Metropolis Monte Carlo simulation of lattice animals

J. Phys. A: Math. Gen. **30** (1997) 8035–8066. Printed in the UK PII: S0305-4470(97)85111-6

Metropolis Monte Carlo simulation of lattice animals

E J Janse van Rensburg[†] and N Madras[‡]

Department of Mathematics and Statistics, York University, North York, Ontario M3J 1P3, Canada

Received 17 June 1997

Abstract. We propose and study a metropolis Monte Carlo algorithm for the simulation of weighted lattice animals in $\mathcal{Z}^d$ in the canonical ensemble (with a fixed number of edges). We examine the implementation and performance of the algorithm. Data obtained by sampling uniformly weighted animals, and animals weighted as critical-percolation clusters, are reported and analysed. In particular, we estimate autocorrelation times and dynamical exponents to study the efficiency of the algorithm, and estimate the metric and branch exponents of lattice animals in two and three dimensions.

1. Introduction

Connected subgraphs of a lattice are called lattice animals, a name which dates back to the 1950s and studies of cell-growth problems (see for example [1]). The finite clusters in lattice percolation are (weighted) lattice animals, and lattice animals are also used as a model for branched polymers [2]. In this paper we study lattice animals in the square and cubic lattices, $\mathcal{Z}^d$, for $d = 2, 3$. In particular, we will be interested in the efficient generation of lattice animals by a metropolis Monte Carlo algorithm.

Two animals are identical if they can be superimposed by a translation. The *order* of an animal is the number of distinct lattice sites occupied by the animal, and the *size* of an animal is the number of edges it contains. The number of distinct animals of a given order, or of a given size, is the most fundamental quantity in the study of animals. Let a_n be the number of animals with n edges, then in $\mathcal{Z}^2$, $a_1 = 2$, $a_2 = 6$, $a_3 = 14$, and so on. These animals are *weak embeddings* or subgraphs of the lattice, and they are also called *edge animals* or *bond animals* counted by their size[§]. There is a close connection between percolation and animals counted by their size [3–5].

There is a large body of literature devoted to lattice animals. The focus in the vast majority of studies is the calculation or approximation of critical exponents. In particular, the sequence a_n is expected to have the following asymptotic behaviour:

$$a_n \sim n^{-\theta}\lambda^n \tag{1.1}$$

where θ is a critical exponent called the *entropic exponent* and λ is the *growth constant* (which determines the exponential rate at which the number of animals a_n grows). Equation (1.1) is the result of the limit $\lim_{n\to\infty}(\log a_n)/n = \log\lambda$, which exists [6] (see

[†] E mail address: rensburg@mathstat.yorku.ca
[‡] E mail address: madras@mathstat.yorku.ca
[§] We use n to indicate the number of edges, and v to indicate the number of vertices in this manuscript, in contrast to the usual notation.

also the work of Klein [7]); the power-law correction to the exponential factor is strongly supported by the numerical simulation of animals [8]. The *metric exponent* of animals has also been studied in much detail. This exponent defines a length scale for animals: If R_n is the root of the mean-square radius of gyration of animals with n edges, then it is expected that

$$R_n \sim n^{\nu}. \tag{1.2}$$

Critical dimensions and mean-field values of exponents were obtained by an ϵ-expansion study of the critical behaviour of lattice animals in good and θ-solvents [2]. The critical dimension of animals in a good solvent is eight, and above this the mean field exponents of animals are encountered: these are $\theta = \frac{5}{2}$ and $\nu = \frac{1}{4}$ (see also [4, 9–12]). In a θ-solvent the critical dimension is six. Incidentally, the critical dimension of percolation is also six, and we shall later see that there is an intimate connection between percolation and animals in a θ-solvent [13–15]. The dimensional reduction of lattice animals in d dimensions to an Ising model in an imaginary magnetic field in $d-2$ dimensions has produced some 'exact values' for θ and ν: In two dimensions, $\theta = 1$, and in three dimensions, $\theta = \frac{3}{2}$ and $\nu = \frac{1}{2}$ [16].

The limiting free energy and phase diagram of a model of interacting lattice animals have also been studied extensively. The existence of a limiting free energy in various models of site animals, and studies describing some of the properties of the free energy can be found in [17–20]. In all these models the free energy is a function of a cycle fugacity [21]. Increasing the cycle fugacity in these models gives animals rich in cycles more weight, and at a critical value of the fugacity the model is thought to undergo a 'collapse transition' to compact animals. It has also been established that a non-analyticity exists in the free energy of a directed version of this model, presumably corresponding to a collapse transition [22]. For weakly embedded animals (counted by vertices) the limiting free energy as a function of solvent contacts (or perimeter of the animal) has also been shown to exist [23]. A related model of weakly embedded animals with a cycle fugacity has been studied in [24], where several properties of the free energy have been proven (see also [13]).

Numerical work on animals falls into two broad categories: series enumeration and Monte Carlo simulations. Counting animals for series analysis has its origin in the study of percolation [25–28]. Later series enumerations for applications to models of animals include those in [8, 17, 29–37]. The first simulations of lattice animals were in studies of percolation. These include cluster growth methods [38–40] where a vertex is selected and the animal (weighted like a percolation cluster) is grown from this 'seed'. A second set of Monte Carlo simulations is of the metropolis type: a random change is made to a percolation cluster and the resulting cluster is proposed as an updated version of the old cluster. It is accepted (or rejected) by throwing a random number, metropolis style. This is a stochastic process which samples along a Markov chain in the state space of animals (weighted as percolation clusters in the cases where percolation is studied). Examples of such algorithms can be found in the work of Stauffer [41], Herrmann [42], and Peters *et al* [31] in the 1970s and followed by the work of Stratychuk and Soteros [43]. The new algorithm proposed in this paper will be of the dynamic metropolis Monte Carlo type, and it will resemble a non-local algorithm for lattice trees introduced in [44, 45]. Other Monte Carlo algorithms for models of branched polymers also focussed on trees (rather than animals); these include the work of Redner [46], Seitz and Klein [47] and Meirovitch [48]. For a detailed account and more references, see [44].

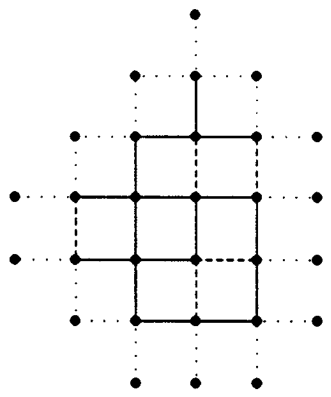

Figure 1. An animal in the square lattice with 15 vertices, 16 edges, two cycles, five contacts and perimeter 18. The contacts are indicated by broken lines, the perimeter edges are indicated by dotted lines and edges are indicated by full lines.

