left[1,\frac{N}{zV}\frac{1}{\mathscr{W}(o)}\right]. \tag{28}$$

We note that for off-lattice configurational-bias Monte Carlo, the number of trial directions k can be chosen at will. The results that we obtain do *not* depend on k, but the statistical accuracy does. In these equations z denotes the fugacity of *ideal* (that is, non-self-avoiding) chains. In Ref. 1 it is explained how this fugacity can be related to the vapor pressure of real chains in a reservoir.

Smit and Maesen[3] used grand-canonical configurational-bias Monte Carlo simulations to compute the adsorption of hydrocarbons in zeolites (micro-porous materials). Whereas conventional grand-canonical simulations are limited to small alkanes (methane or ethane), the grand-canonical configurational-bias Monte Carlo method can be used to predict the adsorption of hydrocarbons that are used in industrial applications. For example, Fig. 2 shows the predicted adsorption isotherm of heptane in the industrially important zeolite silicalite.

So far we have discussed three of the four trial moves used in a Gibbs-ensemble simulation: moving a polymer, insertion of a polymer, and removal of a polymer. The last type of move we need is a change in the volume of each of the two systems such that the total volume V is fixed. To do so, we change the original volume V_1^o of one of the two systems by a random amount ΔV to V_1^n, while the volume of the other system (V_2^o) is changed by $-\Delta V$ to V_2^n. As in Eq. (1), it is convenient to define scaled coordinates s_i that are related to the real coordinates by $\mathbf{r}_i=V^{1/3}\mathbf{s}_i$. When changing the volumes, we keep the scaled coordinates of the polymers in both systems fixed. The real coordinates change with the volume, and hence the potential energy of both systems also changes. Imposing detailed balance leads to the following acceptance criterion:

$$\mathrm{acc}(o\rightarrow n)=\min\left[1,\frac{(V_1^n)^{N_1}(V-V_1^n)^{N-N_1}\exp(-\beta[U_1\{\mathbf{s}^{N_1};V_1^n\}+U_2\{\mathbf{s}^{N-N_1};V_2^n\}])}{(V_1^o)^{N_1}(V-V_1^o)^{N-N_1}\exp(-\beta[U_1\{\mathbf{s}^{N_1};V_1^o\}+U_2\{\mathbf{s}^{N-N_1};V_2^o\}])}\right], \tag{29}$$

where N_1 and $(N-N_1)$ are the numbers of polymers in systems 1 and 2, respectively, and $\mathbf{s}^N$ refers to all the scaled coordinates of N polymers. U_1, the total potential energy of system 1, and U_2, the total potential energy of system 2, are functions of the scaled coordinates of all particles in the system and of its volume. If system 2 is an ideal gas, then $U_2=0$.

We are now in a position to simulate two-phase coexistence using the Gibbs ensemble. First we choose randomly which type of trial move to make. The relative frequencies of these trial moves depend on the specific model under consideration. A good rule of thumb is to spend approximately equal amounts of computer time on all three types of moves. For an N-molecule system, this usually implies that trial displacements are $\mathscr{O}(N)$ times more frequent than volume moves. The frequency of particle exchanges depends on temperature and density—it is usually less frequent than translational moves, yet more frequent than volume moves. After a certain number of trial moves are made, thermodynamic quantities can be computed such as the density and pressure in each system. At the end of the

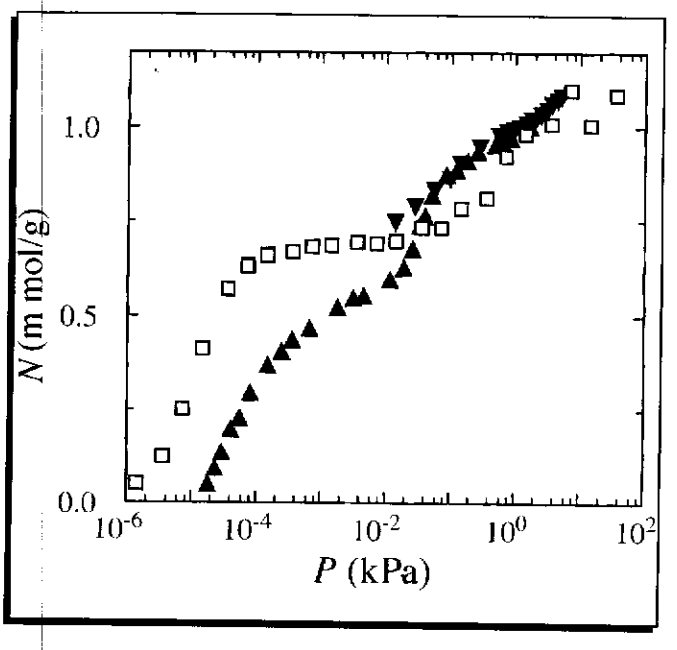

Figure 2. Adsorption isotherms of heptane in the zeolite silicalite (the number of adsorbed molecules N as a function of the pressure P). The closed symbols are experimental data and the open symbols the results from simulations at T=298 K. Zeolites are used as catalysts in the petrochemical industry. For these applications it is important to be able to predict the uptake of hydrocarbons by the zeolite.

