PHYSICAL REVIEW E VOLUME 59, NUMBER 5 MAY 1999

Rosenbluth chain cluster growth in the study of micelle self-assembly

T. Dalby and C. M. Care
Materials Research Institute, Sheffield Hallam University, Pond Street, Sheffield S1 1WB, United Kingdom

(Received 5 October 1998)

A Rosenbluth algorithm [J. Chem. Phys. **23**, 356 (1955)] for enumerating clusters of chains is presented. The method is used to undertake a direct enumeration of the cluster partition function for small clusters in a three-dimensional lattice model of a binary mixture of amphiphile and solvent. In this model, the amphiphiles are represented as connected chains on a lattice, with vacant sites representing the solvent. The results from the Rosenbluth method are compared with those obtained by Metropolis Monte Carlo simulations which allow free self-assembly of clusters. The agreement between the two methods allows an unambiguous identification of the packing entropy associated with micelle self-assembly. Results are presented for unbranched chains having two head and four tail segments (H_2T_4) and also four head and four tail segments (H_4T_4). Although the cluster enumeration method described in this paper has been developed for micellar systems, it will have applications in a variety of areas including nucleation and percolation. [S1063-651X(99)06905-6]

PACS number(s): 02.70.−c, 61.20.Ja, 82.60.Lf, 83.70.Hq

I. INTRODUCTION

The problem of establishing exact results in the statistical mechanics of micellar self-assembly is particularly difficult because of the complexity of the molecules and the microscopic inhomogeneity of the micellar phase. Many workers have studied coarse grained models which allow the free self-assembly of amphiphiles, and particular attention has been given to lattice models [1–7]. Such models have been shown, by computer simulation, to exhibit many of the features of real amphiphile self-assembly such as a critical micelle concentration and a cluster size distribution showing an equilibrium between monomers and micellar like clusters. The work in this paper considers amphiphile clusters formed on a three-dimensional lattice, and results are obtained for amphiphiles which have four tail segments and either two head segments (H_2T_4) or four head segments (H_4T_4). The model is described in more detail in Sec. II.

It is possible to extract, from NVT Monte Carlo simulations, quantities such as the excess chemical potential per monomer and the excess enthalpy and entropy per monomer [8] by assuming that Hill's treatment [9] of imperfect gases in terms of physical clusters can be applied, in the limit of low concentrations, to micelle self-assembly. The work in this paper is motivated by a requirement to test the applicability of Hill's analysis by establishing a method to enumerate the cluster partition function of amphiphile clusters directly. Although the algorithm for cluster enumeration described here has been developed for one particular application, the method is applicable to a range of physical problems in the theory of nucleation, percolation, and branched polymers.

The cluster enumeration method is based on an extension of a scheme first proposed by Rosenbluth and Rosenbluth [10] for enumerating self-avoiding polymer chains. Our scheme is itself an extension of a ‘‘degenerate’’ Rosenbluth scheme developed by Care [11] for lattice animal enumeration. Results are obtained for the cluster partition function together with the enthalpic and entropic contributions to the process of micellization. The results are compared with those obtained using Metropolis Monte Carlo simulations in the NVT ensemble.

As explained in Sec. III, the central problem in developing a Rosenbluth algorithm for cluster counting is the problem of determining a *degeneracy* associated with the cluster growth. The *degeneracy* arises from the number of different ways in which the algorithm can construct essentially the same cluster, a problem which does not arise in linear polymer growth. In the method presented here the degeneracies become simple to calculate, and data can be collected for all cluster sizes as the clusters are grown. However, the penalty for the simplicity of degeneracy calculation is the need to control the way in which a cluster can grow. An alternative method of correcting for the degeneracy was proposed by Pratt [12]. In this latter scheme the correcting weight is more complicated to calculate, and must be recalculated at each stage of the cluster growth; however, the Pratt scheme does not require any restriction on the growth of the cluster.

In Sec. III, the Rosenbluth scheme for the growth of a single polymer chain is described. The Care method for enumerating lattice animals is summarized in Sec. IV A, and its extension to enumerating clusters of chains is given in Sec. IV B. The application of the method to calculating the entropic and enthalpic contributions to micellization and the comparisons with Metropolis simulations are described in Sec. V. Conclusions are presented in Sec. VI.

II. MODEL

The development of the Rosenbluth scheme described in this paper is motivated by an ongoing study of micelle self-assembly in a lattice model of an amphiphile and solvent mixture [13]. The amphiphiles are represented on a simple cubic lattice as unbranched, flexible, chains of s segments with h segments representing the hydrophilic head and the remaining (s-h) segments representing the hydrophobic tail. Vacant sites on the lattice represent the solvent. Each segment of a chain may be considered to represent a number of repeat units of a real amphiphilic chain. If only nearest neighbor interactions are allowed, the potential energy of the system may be written, without approximation, in the form

$$\frac{U}{kT}=\beta(n_{\mathrm{TS}}+\gamma n_{\mathrm{HS}}+\eta n_{\mathrm{HH}}+\epsilon n_{\perp}) \tag{1}$$

where n_{TS}, n_{HS}, n_{HH}, and $n_{\perp}$ are the total number of tail-solvent interactions, head-solvent interactions, head-head interactions, and right-angle chain bends. β is the ratio of tail-solvent energy to kT, and γ, η, and ϵ, respectively, are the head-solvent, head-head, and right-angle bond energies scaled by the tail-solvent interaction energy. In order for the model to represent amphiphilic behavior, the interaction energies are chosen with $\beta>0$ and $\gamma<0$. The parameter ϵ may be set greater than zero in order to introduce the effect of chain stiffness. In the current paper, the explicit head-head interactions are set to zero ($\eta=0$); this is not a serious approximation, since in the micellar region of the phase diagram there are relatively few nearest neighbor head-head interactions. It should also be noted that there is an effective head-head repulsion which arises from the requirement that the head segments be nearly fully solvated for the parameter values used.