2. Lattice animals

In this section we review what is known about the phase diagram of animals with cycle and contact fugacities. Let $a_n(c,k)$ be the number of animals with n edges, c cycles and k contacts. In addition to these, we also identify perimeter edges (or 'solvent contacts') as those edges of the lattice which have exactly one endpoint in the animal. Let s be the number of perimeter edges in an animal. These are all illustrated in figure 1.

Each vertex in an animal is incident with at least one edge in the animal, and then with a number of other contact and perimeter edges. Since there are v vertices, each with coordination number $2d$, we observe that [13]:

$$\begin{aligned} 1 &= v - n + c \\ 2dv &= 2n + 2k + s. \end{aligned} \tag{2.1}$$

In other words, by specifying three (one of which is s or k) of the five numbers $\{c,k,v,n,s\}$, we can compute the other two. In the edge-cycle-contact model, which we will be concerned with in this paper, the number of vertices and perimeter edges are also determined for each animal. The partition function of a model of animals with fugacities given to the numbers of cycles and contacts is

$$Z_n(\beta_{\rm c},\beta_k) = \sum_{c,k} a_n(c,k) \mathrm{e}^{\beta_{\rm c} c + \beta_k k}. \tag{2.2}$$

The values of $\beta_{\rm c}$ and β_k in $Z_n(\beta_{\rm c},\beta_k)$ can be chosen such that the animals are weighted as percolation clusters at a probability p that each edge is open. We see this by computing the probability $P_n(p)$ that the cluster at the origin has n edges [49] (see also [13, 14]):

$$P_n(p) = \sum_{c,k} v\ a_n(c,k) p^n (1-p)^{s+k}. \tag{2.3}$$

Using the relations in (2.1) to eliminate v and s in the above we obtain

$$P_n(p) = p^n(1-p)^{2d+2(d-1)n} \sum_{c,k} (n+1-c) a_n(c,k)(1-p)^{-2dc}(1-p)^{-k}. \tag{2.4}$$

Comparing this with (2.2) one notes that this is an unnormalized average of v with p related to $\beta_{\rm c}$ and β_k. In fact, for each value of p one can compute $\beta_{\rm c}$ and β_k from

$$\begin{aligned} \beta_{\rm c} &= -2d\log(1-p) \\ \beta_k &= -\log(1-p). \end{aligned} \tag{2.5}$$

These relations define a half-line in the (β_k, β_c)-plane along which animals are weighted as percolation clusters at edge probability p. The *free energy* per edge of this model is defined as

$$F_n(\beta_c, \beta_k) = \frac{1}{n} \log Z_n(\beta_c, \beta_k). \tag{2.6}$$

The existence of a limiting free energy (in the limit $n \to \infty$) was proven in a variety of models of trees and animals [7, 24, 50, 51]. Since our model has not been studied directly before, we prove here that there exists a limiting free energy, and that it is convex in both its arguments.

Theorem 1. There exists a function $\mathcal{F}(\beta_c, \beta_k)$, convex in both its arguments, such that

$$\mathcal{F}(\beta_c, \beta_k) = \lim_{n\to\infty} F_n(\beta_c, \beta_k)$$

for all $\beta_c \in [-\infty, \infty)$ and $\beta_k \in [-\infty, \infty)$. Moreover, $\mathcal{F}(\beta_c, \beta_k)$ is finite for all $\beta_c \in [-\infty, \infty)$ and $\beta_k \in [-\infty, \infty)$.

Proof. Let α_1 and α_2 be two animals and suppose α_i has n_i edges, c_i cycles and k_i contacts. The top and bottom vertices of α_i are found by a lexicographic ordering of the vertices by their coordinates. Translate α_2 so that its bottom vertex has first coordinate one bigger than the first coordinate of the top vertex of α_1, and all other coordinates equal. Define $c = c_1 + c_2$ and $k = k_1 + k_2$. We can concatenate α_1 and α_2 by adding a new edge between the top vertex of α_1 and the bottom vertex of α_2, and we obtain a new animal with c cycles, k contacts and $n_1 + n_2 + 1$ edges. Since this animal is uniquely determined by α_1 and α_2, we can let c_1 and k_1 vary over all values between 0 and c and k respectively, with c_2 and k_2 chosen to keep c and k constant; this gives the generalized supermultiplicative inequality:

$$\sum_{c_1,k_1} a_{n_1}(c_1, k_1) a_{n_2}(c - c_1, k - k_1) \leqslant a_{n_1+n_2+1}(c, k).$$

Multiply this equation by $e^{\beta_c c + \beta_k k}$ and sum over c and k. This gives, by (2.2):

$$Z_{n_1}(\beta_c, \beta_k) Z_{n_2}(\beta_c, \beta_k) \leqslant Z_{n_1+n_2+1}(\beta_c, \beta_k)$$

and $Z_{n-1}(\beta_c, \beta_k)$ is a supermultiplicative function. Consequently $F_{n-1}(\beta_c, \beta_k)$ is a superadditive function, and the limit in question exists [52]. To see that it is finite, note that

$$Z_n(\beta_c, \beta_k) \leqslant a_n e^{2(d-1)n\beta_c^*} e^{2dn\beta_k^*}$$

since the maximum number of cycles in an animal is at most $2(d-1)n$, and the maximum number of contacts is at most $2dn$, and where $\beta_c^* = \max\{\beta_c, 0\}$ and $\beta_k^* = \max\{\beta_k, 0\}$.

Convexity of the free energy is shown by standard applications of the Cauchy–Schwartz inequality. $\square$

By theorem 1, $\mathcal{F}(\beta_c, \beta_k)$ is continuous and differentiable almost everywhere [53]. There is at least one point in the (β_k, β_c)-plane where we know that $\mathcal{F}(\beta_c, \beta_k)$ is non-analytic: Edge percolation is a half-line in this plane, parametrized by p and given by equations (2.5). Since $P_\infty(p)$, the percolation probability, is non-analytic at p_c [49], we conclude that the point $(-2d \log(1 - p_c), -\log(1 - p_c))$ is a non-analytic point of $\mathcal{F}(\beta_c, \beta_k)$. The rest of the phase diagram of $\mathcal{F}(\beta_c, \beta_k)$ is conjectured and described in [13, 15] (see figure 2). There is a line of θ-transitions in this diagram which separates a phase of expanded animals (or animals in a good solvent) from collapsed animals (or animals in a poor solvent). The θ-transitions