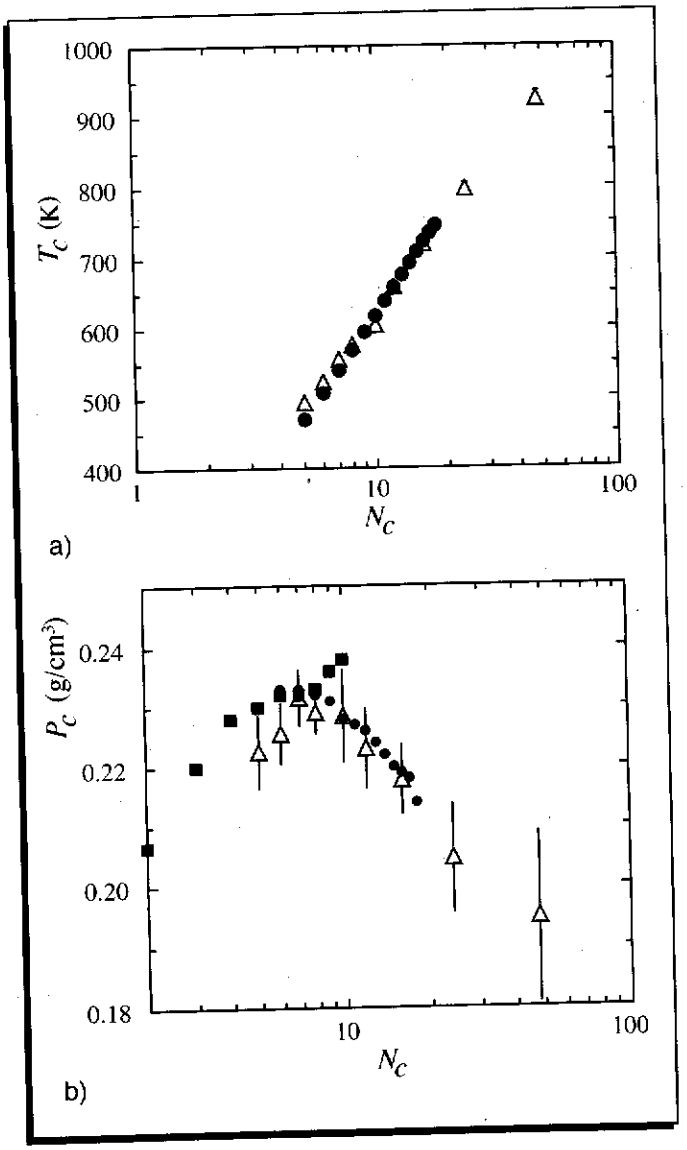

Figure 3. The critical temperature T_c (a) and critical density ρ_c (b) as a function of carbon number N_c. The open symbols are the simulation data and the closed symbols are experimental data (for details see Ref. 5).

simulation overall averages and statistical errors are computed.

An example that illustrates the power of grand-canonical configurational-bias Monte Carlo in the context of Gibbs-ensemble simulations is the prediction of the phase behavior of long-chain alkanes. Alkanes are thermally unstable above approximately 650 K. Therefore, at high temperatures it is necessary to measure very rapidly to reduce the effects of this decomposition on the thermodynamic properties. This requirement makes the experimental determination of the critical point of alkanes longer than decane extremely difficult, and consequently the phase behavior of long-chain alkanes is known only in part. Yet, a knowledge of the (hypothetical) critical properties of pure alkanes is important for predicting the thermodynamic behavior of the hydrocarbon mixtures that are encountered in the petrochemical industry. Siepmann *et al.*[5] used a combination of Gibbs-ensemble and configurational-bias Monte Carlo to simulate vapor–liquid equilibria of the longer n-alkanes for which experimental data are lacking. In Fig. 3 the critical temperatures and densities as predicted in Ref. 5 are plotted versus the carbon number. There are conflicting experimental estimates of the critical densities for the longer alkanes. The simulations lend support to those experiments that report a decrease in critical density with increasing carbon number. Without the use of grand-canonical configurational-bias Monte Carlo, the same simulations would have required astronomical amounts of CPU time.

Progress in computational materials science is not simply a matter of faster computers. Computational tricks, such as configurational-bias Monte Carlo, result in a speedup that far outweighs any gains due to improved hardware. More importantly, the applicability of the method is not limited to simplified model systems—it works for materials of practical importance. The combination of smarter algorithms and more powerful hardware is now rapidly transforming computational materials science from an eternally promising field to an essential tool in the design of novel materials.

Suggestions for further study

The following set of problems will help the reader to construct a grand-canonical configurational-bias Monte Carlo program for Lennard-Jones chain polymers. Because only a few papers have been written on grand-canonical configurational-bias Monte Carlo for chain polymers, there is a good chance that you will obtain new results. The Fortran programs discussed in the problems can be downloaded from the authors' Web site at http://www.hpcn.tudelft.nl/frenkel_smit.html.