The model is equivalent to a spin-1 Ising model, and allows the hydrophilic-lipophilic balance (HLB) to be adjusted independently of the temperature parameter β^{-1}, through the choice of both the chain geometry and the head-solvent interaction parameter γ. This is in contrast to the spin-$\frac{1}{2}$ form of a model used in Ref. [2] and by other workers (e.g., Ref. [14]) in which there is only one relevant energy parameter. Hence, in this latter model, once the temperature has been fixed, the HLB can only be adjusted by changing the relative sizes of the head and tail sections of the molecule. The spin-$\frac{1}{2}$ form of the Larson model can be mapped onto the model used here and it is found, ignoring the chain stiffness term and an additive constant, to be of the form

$$\frac{U}{kT}=\beta(n_{\mathrm{TS}}-n_{\mathrm{HS}}-2n_{\mathrm{HH}}). \tag{2}$$

Thus, in this representation, the spin-$\frac{1}{2}$ model includes an effective head-head attraction although it still forms micellar and other amphiphilic phases.

III. ROSENBLUTH SINGLE CHAIN GROWTH

The original Rosenbluth scheme [10] was developed to calculate the statistical properties of self-avoiding polymer chains using a Monte Carlo growth process. An ensemble of M chains is grown, and during the growth of each chain a weight is calculated which can be used to construct weighted averages whose expectation values give, for example, the number of independent chains, c_N, of length N or the root mean square chain extension.

Each chain in the ensemble is grown by placing an initial segment on a two- or three-dimensional simple-cubic lattice. Successive segments are then added to the end of the chain, thus growing the chain in a linear sequence. If we consider a chain which already has i segments, the $(i+1)$th segment is added to the end of the chain using the following algorithm.

(1) Assign a normalized probability p_i^{ω} for the addition of the $(i+1)$th segment to one of the ω sites adjacent to the end segment of the chain; assign a probability of zero to any site that is occupied, in order to prevent the chain from growing into itself.

(2) Select one of these sites, ω_i, by simple Monte Carlo sampling, with probability $p_i^{\omega_i}$.

(3) Repeat steps (1) and (2) until the chain has grown to the required length N.

(4) Associate a weight W_α with the construction of each member, α, of the ensemble. Rosenbluth and Rosenbluth chose

$$W_\alpha=\frac{1}{P_\alpha}=\frac{1}{\prod_{i=2}^{N} p_i^{\omega_i}}. \tag{3}$$

If it is impossible to complete the growth of the chain, because all directions are blocked, the weight for the chain is set to zero. The chain must be included in the counting associated with the weighted average defined below.

Provided that $\Sigma_\omega p_i^{\omega}=1$, there is no necessity for all of the p_i^{ω} is to be equal, but this is usually the choice made for construction of athermal chains. It is important that the choice of p_i^{ω} allows the growth of all possible chains (i.e., is ergodic); the choice of p_i^{ω} affects the speed with which the method converges to the required averages. A weighted average over the ensemble may be defined for any property O of the chains as follows:

$$\langle O\rangle_W=\frac{1}{N_E}\sum_{\alpha=1}^{N_E} W_\alpha O_\alpha, \tag{4}$$

where N_E is number of chains grown in the ensemble. If W_α is given by Eq. (3), the expectation value of weighted averages is given by

$$E[\langle O\rangle_W]=\sum_{\nu=1}^{c_N} O_\nu, \tag{5}$$

where the summation is over all possible distinguishable chain configurations, ν, and hence, for example,

$$E[\langle 1\rangle_W]=c_N, \tag{6}$$

$$E[\langle R_\nu^2\rangle_W]=\sum_{\nu=1}^{c_N} R_{N\nu}^2=c_N\overline{R_N^2}. \tag{7}$$

It is possible to calculate averages in the canonical ensemble if the p_i^{ω}'s are chosen to be

$$p_i^{\omega}=\frac{\exp^{-\beta(U_i^{\omega}-U_{i-1})}}{\sum_{\omega}\exp^{-\beta(U_i^{\omega}-U_{i-1})}}, \tag{8}$$

and W_α is set to

$$W_\alpha=\prod_{i=1}^{N}\left(\sum_{\omega}\exp^{-\beta(U_i^{\omega}-U_{i-1})}\right), \tag{9}$$

where U_{i-1} is the energy of the chain of length $i-1$, and U_i^ω is the energy of the chain if the ith segment is placed in position ω. Using this choice of p_i^ω and W_α, we can obtain an estimate of the canonical partition function using

$$E[\langle 1\rangle_W]=\sum_{\nu=1}^{c_N} \exp^{-\beta U_\nu}, \tag{10}$$

and also obtain canonical averages of other quantities.

IV. ROSENBLUTH CLUSTER GROWTH

In a previous paper [11] one of the authors developed a Rosenbluth technique for the enumeration of the number of distinct lattice animals of N sites (clusters of single sites) on any given lattice. The method constructs each cluster by repeatedly adding sites to the surface of a connected cluster and calculating an associated weight as in the standard Rosenbluth scheme. The principal problem to be overcome is that the growth process introduces a ''degeneracy'' which arises from the many different ways in which the same cluster can be constructed. Thus if each addition to the cluster is labeled sequentially as the cluster grows, there are many different labelings which may be given to the same final cluster shape. Unfortunately, the calculation of the degeneracy is nontrivial; its value depends upon the details of the growth algorithm and the connectivity of any given cluster.