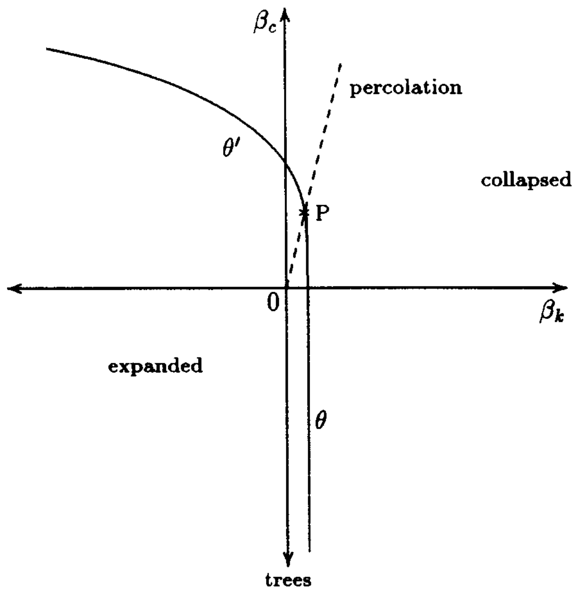

Figure 2. The conjectured phase diagram for animals in the contact-cycle ensemble. The percolation point (marked with a P) is known to be a non-analyticity in the free energy. Two lines of distinct θ-transitions (θ and θ') are thought to meet at the percolation point (these are lines of tricritical points). This makes the percolation point a multicritical point. Uniformly weighted lattice trees are at $\beta_c = -\infty$ and $\beta_k = 0$ in this diagram, and they undergo a collapse transition as β_k approaches a critical value. Percolation clusters are found along the broken line marked percolation which starts at the origin. In two dimensions, $p_c = \frac{1}{2}$, and a numerical estimate of the critical value of β_k in lattice trees implies that (at least in two dimensions), the line θ may be a straight line.

should include the critical percolation point, since all other points on the percolation line are believed to be analytic†. There is a conjecture that the percolation point separates the line of θ-transitions into two universality classes [13, 15, 45, 54, 55], each characterized by its set of tricritical exponents (the θ-transition is believed to be tricritical). In the case that β_c is small or negative, the θ-transition is believed to be into a collapsed phase poor in cycles, while for large β_c, the collapse is to a phase rich in cycles. The θ-transition for collapse into a cycle-rich phase is suggested to be in the universality class of the Ising model [54, 55], with critical exponents determined by that critical point‡. It is not obvious that these θ-phases are indeed different phases (for example, they have the same metric exponent). In this picture, the percolation point is a multicritical point where two lines of distinct tricritical points meet, and associated with it is a set of multicritical exponents (which are the percolation exponents). All of the above is conjecture, with the exception that the location of the percolation point is known in two dimensions, and that $\mathcal{F}(\beta_c, \beta_k)$ is

† $\mathcal{F}(\beta_c, \beta_k)$ is conjectured to be non-analytic at θ-points along any line in the phase diagram which intersects the line of θ-transitions. It is believed that there are no other non-analyticities along the percolation line (except at the critical percolation point; the percolation probability is known to be continuous along the percolation line, except for possibly at the critical point in three and more dimensions) [49]. Thus, we expect that the critical percolation point is at the intersection of the percolation line and the line of θ-transitions.
‡ We reserve judgement on collapse into a cycle-poor phase; it is possible that this is also in the Ising universality class, but simulations for lattice trees [45, 54] suggests a value of the crossover exponent ϕ which is slightly smaller than predicted by the Ising universality class [56] (see also [55]).

non-analytic at this point.

In this paper we present an algorithm for studying the above problem. In particular, we test the algorithm by simulating animals in the expanded phase at $(0, 0)$ and at the critical percolation point in two and three dimensions. We focus in particular on the numerical efficiency of the algorithm, and on the scaling behaviour of animals as observed through various properties that we can measure by Monte Carlo. The numerical efficiency will be tested by measuring autocorrelation times on some properties associated with the animals, and we will discuss these in section 4. The properties of animals that we measure includes the following.

2.1. Mean square radius of gyration

The algorithm will simulate animals with a constant number of edges. The number of vertices will change as the cyclomatic index of the animals changes during the simulation. The size (number of edges) of the animal will therefore be taken as a measure of the 'mass' of the animal. Thus, let every edge e_i, $i = 1 \ldots n$, be taken to have unit mass, concentrated at its midpoint. The mean square-radius of gyration, R_n^2, of an animal with n edges is defined by

$$R_n^2 = \frac{1}{n} \sum_{i=1}^{n} (\boldsymbol{r}(e_i) - \boldsymbol{r}_{\rm c})^2 \tag{2.7}$$

where $\boldsymbol{r}_{\rm c}$ is the centre-of-mass of the animal and $\boldsymbol{r}(e_i)$ the position vector of the midpoint of the ith edge. The mean-square radius of gyration, $\langle R_n^2 \rangle$, is the average of R_n^2 over all animals. Naturally, $R_n^2 \leqslant n^2$, and $\langle R_n^2 \rangle \geqslant C n^{2/d}$ in d dimensions, where C is a constant. Thus, we expect that the mean-square radius of gyration scales as

$$\langle R_n^2 \rangle \sim n^{2\nu} \tag{2.8}$$

where ν is the metric exponent, and $1/d \leqslant \nu \leqslant 1$.

The metric exponent takes on different values in the phases identified in figure 2. In particular, in the collapsed phase we expect that $\nu = 1/d$, whereas in the expanded phase it takes on the universal branched polymer value. At the percolation point one expects its value to be that of percolation, and it has yet other values on the θ and θ' lines.

2.2. Mean span

Let $x_i(v_j)$ be the ith Cartesian coordinate of the jth vertex of the animal. The span of an animal with n edges and m vertices $\{v_i\}_{i=1}^{m}$ is defined by

$$S_n = \frac{1}{d} \sum_{k=1}^{d} \max_{1 \leqslant i,j \leqslant m} |x_k(v_i) - x_k(v_j)|. \tag{2.9}$$

If we accept that there is only one length scale in this model, then we conclude from equation (2.8) that the mean span behaves as

$$\langle S_n \rangle \sim n^{\nu}. \tag{2.10}$$

Equations (2.8) and (2.10) provide us with two different methods for estimating ν. Since corrections to scaling tends to obscure the true value of ν, a comparison between values obtained from these two quantities are useful in assessing the accuracy of an estimate of ν.

2.3. Mean branch size

Let e_i be a cut edge of an animal α. Then $\alpha - e_i$ is a disconnected graph of two components, called *sub-animals*. We call the smaller sub-animal a *branch* and let b_n be its order. Then surely $b_n \leqslant n$. The expected value of b_n is defined by taking its average over all branches in all animals of size n. We define the exponent ϵ by

$$\langle b_n \rangle \sim n^{\epsilon}. \tag{2.11}$$

$\langle b_n \rangle$ can be estimated by uniformly choosing a cut edge and considering the resulting branches in a uniform sample of animals of size n. We can simplify this by choosing an edge uniformly in each animal and, if a selected edge is not a cut edge, by defining the resulting branch to have size zero. If the probability that a cut edge is selected is q_n in animals of size n, then the expected number of vertices in a branch will be

$$\langle B_n \rangle \sim q_n b_n + (1 - q_n)0 \sim q_n n^{\epsilon}. \tag{2.12}$$

We do not expect q_n to approach zero as $n \to \infty$†, even on the lines of θ-transitions, and so we conclude that

$$\langle B_n \rangle \sim n^{\epsilon}. \tag{2.13}$$

The fact that animals are believed to belong to the same universality class as trees indicates that the value of ϵ should be given by the branch exponent of trees. The number of edges in the longest path in a lattice tree, ρ_n, is expected to scale as $\langle \rho_n \rangle \sim n^{\rho}$. A heuristic argument in [44] shows that the mean branch size of trees also scales with the exponent ρ. Thus, we conclude that $\epsilon = \rho$ in (2.13). The mean-field value of ρ is $\frac{1}{2}$ [44], and numerical results indicate that it is larger than $\frac{1}{2}$ in dimensions lower than the critical dimension. Similarly to the metric exponent, we expect ϵ to assume different values in the different phases in figure 2.