(1) Write an NVT Metropolis Monte Carlo program to simulate Lennard-Jones particles. Use the algorithm in Box 2 as a starting point. You will need to evaluate the pressure from the virial (see Ref. 1, p. 75). Adopt the truncated and shifted Lennard-Jones potential

$$u(r)=\begin{cases} u_{\mathrm{LJ}}(r)-u_{\mathrm{LJ}}(r_c), & r\leq r_c \\ 0, & r>r_c \end{cases} \tag{30}$$

with

$$u_{\mathrm{LJ}}(r)=4\epsilon\left[\left(\frac{\sigma}{r}\right)^{12}-\left(\frac{\sigma}{r}\right)^{6}\right]. \tag{31}$$

You can download a program from the authors' Web site (Case Study 1). Use this program or one that you have written to calculate the pressure as a function of density for $T=2.0$ and $T=0.9$ in the density range $0<\rho<0.8$. Note that the temperature and density are expressed in reduced units, that is, we assume $\epsilon=\sigma=1$. Explain the qualitative differences between the two isotherms.

(2) Extend the program used in Problem 1 to include particle insertions and deletions (see Box 3) so that the grand-canonical ensemble is used. A program for this problem can be obtained from the Web (Case Study 9). Repeat

the calculations of the equation of state. In this case we impose the temperature and chemical potential, and the pressure and density are the results of the calculation. Check the consistency of the results by comparing the isotherms with the ones obtained in Problem 1. At low pressures the fugacity of the system should be equal to that of an ideal gas $z=\rho$. Can you reach the regime where this relation holds? What is the maximum density for which the results are still reliable? Your answer depends on the amount of CPU time you have available and the system size you want to study, but you should be able to observe a sharp decrease of the number of accepted insertions/deletions as a function of the density.

(3) As a simple model of a porous media, we can use a slitlike pore that can be described with the following potential:

$$U(z)=\begin{cases}\infty, & z<-H/2\\ 0, & -H/2<z<H/2\\ \infty, & z>H/2.\end{cases}$$

Modify the program used in Problem 2 so that this external potential is taken into account in the energy calculation. Consider the simulation of molecules in a single pore, and compute the adsorption isotherms (number of molecules adsorbed per unit volume as a function of the chemical potential) for $T=2.0$ and for $T=1.0$ with $H=4.0$ and $H=1.5$. Explain the shape of the isotherms.

(4) Use Boxes 4 and 5 as a guide to writing a configurational-bias Monte Carlo program for a single (off-lattice) chain in vacuum. Instead of generating the configurations on a lattice, generate a set of trial orientations on the surface of a sphere. (Case Study 20 on the Web site gives some useful hints on how to generate vectors on the surface of a sphere.) As an example, use this program to compute the radius of gyration R as a function of the number of monomers N of the chain. We expect $R\propto N^{\nu}$. Extract the exponent ν from a log–log plot of R versus N. If you plot ν as a function of temperature, you will be able to see the collapse transition of the chain where ν changes sharply.

(5) Now you should have all the ingredients to write a grand-canonical configurational-bias Monte Carlo program for chain molecules. Problem 4 gave the general scheme for generating a new configuration and for retracing the old one. This scheme should then be placed in a general framework of a grand-canonical Monte Carlo method as we have developed for simple Lennard-Jones molecules. Some hints can be obtained from Case Study 19 on the Web site, where we use configurational-bias Monte Carlo for a simulation at constant pressure in which the total number of particles remains constant but the volume can fluctuate.

As a test of the program, compute the equation of state in the grand-canonical ensemble and in the NVT ensemble for a system of chains consisting of eight Lennard-Jones beads. (Note that an NVT simulation can be done using a grand-canonical program with zero exchanges.) To do so we need to compute the pressure of a system with chain molecules. How can this calculation be done? (Hint: compute the virial between the centers of mass of the molecules.) What is the maximum density that can be obtained?

(6) Compute an adsorption isotherm for eight-bead chain molecules in a slitlike pore. How do these isotherms shift as a function of the number of monomers? Study the density profile in the slit.

Acknowledgments

The work of the FOM Institute is part of the research program of FOM and is supported by Nederlandse Organisatie voor Wetenschappelijk Onderzoek. We thank Jan Tobochnik and Harvey Gould for their critical reading of the manuscript and for suggesting numerous improvements in the text.

References

1. D. Frenkel and B. Smit, *Understanding Molecular Simulations: From Algorithms to Applications* (Academic, Boston, 1996).
2. A. Z. Panagiotopoulos, Mol. Phys. **61**, 813 (1987).
3. B. Smit and T. L. M. Maesen, Nature **374**, 42 (1995).
4. M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23**, 356 (1955).
5. J. I. Siepmann, S. Karaborni, and B. Smit, Nature **365**, 330 (1993).