The method developed for correcting for this degeneracy [11] involves restricting the cluster growth sequence in such a way that each possible cluster labeling can be grown in only one way. This yields a degeneracy of exactly $N!$. In this scheme, the clusters are constructed using a set of labeled ''bricks,'' and the method is based on the observation that for any *given* connected cluster of labeled objects, a *unique* growth sequence can be devised in which, at each stage of the growth of this given cluster, the next object to be added to the surface of the growing cluster is that object which is (i) connected to the cluster, and (ii) carries the lowest label. An arbitrary cluster of seven labeled bricks is shown in Fig. 1(a), and the associated growth sequence is shown in Figs. 1(b1)–1(b7). In Ref. [11] the following algorithm is shown to enforce the above growth sequence during the growth of a labeled cluster.

A. Single site cluster algorithm

An ensemble of clusters of size N is constructed, and for each cluster a weight is calculated which can subsequently be used to calculate weighted averages of cluster properties. Each cluster is constructed in a connected growth sequence from a set of labeled ''bricks''; the label on each brick is denoted by κ, an integer in the range $1,\ldots,N$ inclusive. As bricks are added to the cluster we maintain a record of the set of ''surface'' sites (ω in number) which are adjacent to the cluster, i.e., vacant, and connected to the cluster. For each such surface site we record a quantity κ_m which denotes the minimum κ value allowed at that site. This is the method which is used to control the growth of the cluster to ensure the required degeneracy. The value of κ_m for any given surface site changes, in a manner described in detail below, as the cluster is grown. Each cluster growth is begun by placing

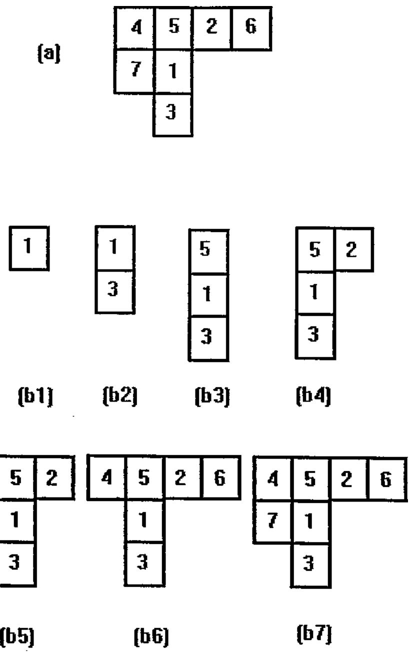

FIG. 1. Example of growth sequence for a cluster of seven labeled bricks. (a) Final cluster with arbitrary labeling. (b1)–(b7) Unique growth sequence described in Sec. IV to achieve a given labeled cluster.

the brick with $\kappa=1$ on the lattice and repeating the following steps until the cluster is fully grown.

(1) Select one of the surface sites as the site which is next to be occupied and delete the site from the list of available surface sites.

(2) Select one of the remaining bricks with a κ value greater than or equal to the κ_m for that surface site.

(3) Add the brick to the cluster and remove it from the set of available bricks.

(4) Adjust the record of surface sites and their associated κ_m values.

(5) Accumulate the data necessary to calculate the weight to be associated with the cluster.

We now comment on each of these steps in more detail.

(1) The surface site to be used for the attachment of a brick is chosen from the set of ω available surface sites by simple Monte Carlo sampling with a normalized probability p_i^ω. The value $p_i^{\omega_i}$ associated with the selected surface site is recorded for the subsequent evaluation of the weight, W_α, to be associated with cluster α.

(2) A brick is selected from the subset of remaining bricks which have $\kappa\geqslant\kappa_m$, where κ_m is the minimum allowed κ value for that surface site. The brick is chosen with a normalized probability p_i^κ. The value $p_i^{\kappa_i}$ for the chosen brick is recorded to calculate W_α. In the following we assume that the selected brick has $\kappa=\kappa_s$. It was found in Ref. [11] to be preferable if the p_i^κ's are chosen such that $p_i^\kappa\propto\lambda^\kappa$, where $1\geqslant\lambda\geqslant 0$, and the parameter λ is chosen empirically, for given N, to minimize the skewness of the probability distribution of the weights, $P(W)$. λ was typically set to be 0.9.

(3) The brick is added to the record of the current cluster and removed from the list of available bricks.

(4) For each *old* surface site in the *current* table of surface sites, set

$$\kappa_m|_{\rm new} = \max(\kappa_m|_{\rm old}, \kappa_s + 1). \tag{11}$$

If none of the remaining bricks has a κ value greater than, or equal to, $\kappa_m|_{\rm new}$, the site is removed from the list of surface sites since no bricks could subsequently be placed at that site. *New* surface sits associated with the brick placed at step (2) are identified; in order to qualify as a new surface site, the site must not appear in the current table of surface sites. The new surface sites are added to the list of adjacent sites, and, for each new surface sites, κ_m is set to the lowest κ value of the remaining bricks since there is no restriction on the bricks which may be placed on *new* surface sites. *The adjustment of the surface sites and their associated κ_m is the key process in enforcing the unique cluster growth sequence described at the beginning of Sec. IV.*

(5) A weight $W_\alpha^M = 1/(d_M \Pi_{i=2}^M p_i^{\omega_i} p_i^{\kappa_i})$ is associated with each cluster of size $1 \leqslant M \leqslant N$ where the degeneracy $d_M = M(N-1)!/(N-M)!$. If it is impossible to complete the growth of the cluster, because there are no available surface sites, the weight for the cluster is set to zero. Thus data can be collected for all cluster sizes up to size N simultaneously.

Once an ensemble of clusters has been generated, weighted averages can be calculated. More details of the method were given in Ref. [11], where it was shown that the method successfully enumerates lattice animals up to size 30 on both two- and three-dimensional simple cubic lattices. Results are presented in Table I.