2.4. Mean number of contacts

The mean number of contacts, $\langle k_n \rangle$, is the contact energy of the animal, since it is the derivative of the free energy to β_k. A pattern theorem for animals (see footnote below) implies that $\langle k_n \rangle \geqslant K'n$, where K' is a constant. On the other hand, since each vertex has maximum degree at most $2d$, we also conclude that $\langle k_n \rangle \leqslant 2dn$. Consequently, we expect

$$\langle k_n \rangle \approx Kn \tag{2.14}$$

where K is a constant. Since the free-energy density in animals seems to undergo a continuous transition as β_k increases through its critical value, there is a non-analyticity in K at the collapse transition.

2.5. Mean number of cycles

The same arguments (made for the mean number of contacts) apply here. Thus, we expect that

$$\langle c_n \rangle \approx Cn \tag{2.15}$$

† Neither does q_n approach 1 as $n \to \infty$. This is a consequence of a 'pattern theorem' for animals (this result is due to Madras, unpublished), which implies that there will be a density of cycles of length 4 in the animal. In other words, the probability $1 - q_n$ is not expected to approach 0 as $n \to \infty$. On the other hand, (expanded) animals are also expected to scale as trees, which suggests that there is a non-zero density of cut edges.

where $\langle c_n \rangle$ is the mean number of cycles and C is a constant. Observe that the mean number of vertices in an animal is given by $n - \langle c_n \rangle + 1$, so that it is equivalent to the mean number of cycles (by equation (2.1)).

2.6. Mean perimeter

The pattern theorem for animals indicates that for uniformly weighted animals, the number of perimeter sites will grow proportionally to the number of edges. We expect this to be true for all animals in the expanded phase in figure 2. Thus

$$\langle s_n \rangle \approx An \tag{2.16}$$

for some constant A.

We define the *total perimeter* of a lattice animal to be sum of its contacts and perimeter edges. There is a lengthy history associated with the total perimeter of percolation clusters [28]. Let t_n be the total perimeter of a percolation cluster. Then close to the critical value p_c of the edge probability p, t_n is expected to scale as $t_n = ((1-p)/p)n + n^\sigma f((p-p_c)n^\sigma)$, where σ is a 'crossover' exponent which describes the crossover to critical behaviour as p approaches p_c. The difference $t_n' = t_n - ((1-p)/p)n$ is called the *excess perimeter*†. The excess perimeter plays the role of a surface area in percolation, and the approach to criticality is often described in terms of the condensation of a liquid to droplets at $p = p_c$ into drops for $p > p_c$. For $p > p_c$ the excess perimeter scales as the surface area of a liquid sphere of volume n: $t_n' \propto n^{1-1/d}$ in d dimensions. This behaviour breaks down at p_c, and scaling arguments (see [28] for details) show that $t_n' \propto n^\sigma$, and $\sigma - 1$ may be interpreted as the negative inverse of a 'fractal dimension' of the cluster at $p = p_c$. For $p < p_c$ this picture does not apply, instead, the excess perimeter will scale proportionally to n, and we cannot interpret this as the contribution of a surface area (of a drop) to the perimeter.

The arguments above have some interesting implications for us: our expanded animals are in the same phase in figure 2 as subcritical percolation clusters, and we can expect that the perimeter will scale as in (2.16) above. For critical percolation clusters, we obtain the additional surface correction, so we postulate that

$$\langle s_n \rangle \approx An + Bn^\sigma \tag{2.17}$$

at the critical percolation point. Moreover, if we consider the total perimeter of our clusters by considering instead $\langle t_n \rangle = \langle s_n + k_n \rangle$, then at the percolation point we should obtain $\langle s_n + k_n \rangle / n \to [(1-p_c)/p_c]$ as $n \to \infty$, and we should be able to compute the critical value of the percolation probability from our data.

2.7. The mean number of edges per cycle

Suppose that an edge e is selected uniformly from the animal. Let C_n be the size of the smallest cycle containing e (say that a cycle of size zero is found if a cut edge is selected). Since expanded animals scale like trees, we expect e either to be not in a cycle, or to be in a small cycle; thus we expect that $\langle C_n \rangle \to$ constant as $n \to \infty$. This argument does not apply to critical percolation clusters.

† The term $((1-p)/p)n$ is a 'bulk term', describing the contributions from the bulk of the cluster. It is obtained by defining the generating function of percolation clusters at edge probability p: $\mathcal{Z}_n(p) = \sum_{c,k} a_n(c,k) p^n q^{s+k}$ where $q = 1-p$, following (2.3). Direct computation shows that $\langle s_n + k_n \rangle = \frac{q}{p} n + q \frac{\mathrm{d}}{\mathrm{d}q} \log \mathcal{Z}_n(p)$. The usual finite size scaling assumption $\mathcal{Z}_n(p) \sim g((p-p_c)n^\sigma)$, where σ is a crossover exponent describing the crossover to critical behaviour and $g(x)$ is a universal scaling function, gives the n and p dependence of t_n. The exponent σ is estimated to have a value close to 0.4 in both two and three dimensions [28]; we will estimate its value later from our numerical data.

2.8. The fraction of vertices of degree i

A pattern theorem for animals implies that vertices of any given degree will occur with positive density in the limit $n \to \infty$. Let δ_i be the density of vertices of degree i. The mean valence is $V = \sum i\delta_i$. An observable closely related to V and δ_i has been studied in [29, 30] for percolation clusters.

3. Metropolis Monte Carlo of animals

There are a large number of numerical studies of animals and percolation clusters in the physics and mathematics literature. The vast majority of these studies are exact enumeration of animals or percolation clusters with a given number of edges or vertices (sites) [57–59, 8, 21].

Monte Carlo studies of percolation were already done in the early 1960s [60]. Dean and Bird [61] used a Monte Carlo algorithm to compute extensive tables of critical probabilities for percolation. Their algorithm proceeds by selecting an unoccupied site in a finite lattice uniformly. Turning the site on changes the distribution of clusters (animals) in the lattice. The distribution is recorded and the process is repeated. Leath's algorithm [38, 39] grows a weighted percolation cluster by selecting a perimeter site of the cluster uniformly and turning it on. The algorithm of Redner [46] also grows animals by choosing an active site in the animal and 'growing' a new set of branches at this site. These algorithms can be thought of as 'static' Monte Carlo algorithms.