B. Chain cluster growth

In this section we describe how the single site cluster algorithm described in Sec. IV A may be extended to grow Boltzmann weighted clusters of amphiphilic *chains*. An ensemble of clusters of labeled chains is grown in an analogous manner to the growth of a cluster of single sites described in Sec. IV A with adjacent ''surface'' sites labeled with κ_m values. The weights accumulated during the cluster growth may be used for the direct enumeration of the partition function as shown in Sec. III, as well as other Boltzmann weighted cluster averages.

Each chain in the cluster is itself grown using a Rosenbluth scheme modified to allow the chain to be grown from any position along its length and not in a linear sequence. This change is necessary in order to allow all possible cluster configurations to be observed. It is also necessary to introduce an additional ''degeneracy'' associated with the number of contact points the chain makes with the cluster. We now describe in more detail the addition of a chain to an established cluster of chains.

(1) A Monte Carlo choice is made to determine if the first segment of the new chain to be placed should be a head or a tail. This choice is made with a probability $p_1^{\rm seg} = 1 - \phi$ for a tail and $p_1^{\rm seg} = \phi$ for a head. In order to correct for the bias introduced by this choice, the weight associated with the cluster is multiplied by $1/p_1^{\rm seg}$. It is necessary to introduce the quantity ϕ because the use of unbiased Boltzmann weights results in the first segment nearly always being selected as a head segment, and this leads to poor sampling of the partition function, even for monomers. The value of ϕ is adjusted empirically to minimize the variance in the calculated averages and maximize the rate of convergence of the averages. The value of this parameter depends greatly on the ratio of head to tail segments present in an amphiphilic chain; for a chain of length 6 (H_2T_4), it was set at 0.25.

(2) Once the type of segment has been selected, a Boltzmann weighted Monte Carlo choice is made of the surface site at which the segment is to be placed. Thus the probability of the segment being placed at a given surface site, ω, is made proportional to $B_1^\omega = \exp(-\beta \Delta U_s^\omega)$ where ΔU_s^ω is the change in energy associated with placing the segment at that site. The weight for the cluster is multiplied by $Z_1 = \Sigma_\omega B_1^\omega$ in order to achieve Boltzmann weighted averages, as explained in Sec. III.

(3) The growth of the chain is continued by adding segments at either end of the chain. For each segment a choice is first made of the type of segment to be placed and then a Boltzmann weighted decision is made as to the position of the segment. Once all the segments from a type (head or tail) have been placed, the probability $p_\mu^{\rm seg}$ for the addition of segment μ must be modified to either 1 or 0 to ensure that the final chain has the correct structure. Note that only tails may be grown onto the tail segment and heads onto the head segment.

TABLE I. Estimates of the number of lattice animals of size N on a three-dimensional simple-cubic lattice. Comparison of (i) the chain cluster algorithm using 1×10^9 sample clusters, (ii) the original single site algorithm [11] using 1.8×10^7 sample clusters, (iii) exact values, and (iv) Lam's estimate [17].

N	This paper	Care estimate	Exact value	Lam estimate
2	3.000×10^{0}	3.000×10^{0}	3.000×10^{0}	
3	1.500×10^{1}	1.500×10^{1}	1.500×10^{1}	
4	8.600×10^{1}	8.597×10^{1}	8.600×10^{1}	8.594×10^{1}
5	5.340×10^{2}	5.339×10^{2}	5.340×10^{2}	5.321×10^{2}
6	3.481×10^{3}	3.485×10^{3}	3.481×10^{3}	3.475×10^{3}
7	2.350×10^{4}	2.352×10^{4}	2.350×10^{4}	2.353×10^{4}
8	1.629×10^{5}	1.629×10^{5}	1.629×10^{5}	1.631×10^{5}
9	1.153×10^{6}	1.154×10^{6}	1.153×10^{6}	1.155×10^{6}
10	8.295×10^{6}	8.296×10^{6}	8.295×10^{6}	8.291×10^{6}
11	6.050×10^{7}	6.050×10^{7}	6.049×10^{7}	6.042×10^{6}
12	4.462×10^{8}	4.461×10^{8}	4.462×10^{8}	4.442×10^{6}
13	3.323×10^{9}	3.321×10^{9}	3.323×10^{9}	3.291×10^{6}
14	2.495×10^{10}	2.493×10^{10}		2.461×10^{10}
15	1.886×10^{11}	1.884×10^{11}		1.862×10^{11}
16	1.435×10^{12}	1.434×10^{12}		1.416×10^{12}
17	1.098×10^{13}	1.095×10^{13}		1.082×10^{13}
18	8.440×10^{13}	8.412×10^{13}		8.329×10^{13}
19	6.516×10^{14}	6.507×10^{14}		6.446×10^{14}
20	5.049×10^{15}	5.036×10^{15}		5.002×10^{15}
21	3.927×10^{16}	3.917×10^{16}		3.897×10^{16}
22	3.064×10^{17}	3.059×10^{17}		3.052×10^{17}
23	2.397×10^{18}	2.338×10^{18}		2.391×10^{18}
24	1.883×10^{19}	1.872×10^{19}		1.877×10^{19}
25	1.483×10^{20}	1.461×10^{20}		1.480×10^{20}
26	1.165×10^{21}	1.165×10^{21}		1.168×10^{21}
27	9.156×10^{21}	9.321×10^{21}		9.209×10^{21}
28	7.281×10^{22}	7.251×10^{22}		7.290×10^{22}
29	5.804×10^{23}	5.555×10^{23}		5.786×10^{23}
30	4.608×10^{24}	4.359×10^{24}		4.610×10^{24}

(4) At each stage in the growth, the weight is multiplied by the appropriate factors to generate the Rosenbluth weight. If at any point during the addition of the Mth chain the growth becomes blocked, the weights for the clusters from M to N inclusive are set to zero. Note that it is essential that these zero weight terms are included in the calculation of any weighted averages. After the addition of the ith chain, the following factor is calculated:

$$w_i = \prod_{\mu=1}^{s} \frac{Z_\mu}{p_\mu^{\mathrm{seg}}}. \tag{12}$$

(5) As the chain is growing, a note is kept of the *maximum* κ_m value for any surface site that becomes occupied by the chain growth. When the chain is complete, a κ label is selected for it from the set of remaining κ labels which are greater or equal to this value of κ_m with probability $p_i^\kappa \propto \lambda^\kappa$ as for the single sites. Once again the parameter λ is selected empirically to minimize the variance of the final weighted averages; for a chain of length $6(H_2T_4)$, it was set at 0.95.

(6) After the label for the added chain has been selected, all the old and new adjacent sites have their κ_m values set in the same way as in the single site cluster algorithm.

(7) A degeneracy d_c is associated with the growth of the individual chain. This degeneracy arises because the final chain could have started growth from any of its contact points with the original cluster, and is given by

$$d_c = \sum_{\{c\}} \binom{s-1}{c-1}, \tag{13}$$

where the summation $\{c\}$ is over the set of segments which make contact with the original cluster onto which it has just been grown; future contact points of the chain are not included in this degeneracy. In Eq. (13) the numerical value of c is determined by indexing the segments of the chain sequentially from $c=1$ to $c=s$. If the chain makes contact at every segment $d_c = 2^{s-1}$, as would be expected.

(8) The weight associated with each cluster of size $1 \leqslant M \leqslant N$ is

$$W_\alpha^M = \frac{1}{d_M} \prod_{i=1}^{M} \frac{w_i}{d_c^i p_i^{\kappa_i}}, \tag{14}$$

where the degeneracy $d_M = (M(N-1)!/(N-M))!$. Hence data are collected for all cluster sizes up to size N simultaneously.

It should be noted that the parameters ϕ and λ do not affect the expectation value of the weighted averages, since the bias they introduce is canceled through the Rosenbluth weights. Thus the weighted averages yield essentially exact values for the measured quantities, since the Rosenbluth weights are chosen to correct for any bias introduced in the sampling process. A schematic outline of the structure of the code used to implement the algorithm is given in the Appendix. The following validation tests were undertaken of the final code.

(1) Lattice animals (i.e., clusters of chains of length 1) up to size 30 were grown, and the results compared with exact results and also estimates obtained in Refs. [11] and [17]. The results are given in Table I, and it can be seen that the new algorithm is in good agreement with the previous results.

TABLE II. Comparison of results from the new Rosenbluth algorithm using 1×10^6 sample chains and values obtained by direct enumeration of the possible athermal configurations of a single chain.

Chain length	Direct evaluation	Rosenbluth result
2	6	6.00
3	30	30.00
4	150	150.00
5	726	726.07
6	3534	3533.36

(2) A direct enumeration method was used to count the number of independent athermal arrangements of a single chain. The results from the new chain cluster algorithm are compared with these results for chains up to chains of length 6 in Table II.

(3) An independent code was written to grow athermal clusters of two chains using a simple extension of the standard Rosenbluth scheme [10]. This scheme grew nonpolar chains from any point along the chain. The new chain cluster scheme replicates these results by setting the number of tails to zero and also setting β very low (e.g., $\beta = 1\times10^{-9}$). The results for both simulations are compared in Table III.

V. RESULTS

A. Rosenbluth method

The Rosenbluth scheme described above has been used directly to evaluate the cluster partition function

$$Q_N^c(V,T) = \sum_{\langle i\rangle} \exp(-U_i^N/kT), \tag{15}$$

where the summation $\langle i\rangle$ is over all distinguishable connected N clusters, assuming that the amphiphile chains are indistinguishable. The superscript c indicates that the summation does not include any translation of the cluster. It is also possible to evaluate the canonical averages $\overline{t^N}$, $\overline{h^N}$, and $\overline{r^N}$ of the number of tail-solvent interactions, head-solvent interactions, and right-angle bonds for clusters of size N defined, for example, by relations of the form

$$\overline{t^N} = \langle t_N\rangle_W = \sum_{\langle i\rangle} p_i^{cN} t_i^N, \tag{16}$$

TABLE III. Number of distinguishable clusters of two chains. Comparison of results from new Rosenbluth scheme and an independent code both using 1×10^6 sample clusters.

Chain length	New Rosenbluth	Check Rosenbluth
3	1.105×10^4	1.105×10^4
4	3.882×10^5	3.880×10^5
5	1.180×10^7	1.179×10^7
6	3.469×10^8	3.464×10^8

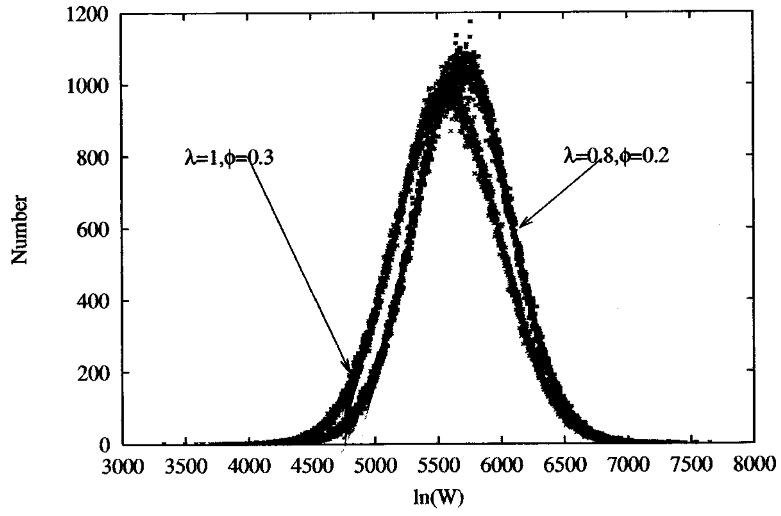

FIG. 2. Probability distribution $P(\ln(W))$ of the weights W for different values of the parameters λ and ϕ.