In contrast to these static algorithms, dynamic Monte Carlo algorithms sample along a Markov chain in the state space of (weighted) animals. Stoll and Domb [62] used Monte Carlo simulation of the Ising model with a one-spin flip to accumulate statistics on percolation clusters. Stauffer's algorithm [5] is a dynamical Monte Carlo algorithm which generates clusters with a fixed number of vertices. The elementary move is the exchange of a cluster site with a perimeter site (both selected uniformly). The newly proposed cluster is accepted metropolis style with probability $(1-p)^{\Delta t}$ where p is the probability that a site is on, and Δt is the change in the perimeter length of the animal [28, 41]. This algorithm is particularly important in the development of animal simulations. By taking $p = 0$, one can simulate (uniformly weighted) animals (these are at the origin of the graph in figure 2), and properties of animals and percolation clusters can in principle be efficiently simulated for any value of p. In fact, it is apparent that by adapting the transition probability, one can weight animals with respect to cycles, or contacts, or any intrinsic property; and p can also be taken negative. In that case the algorithm will sample along a Markov chain of animals weighted to have few cycles and few contacts (this is noted by continuing the percolation line in figure 2 by taking p negative in equations (2.5)).

All the algorithms described above are defined to simulate site-percolation or site-animals. In some of these (Dean and Bird, Leath, and Stauffer's algorithms), it is not a difficult problem to adapt the algorithm to similate edge animals and edge or bond percolation. In addition, there is an efficient algorithm for edge animals in the literature [63]. This algorithm does not preserve the size of the animals, and incorporates a global move of the type used for trees in [44]. The invariant limit distribution is

$$\frac{1}{Z(\beta_1, \beta_2, \beta_3)} \mathrm{e}^{\beta_1 s + \beta_2 k + \beta_3 v} \tag{3.1}$$

where the β_i are weights conjugate to the perimeter s, the contact number k and the number of vertices v in the animals. Approximations of expected values of properties of animals can be computed with respect to this distribution.

3.1. A non-local algorithm for lattice animals

In this section we explain a new algorithm for lattice animals with a fixed number of edges. The algorithm will have the non-local character of the algorithm for trees in [44], and it will sample animals weighted with a cycle and a contact fugacity (in other words, it will sample animals from the phase diagram in figure 2 for finite values of n). We are primarily interested in the numerical behaviour of the algorithm, and so we will simulate animals only uniformly (at the origin in figure 2), or weighted like percolation clusters (at the percolation point in figure 2).

The metropolis Monte Carlo algorithm for animals below will sample along a Markov chain in the state space of animals with n edges. It will attempt to find the next state in the chain by means of an elementary move on the current state. To simplify matters, we first discuss sampling along a symmetric chain, converging to the uniform distribution on the state space. Suppose that we have an animal α_1 with n edges, k contacts, v vertices and c cycles. The algorithm proceeds as follows.

Algorithm S.

(1) Pick an edge e uniformly in the animal.

(2a) If e is a leaf: delete e and select a vertex V uniformly in $\alpha_1 - e$ (and do not select the isolated point). Choose a nearest neighbour U of V uniformly. If UV is an edge in $\alpha_1 - e$, then we reject this attempt, count α_1 as the next state and start again at step 1. If UV is a perimeter edge, then we move e there, creating a new animal $\alpha_2 = \alpha_1 - e + UV$ which we propose as the next state. If UV is a contact, then we choose $\alpha_2 = \alpha_1 - e + UV$ with probability $\frac{1}{2}$ as the next proposed state, otherwise we reject the attempt and return to step 1 (after counting α_1 as the next state).

(2b) If e is an edge in a cycle: delete e and select a vertex V uniformly in $\alpha_1 - e$. Choose a nearest neighbour U of V uniformly. If UV is an edge in $\alpha_1 - e$, then reject this attempt, and count α_1 as the next state and return to step 1. If UV is a contact or a perimeter edge, then accept $\alpha_2 = \alpha_1 - e + UV$ as the next proposed state.

(2c) *(Tree-move)* otherwise e is a cut edge and $\alpha_1 - e$ consists of two sub-animals: Let A_1 be the smaller of these sub-animals. We attempt to reattach A_1 in order to create a new animal. Rotate A_1 to A_1' by operating on it with a randomly chosen element of the octahedral group. Pick two vertices at random, one on each sub-animal. Translate A_1' so that the two chosen vertices are nearest neighbours (in one of $2d$ possible orientations). If there are intersections between the two sub-animals, then we reject this attempt (and count α_1 as the next state and return to step 1). Otherwise we reconnect the animal by adding an edge between the selected vertices to create α_2.

(3) Accept α_2 as the next state in the Markov chain and start again at step 1.

It is not immediately obvious that this algorithm is symmetric. There are several possible situations. Suppose that α_1 and α_2 are distinct states following one another in the Markov chain. Then we obtain α_2 from α_1 by exactly one of the following: (1) a tree move, (2) replacing a leaf by another leaf, (3) removing a leaf and creating a cycle (by turning a contact into an edge), (4) removing an edge from a cycle and creating a leaf, (5) removing an edge from a cycle and create another cycle by changing a contact into an edge. We can now explicitly compute transition probabilities for each of these cases.

(1) Suppose that the two sub-animals created in the tree move have $v - k$ and k vertices. Then if the octahedral group has o_h elements,

$$P(\alpha_1 \to \alpha_2) = \frac{1}{2dnk(v-k)o_h} \tag{3.2}$$

since the edge is selected with probability $1/n$, the two vertices are selected on the sub-animals in one of $k(v-k)$ ways, and the smaller is translated for reconnection in one of $2d$ ways, and rotated in one of o_h ways. The probability for obtaining α_1 from α_2 by reversing every operation here is also given by (3.2).

(2) The probability of obtaining α_2 from α_1 is

$$P(\alpha_1 \to \alpha_2) = \frac{1}{2dn(v-1)} \tag{3.3}$$

since the edge is selected with probability $1/n$, and removing it leaves $(v-1)$ vertices from which we select a new edge in one of $2d(v-1)$ ways. Selecting a leaf in α_2 to create α_1 follows exactly the same probabilities. Hence, this move is symmetric.

(3, 4) Suppose that α_2 is obtained from α_1 by selecting a leaf e and putting it back to create a cycle. The probability of selecting the leaf is $1/n$, and the probability of selecting a contact in $\alpha_1 - e$ is $2/2d(v-1)$, since their are $(v-1)$ vertices in $\alpha - e$ and the contact can be selected from either of its endpoints. In addition, this elementary move is accepted with probability $\frac{1}{2}$, so that the transition probability is

$$P(\alpha_1 \to \alpha_2) = \frac{1}{2dn(v-1)}. \tag{3.4}$$

α_1 can be obtained from α_2 by selecting the edge out of the cycle, and putting it back as a leaf. The edge is selected with probability $1/n$, and since α_2 has $(v-1)$ vertices, the probability of selecting the perimeter edge to put down the leaf is $1/2d(v-1)$. Consequently, the probability of changing α_2 back into α_1 is given by (3.4), and these cases are also symmetric.