where $\langle\, \rangle_W$ is the Rosenbluth Boltzmann weighted average, and

$$p_i^{cN}=\frac{\exp(-U_i^N/kT)}{Q_N^c} \tag{17}$$

is the canonical ensemble probability of a particular N cluster. Note that the vacant sites are solvent sites, and hence $\overline{t^N}$ and $\overline{h^N}$ are determined by the surface sites of the cluster. The average internal energy per N cluster, $\overline{U^N}$, is defined by

$$\frac{1}{kT}\overline{U^N}=\beta(\overline{t^N}+\gamma\overline{h^N}+\epsilon\overline{r^N}). \tag{18}$$

We may use the relation

$$-\ln(Q_N^c)=\frac{\overline{U^N}}{kT}-\frac{S^N}{k}=\frac{\overline{U^N}}{kT}+\sum_{\langle i\rangle} p_i^{cN}\ln p_i^{cN} \tag{19}$$

to calculate the excess packing entropy per monomer defined by

$$\frac{1}{k}\left(\frac{S^N}{N}-S_1\right)=-\frac{1}{N}\sum_{\{i\}} p_i^{cN}\ln(p_i^{cN})+\sum_{\langle i\rangle} p_i^{c1}\ln(p_i^{c1}). \tag{20}$$

The Rosenbluth scheme was used to evaluate the cluster excess entropy and excess enthalpy, $(1/kT)(\overline{U^N}/N-\overline{U^1})$, for clusters up to size 6 for H_4T_4 and up to size 8 for H_2T_4 by growing 10^9 clusters and calculating appropriate weighted averages. The values of the parameters ϕ and λ were established empirically to minimize the variance of $P(W)$, the probability distribution of the cluster weights. The effect of varying these parameters is illustrated in Fig. 2, where it can be seen that this distribution is approximately log normal.

B. Comparison with Metropolis simulations

Extensive Metropolis Monte Carlo simulations were also undertaken in the NVT ensemble (cf. Refs. [15,8] for more details). In the absence of cluster-cluster interactions, it is possible to use Hill's analysis [9] of physical clusters to show [8] that the excess chemical potential, at infinite dilution, of a cluster of size N is given approximately by

$$\frac{1}{kT}(\mu_N^0-\mu_1^0)=\frac{1}{kT}\left(\frac{\overline{U^N}}{N}-\overline{U^1}\right)-\frac{1}{k}\left(\frac{S^N}{N}-S^1\right)+\left(1-\frac{1}{N}\right)\ln(s). \tag{21}$$

The excess chemical potential at infinite dilution, $(1/kT)(\mu_N^0-\mu_1^0)$, is measured from the NVT simulations by appropriate inversion of the cluster size distribution. Results can also be obtained for the averages $\overline{t^N}$, $\overline{h^N}$, and $\overline{r^N}$. Hence the NVT Metropolis simulations yield estimates of the excess enthalpy and entropy as a function of cluster size.

The monomer partition function Q_1^c can be calculated exactly and was found to agree with the Rosenbluth estimate within 0.003% for H_2T_4. The quantities $\overline{t^1}$, $\overline{h^1}$, and $\overline{r^1}$ were also calculated from Q_1^c and found to agree with the Rosenbluth values to within 10^{-5} for H_2T_4, and with the Metropolis data to within 0.6%.

Data from the Rosenbluth scheme and the Metropolis simulations are shown in Figs. 3 and 4. It can be seen that the two methods are in good agreement up to clusters of size 8 for H_2T_4 and size 6 for H_4T_4. Above these cluster sizes, the Rosenbluth scheme becomes unusable because the sampling distribution of the weights becomes highly skew and the averages obtained have very high variance. This problem was recognized by Batoulis and Kramers [16] in their study of polymer chain growth using the Rosenbluth method. However, these authors concluded that acceptable sampling of polymer chains up to length $N=240$ was possible. Although a linear chain is not directly comparable with a cluster of individual chains, it still seems possible that improvements to the algorithm described here may yield accurate information on significantly larger clusters than have been possible here. It seems likely that the problem arises here because the algorithm is not efficiently sampling the important clusters for large N. This is probably due to the amphiphile clusters being constructed sequentially, which does not efficiently sample the lowest energy structures for large N. However, it

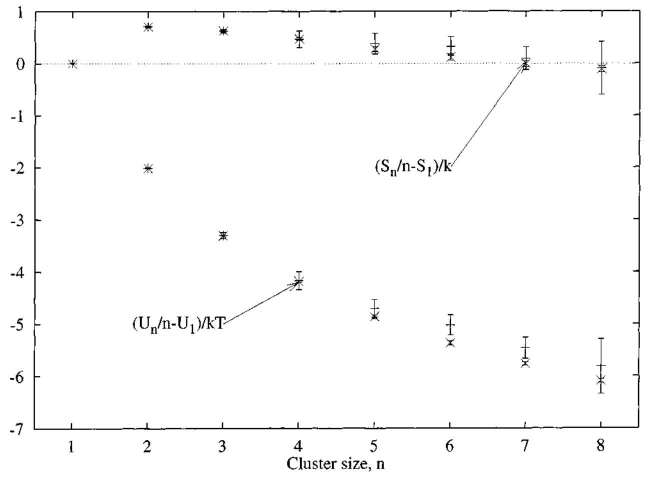

FIG. 3. Comparison of excess entropy per molecule and excess enthalpy per molecule for the H_2T_4 amphiphile. (×): Metropolis Monte Carlo; (+): Rosenbluth cluster growth.

may be possible to improve the cluster growth algorithm to more efficiently sample these low energy structures.

The logarithm of the cluster partition function is estimated to have an error of less than 1% for the data presented here. This error estimate is obtained from appropriate block averaging of the results. However the sampling distributions are highly skew, and it is therefore possible that these errors underestimate the true errors. The excess enthalpy is very sensitive to the nature of the clusters which are sampled, and the measured uncertainties are correspondingly much higher. These larger uncertainties are therefore observed in the estimates of the excess entropy which is calculated from Eq. (19).

The Rosenbluth results provide an independent check of the validity of using the partition function for independent clusters in the analysis of micellar systems; this assumption is implicit in the derivation of Eq. (21). The agreement between the Metropolis and Rosenbluth data for small *N* supports the use of Eq. (21) to analyze the Metropolis data over the wider range of *N*. The results from the Metropolis calculation are shown in Fig. 5 for larger cluster sizes. It can be seen that the difference in behavior of the two species can be attributed to the excess entropy rather than the excess enthalpy. It can be seen that the excess entropy per monomer initially grows for the smallest cluster sizes and then monotonically decreases as the clusters grow. The initial growth arises because of the rapid increase in the number of distinguishable clusters as the number of monomers in a cluster increases. The subsequent reduction in entropy per monomer, as the clusters grow, is related to the loss of freedom associated with packing the monomers in such a way as to maintain the head-solvent interactions while minimizing the tail-solvent interactions (hydrophobic effect).

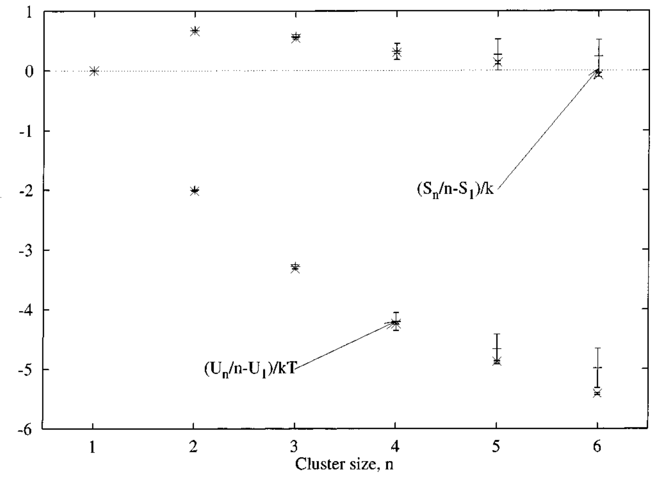

FIG. 4. Comparison of excess entropy per molecule and excess enthalpy per molecule for the H_4T_4 amphiphile. (×): Metropolis Monte Carlo; (+): Rosenbluth cluster growth.

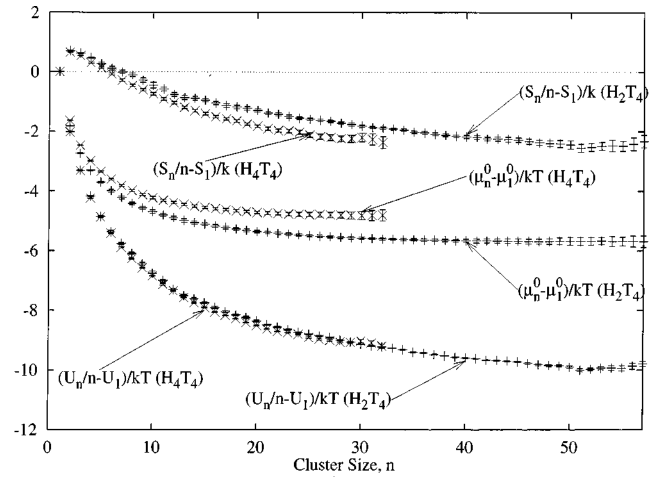

FIG. 5. Metropolis Monte Carlo data. Excess chemical potential per molecule, $(\mu_n^0-\mu_1^0)/kT$, excess enthalpy per molecule, $(\overline{U}^n/n-\overline{U}^1)/kT$, and excess entropy per molecule, $(S^n/n-S^1)/k$.

VI. CONCLUSIONS

It has been shown that the modified Rosenbluth scheme can successfully calculate the partition function for clusters up to size 8 for H_2T_4 and up to size 6 for H_4T_4. The results obtained are in good agreement with data extracted from Metropolis *NVT* simulations, and allow an unambiguous identification of the entropy of packing within a micelle. This validation of the analysis of the Metropolis data supports the use of the Metropolis method to extract the excess entropy and enthalpy to higher cluster sizes than is possible with the Rosenbluth scheme. Analysis of the data obtained in this way yields an insight into the relative importance of the enthalpy and packing entropy in micellization, and these results will form the basis of a future publication.

The Rosenbluth method samples from distributions which become highly skew as the cluster size increases, and this limits the applicability of the current method to relatively small clusters. However, it may be possible to improve the method of cluster growth, within the spirit of the current algorithm, and obtain data for larger cluster sizes. The method described in this paper is also only applicable to enumerating the partition function associated with clusters on a lattice; the method will not be directly applicable to off-lattice systems, since these require the evaluation of an integral rather than a summation. However, it may be possible to adapt the method for use in off lattice configurational-bias Monte Carlo, in a manner analogous to the method described in Ref. [18].

APPENDIX: CLUSTER SURFACE ALGORITHM

The partition function of a cluster of chains grows rapidly with the cluster size because of the large number of possible configuration; approximately 1.5×10^{50} for a cluster of size $10(H_2T_4)$. In order to obtain a reasonable estimate of the partition function, it is therefore necessary to sample a sufficiently large and representative set of these clusters. If the sampling is to be carried out in an acceptable time, it is necessary for the code to be very efficient. The slowest part of the cluster growth code is that which keeps track of the cluster surface, and it is therefore this part which needs the most optimizing. The following algorithm is the result of this optimization, and describes the addition of single chain to a pre-existing cluster.