(5) In this case, α_2 is obtained by moving an edge from a cycle in α_1 to create a new cycle. Note that α_1 and α_2 have the same number of vertices v. Thus, the transition probability is

$$P(\alpha_1 \to \alpha_2) = \frac{2}{2dnv} \tag{3.5}$$

since each edge can be selected in n ways, and a contact in $\alpha_1 - e$ is selected in two ways from $2dv$. The transition probability from α_2 to α_1 is calculated in exactly the same way, also giving (3.5). Thus, this move is also symmetric.

Consequently, algorithm S is symmetric and aperiodic. We note that it is irreducible since any animal can be turned into a straight line by selecting edges in cycles and leaves and apending them (as leaves) onto the top vertex (lexicographic most vertex) of the animal in, for example, the x-direction. By the fundamental theorem of Markov chains, we see that the invariant limit distribution of algorithm S is the uniform distribution in the space of lattice animals with n edges in the hypercubic lattice [64].

Our aim is to sample animals weighted as in the summand of equation (2.2). To this end, we modify algorithm S as follows by introducing two parameters β_c and β_k and by replacing step 3 by step 3′ to define algorithm R.

Algorithm R.

Steps 1 and 2 are identical to steps 1 and 2 of algorithm S.

(3′) Suppose that α_2 has c' cycles and k' contacts. Let $\Delta c = c' - c$ and $\Delta k = k' - k$, where α_1 has c cycles and k contacts. Accept α_2 as the next state in the Markov chain with probability

$$\min\{1, \mathrm{e}^{\beta_c \Delta c + \beta_k \Delta k}\}. \tag{3.6}$$

If α_2 is rejected then α_1 is the next state in the Markov chain.

The fundamental theorem for Markov chains then implies that the invariant limit distribution of algorithm R is

$$\Pi(\alpha) = \frac{e^{\beta_c c(\alpha)+\beta_k k(\alpha)}}{Z_n(\beta_c, \beta_k)} \tag{3.7}$$

where $Z_n(\beta_c, \beta_k)$ is a normalizing constant given by equation (2.2) [64].

3.2. Implementation of algorithm R

Algorithm R was coded in C with data structures carefully designed to achieve an efficient implementation. Suppose that the animal has v vertices in d dimensions. Implementation of the algorithm requires the following operations in the animal: (1) detection of cycles and sub-animals, (2) detection of self-intersections and (3) counting contacts and cycles. The vertices of the animal were stored in an array $V(v, 3d)$. The first d addresses in the ith row of V keep the coordinates of the ith vertex in the animal. The remaining $2d$ addresses are pointers which point to vertices adjacent to vertex i in the animal. The degree of the ith vertex is given by the number of pointers in these addresses (some of these will be empty if the degree of vertex i is less than $2d$).

We detected self-intersections in the animal, and counted the number of contacts, by using hash coding (see [65]). Vertices of the current animal are put into a hash table $HASH(10 * n, d + 1)$ using a hash function and linear probing (see [66] for details). The vertices are kept in the hash table for most of the simulation, and stored by writing coordinates into the first d positions, and labels into the $(d + 1)$th position. An array $SUB(2, \lfloor v/2 \rfloor, d + 1)$ was used to store the coordinates of vertices in sub-animals in the case that a tree move is proposed. Vertices are read into $SUB(, ,)$ as they are encountered in a double-breadth first search† for the smaller sub-animal. The first d positions in an address in $SUB(, ,)$ will be occupied by the coordinates of vertices in the sub-animal, and the $(d + 1)$th position will be for its label. Using these data structures, the implementation of the algorithm was as follows:

(1) An edge e can be selected uniformly from $V(v, 3d)$ by selecting uniformly two random integers in $[1, n]$ and $[1, 2d]$ respectively. The first selects a vertex uniformly, and the second a pointer to another vertex. If the pointer is empty, then we reject the attempt and try again by selecting two new random integers.

(2) Determine whether e is a leaf by checking the degrees of its endpoints. If e is a leaf then step (2a) is executed next. Otherwise we must decide if e is in a cycle. This is done efficiently by a breadth-first search in the animal, starting at an endpoint of e (see for example [67], and see [44] for details). An advantage of a breadth-first search (over a depth-first search) is that it will detect a smallest cycle containing e efficiently. At the same time, we can collect data on the size of smallest cycles in the animal. If e is in a cycle, then step (2b) is executed next, otherwise e is a cut edge of the animal, and a tree move can be attempted by executing step (2c). The breadth-first searches are done directly in $V(v, 3d)$ by searching from vertex to vertex along the pointers; the implementation is quite natural. Notice that we write vertices detected by the bread first search into $SUB(, ,)$ *presuming* that we will do a tree move. If a sub-animal is detected, then we already have a proposed tree

† The double-breadth first search [67] is started at the endpoints of the selected edge, and alternates from side to side, while writing vertices into $SUB(1, ,)$ and $SUB(2, ,)$ until the smaller sub-animal is detected. The labels and coordinates of the smaller animal is then recorded in one of these sub-arrays.

move. If we detect a leaf or a cycle instead, then we abandon the breadth-first search, and execute a leaf or a cycle move.

(2a) Delete the leaf e from the animal by removing a vertex and pointers from $V(v, 3d)$. Next, select a vertex x and a direction in the animal, and suppose that y is the vertex selected opposite x. If y is not in the animal (check this by querying the hash table), then we attempt to add the edge xy to the animal. Proceed to step 3 for the metropolis check: the number of cycles remains the same, but the change in the number of contacts can be counted by checking the hash table. If y is a vertex already in the animal, then there is the possibility that xy is already an edge. If it is, then we have a self-intersection; we reject the move and start again at step 1. Otherwise, addition of xy will create a cycle: reject this attempt with probability 0.5. Apply the metropolis check in step 3 to accept or reject the move.

(2b) Delete the edge by removing pointers from $V(v, 3d)$. Next, select a vertex x and a direction in the animal, and suppose that y is the vertex selected opposite x. If y is not in the animal, then we attempt to append xy to the animal: this move reduces the number of cycles by one, and we query the hash table to count the change in the number of contacts. Accept the attempt by the metropolis check in step 3. If y is in the animal, then xy might be an edge, which gives a self-intersection: reject this attempt. Otherwise, xy is a contact edge. Appending xy to the animal leaves the number of contacts and cycles the same, so we accept the attempt.

(2c) Delete the edge e by removing pointers from $V(v, 3d)$. The breadth-first search has written the labels of the vertices in the smaller sub-animal in *SUB*(, ,), and we must now rotate and translate these vertices in our attempt to perform a tree move. Choose (uniformly) a vertex x from the sub-animal (from the list in *SUB(,,)*), and a vertex y in the rest of the animal (this can be done by uniformly picking y from $V(v, 3d)$ and rejecting it if y is in *SUB(,,)*). Choose a random rotation from the octahedral group; we will rotate the sub-animal about the vertex y. Choose a translation of the sub-animal which will translate y to a random nearest neighbour of x. The vertices in the smaller animal are now rotated and translated and stored in *SUB(,,),* starting from y and using the ordering generated by the breadth-first search. At the same time, we remove the old vertices from the hash table, as they are rotated and translated. Next, we check for self-intersections by adding the rotated and translated vertices to the hash table; if a self-intersection is detected, then the attempt is abandoned, and we restore the hash table before going back to step 1. The change in the number of contacts is found by counting the number of nearest neighbours of vertices in the hash table as the old vertices are removed and the rotated vertices are added. Accept the animal by applying the metropolis check in step 3. If it is rejected, then restore the hash table before returning to step 1.

(3) Apply the metropolis rule as set out in step 3′ and equation (3.6).

The amount of work in the algorithm in each attempt depends on the type of move. We decide in O(1) CPU time if a selected edge is a leaf. We also expect the breadth-first search to terminate quickly if the selected edge is in a cycle (since cycles are likely to be small). Hence, we expect that all moves, except for tree moves, can be executed in O(1) CPU time, as a rough estimate. On the other hand, a sub-animal of size k will be detected in $O(k)$ CPU time by the double-breadth first search. Moreover, both searching for intersections, and accepting or rejecting the move, takes at most $O(k)$ CPU time, if the algorithm is optimally coded. Let $p_n(X)$ be the probability that a move of type X is *attempted*, then the computational complexity of the algorithm is

$$C_A \sim (1 - p_n(T))O(1) + p_n(T)\langle b_n \rangle. \tag{3.8}$$

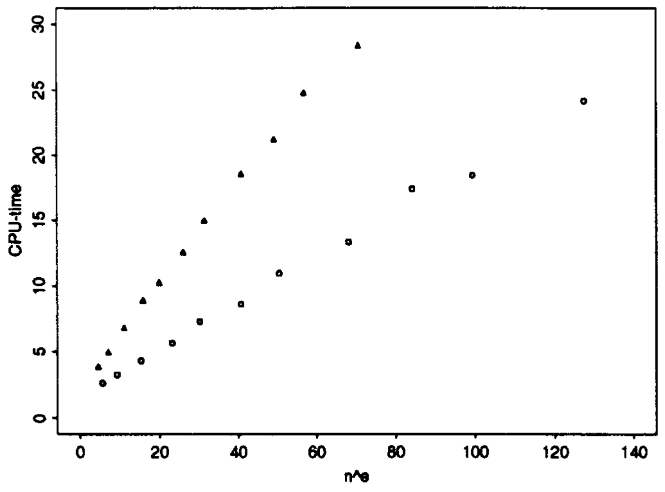

Figure 3. The CPU time taken for 10 000 iterations of the algorithm ($\beta_k = \beta_c = 0$) as a function of n^ϵ in ($\circ$) two dimensions and ($\triangle$) three dimensions.

We expect $p_n(T)$ to converge to a constant between 0 and 1, as $n \to \infty$, and so, by (2.11),

$$C_A \sim n^\epsilon. \tag{3.9}$$

The number of iterations performed for each value of n was $50\,000N$, where $N = 500$ for most values of n (it was taken to be less than 500 for the smallest animals). Data were collected during the run and written to a file every N iterations. This gives a time series of length 50 000 which was subjected to a time-series analysis to find autocorrelation times and estimates of the properties of the animal. Care was taken to avoid initial bias in the simulation by relaxing the animals in runs with 10^6 iterations. The CPU times used for short runs of 10 000 iterations are plotted in figure 3 for uniformly weighted animals in two and in three dimensions (with $\beta_c = \beta_k = 0$ in (2.2)). In two dimensions, $\epsilon \approx 0.74$ and in three dimensions, $\epsilon \approx 0.65$, for uniformly weighted trees [44]. The linear nature of the plots supports (3.11). Linear fits to the data give $C_A \approx 1.7 + 0.18n^{0.74}$ in two dimensions and $C_A \approx 2.7 + 0.38n^{0.65}$ in three dimensions†. These results support our expectations expressed in equation (3.11).

We analyse the performance of the algorithm further by first estimating the probabilities that tree moves will be successful. We then consider the effect these probabilities have on the autocorrelation times of the algorithm. Let $p_n(k|T)$ be the conditional probability that a tree move involving a branch of size k is attempted, *given* that a tree move is attempted. Then by (2.11), $\sum_k kp_n(k|T) \sim n^\epsilon$, and this suggests that $p_n(k|T) \sim k^{\epsilon-2}$. Let $Q_n(k, T)$ be the probability that a tree move on a branch of size k is successful, given that it is proposed. The dependence of $Q_n(k, T)$ on k can be heuristically estimated as follows (using an argument developed in [66]). Suppose that a cut edge was selected and deleted, producing two sub-animals, one of size k and the other of size $(n - k - 1)$. If we assume that cycles can be neglected, then the number of ways that we may attempt to put these together to construct a new animal is given by $2dk(v-k)o_h$, where we follow the argument leading up to equation (3.2). However, we may have $a_k a_{n-k-1}$ different pairs of animals

† In section 4.3 we compute the values of ϵ. The estimates for trees used here is close to the values we will obtain; see equation (4.16). If we are simulating critical percolation clusters, then the values of ϵ used for uniformly weighted trees are inappropriate, and we must use the values obtained for critical percolation clusters.

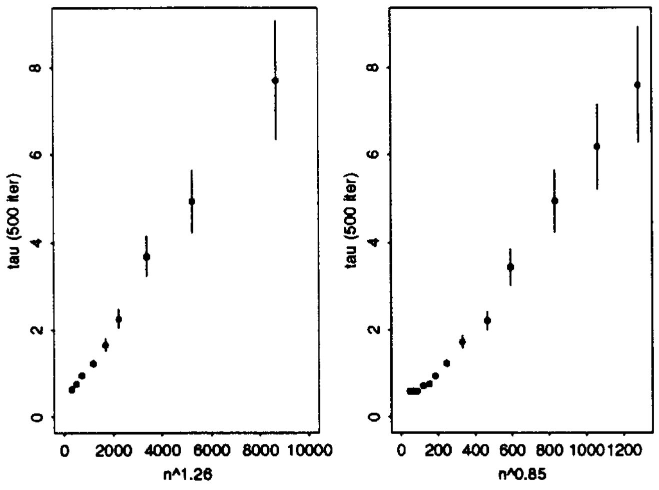

Figure 4. The dependence of τ on n in (left) two dimensions and (right) three dimensions. τ was measured in units of 500 attempted elementary moves.