(1) Grow an amphiphilic chain on a three dimensional lattice as described in Sec. IV B. As the chain is grown, store all of the six nearest neighbors of each placed chain segment in a temporary ''surface'' list (*templist*) as an integer value $index = x_{\text{pos}} l^2 + y_{\text{pos}} l + z_{\text{pos}}$, where x_{pos}, y_{pos}, and z_{pos} correspond to the position on the three-dimensional lattice of the surface site, and l is the lattice box length.

(2) Once a chain has been fully grown, *templist* is merged with a permanent surface list, *permlist*, from which the start of the next chain will be selected. As the lists are merged undertake the following.

(a) Check each surface site in the *templist* to determine if it is occupied. This is done by examining the corresponding lattice site in the *lattice*, a list which records occupied sites on the lattice, using the stored *index* number. The site is not added to the *permlist* if it is occupied.

(b) Check each surface site in the *templist* to determine if it is already present on the *permlist*. This is done using the list κ_m*list*, which gives the κ_m value of the surface site. If the lattice site has not been examined before (i.e., it is not on the *permlist*), then it will have a κ_m value of zero. If the surface site has been visited before then it is not added to the *permlist*; however, the environment of the surface site has changed and therefore this has to be updated; see (2c) and (2d).

(c) Calculate the Boltzmann weights associated with placing either a head or tail at a surface site to be added to the *permlist*. These values are stored in a list, the *boltzlist*, which is linked to the *permlist* using a list called the *sortlist*.

(d) Calculate the change in head-solvent and tail-solvent interactions associated with placing either a head or tail at a surface site to be added to the *permlist*. These values are used in calculating the enthalpy and are stored in the *intlist*, which is linked to the *permlist* by the *sortlist*.

(3) Remove all surface sites in the *permlist* that are now occupied due to the chain just grown.

(4) Update the κ_m values of all the surface sites in the *permlist*.

(a) Surface sites are given a κ_m value, as explained in Sec. IV B.

(b) Surface sites that are no longer available because they have a κ_m value greater than that of the largest remaining κ label are given Boltzmann weights of zero (so that they are never selected or grown into).

(5) Update weights and start growth of next chain.

To summarize, we use a total of three lattice lists and four sequential lists for this algorithm, and they are the following.

(i) *lattice*—This is a lattice which records if a site is occupied by using a number greater than 0 which also identifies the amphiphile present at that site.

(ii) *sortlist*—This lattice gives the appropriate position in the Boltzmann and interaction lists for a specific surface site, expressed as an index number (*index*).

(iii) κ_m*list*—This lattice stores the κ_m values of each surface site. For speed, old sites that are now occupied by chain segments are not reset to zero; only new or changed sites are modified.

(iii) *templist*—This stores all the surface sites of the new chain grown onto the cluster.

(iv) *permlist*—This stores all the surface sites of the cluster, expressed as an index number, available to place the first segment of the next chain.

(v) *boltzlist*—This stores the Boltzmann weight associated with placing either a head or a tail at a certain lattice site, found using the *sortlist*. It is used to calculate the weights and also the probability of placing a segment at that position.

(vi) *intlist*—This stores the change in head-solvent and tail-solvent interactions associated with placing a head or a tail at a certain lattice site. This is used to calculate the enthalpy.

It should be noted that as a chain is grown, surface sites added to the *templist* may be repeated or even grown into. We could precheck every surface site added to the list, but this is computationally expensive, and the checking is only done as the *templist* is added to the *permlist*.

[1] S. W. Haan and L. R. Pratt, Chem. Phys. Lett. **79**, 436 (1981).
[2] R. G. Larson, L. E. Scriven, and H. T. Davis, J. Chem. Phys. **83**, 2411 (1985).
[3] C. M. Care, J. Chem. Soc., Faraday Trans. 1 **83**, 2905 (1987).
[4] K. Rodrigues and W. L. Mattice, J. Chem. Phys. **95**, 5341 (1991).
[5] A. D. Mackie, E. O'Toole, D. A. Hammer, and A. Z. Panagiotopoulos, Fluid Phase Equilibria **82**, 251 (1993).
[6] A. T. Bernardes, V. B. Henriques, and P. M. Bisch, J. Chem. Phys. **101**, 645 (1994).
[7] C. M. Wijmans and P. Linse, Langmuir **11**, 3748 (1995).
[8] C. M. Care, T. Dalby, and J.-C. Desplat, Prog. Colloid Polym. Sci. **103**, 130 (1997).
[9] T. L. Hill, *Statistical Mechanics: Principles and Selected Applications* (Dover, New York, 1956).
[10] M. N. Rosenbluth and A. W. Rosenbluth, J. Chem. Phys. **23**, 356 (1999).
[11] C. M. Care, Phys. Rev. E **56**, 1181 (1997).
[12] L. Pratt, J. Chem. Phys. **77**, 979 (1982).
[13] C. M. Care, J. Phys. C **20**, 689 (1987).

[14] A. D. Mackie, A. Z. Panagiotopoulos, and I. Szleifer, Langmuir **13**, 5022 (1997).
[15] J.-C. Desplat and C. M. Care, Mol. Phys. **87**, 441 (1996).
[16] J. Batoulis and K. Kremer, J. Phys. A **21**, 127 (1988).
[17] P. M. Lam, Phys. Rev. A **34**, 2339 (1986).
[18] D. Frenkel, G. C. A. M. Mooij, and B. Smit, J. Phys. Condens. Matter **3**, 3053 (1991).