to join, and at most a_n of these will be successful. Therefore, the probability that we will be successful in our attempt is at most $a_n/(2dk(v-k)o_h a_k a_{n-k-1})$, since not every animal can be made in this way. Now note that $n/2d \leqslant v \leqslant n+1$, and substitute (1.2) for a_n, a_k and a_{n-k-1}:

$$Q_n(k,T) \sim n^{-\theta} k^{\theta-1}(n-k)^{\theta-1}. \qquad (3.10)$$

The key to the performance of the algorithm is the growth of autocorrelation times with n. We can estimate this dependence heuristically as follows: let u and v be two vertices in the animal. The probability that the correlations between u and v are destroyed is approximated by the probability that a tree move is proposed and accepted *with only one of u or v in each sub-animal.* This probability is given by

$$\sum_k p_n(k|T) Q_n(k,T) \frac{2k(n-k)}{n^2} \sim n^{\epsilon+\theta-3} \qquad (3.11)$$

since the probability that u and v are in different sub-animals is $2k(n-k)/n^2$, and after substitution of the expected behaviour of $p_n(k|T)$ and $Q_n(k,T)$. Thus, the correlations are destroyed on a time scale with asymptotic n-dependence given by

$$\tau \sim n^{3-\epsilon-\theta} \quad \text{iterations.} \qquad (3.12)$$

In two dimensions this gives a dependence of $\tau \sim n^{1.26}$ and in three dimensions, $\tau \sim n^{0.85}$, if we use the values for ϵ in [44] and the accepted values for θ. In more than eight dimensions, we predict that τ approaches a constant or diverges logarithmically, since $\epsilon = \frac{1}{2}$ and $\theta = \frac{5}{2}$.

In figure 4 we plot the autocorrelation time associated with the mean-square radius of gyration (see section 4.2) as a function of $n^{3-\epsilon-\theta}$ in two and three dimensions. In two dimensions the data include values of n up to $n = 1500$; the confidence intervals in the measured autocorrelation times for larger values of n are too large (more than 25% of the size of the autocorrelation time) to contribute usefully to the graph. The data in figure 4 support the notion that $\tau \sim n^{1.26}$ in two dimensions and $\tau \sim n^{0.85}$ in three dimensions. The autocorrelation times in figure 4 are measured in terms of units of 500 iterations; for small values of n this unit is too large to measure τ accurately. This explains the data points in figure 4 which corresponds to small n. In CPU time seconds, the autocorrelation time will increase as $\tau \sim n^{3-\theta}$ CPU seconds; in two dimensions this gives $\tau \sim n^2$ CPU seconds and in three dimensions this gives $\tau \sim n^{3/2}$ CPU seconds.

The properties of the animals can also be efficiently computed. The number $n^2 R_n^2$ (R_n^2 is the square radius of gyration) can be updated after every attempted move. The mean branch size can be estimated using the data generated by the breadth-first searches in the algorithm. The mean number of cycles and mean number of contacts, as well as the mean perimeter, the mean number of edges per cycle and the fraction of vertices of various degrees are similarly generated as part of the implementation of the algorithm, and can be efficiently recorded. In other words, most of the CPU time in a run is spent generating animals.

4. Numerical results

In this section we discuss numerical results obtained by simulations of animals with the algorithm described in section 3. The simulations were done for uniformly weighted animals ($\beta_k = \beta_c = 0$), and for animals weighted as critical percolation clusters (the values of the fugacities are then given by equations (2.5) with $p = p_c$). We present data which measures the performance of the algorithm in section 4.1. These data include statistics on the acceptance fraction, the acceptance fractions of various moves, and autocorrelation times of various observables. In section 4.2 we shift the focus by considering instead the properties of animals, weighted uniformly or as critical percolation clusters. Our simulations were done for a variety of sizes (n) of the animals. The value of n was increased until we could not reliably compute the autocorrelation times associated with the properties of the animal for many of the observables. For uniformly weighted animals, we succeeded in simulating animals up to size 5000, but at the critical percolation point we considered animals only up to size 2000.

4.1. Acceptance and rejection of moves in algorithm R

The incidences of various moves in the algorithm and their acceptance fractions are good indicators of the effectiveness of the algorithm. The probability that a certain type of move will be proposed is related to the incidence of certain kinds of edges in the animal. For example, a cycle–cycle or a cycle–leaf move will be proposed with the probability that a uniformly selected edge is in a cycle. Tree moves are proposed when a cut edge is selected.

In figure 5 we plot the incidence of tree moves in the algorithm for simulating uniformly weighted animals in two dimensions. The probability that a tree move of a given size is proposed is estimated by the data presented with $\triangle$'s, and the acceptance fraction is plotted on the same graphs with $+$'s. The crucial observation here is that neither the acceptance fraction of the larger tree moves, nor the proposals of these, are negligible. There is always a (not insignificant) probability that a large tree move will succeed, and this leads to better mixing in the algorithm (and thus to shorter autocorrelation times in observables measured along the Markov chain). Data in three dimensions, and for animals simulated at the percolation point, are similar to those illustrated in figure 5.

As before, we let $p_n(X)$ be the probability that a move of type X is proposed, and define $q_n(X)$ to be the conditional probability that it is accepted (given that it was proposed). Then $p_n(T)$ is the probability that a tree move is proposed (cf equations (3.10) and (3.11)). Similarly, we let X take values CC for a cycle–cycle move, CL for a cycle–leaf move, LL for a leaf–leaf move and LC for a leaf–cycle move. The incidence and the acceptance

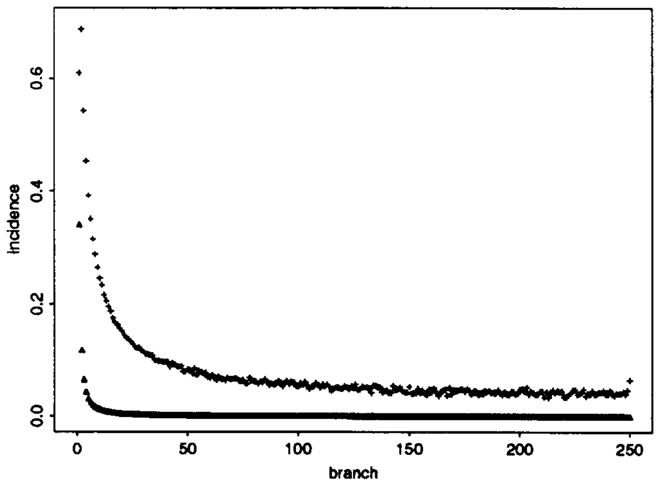

Figure 5. ($\triangle$) The fraction of tree moves of a given size proposed by the algorithm, and (+) the acceptance fraction of these proposed moves. These data were compiled from a run of 5000 000 iterations of uniformly weighted animals of size 500 in two dimensions. The proposals of tree moves involving larger branches, and their acceptance fraction, decay very slowly. This observation explains the fast convergence of algorithm R: there is a reasonable probability that a large move will be proposed and accepted, making a big change in the structure of the animal. We counted cycle moves and leaf moves as having size 1 in this graph. Data at the percolation point, and in three dimensions, are very similar to the data above, and the conclusions are the same.

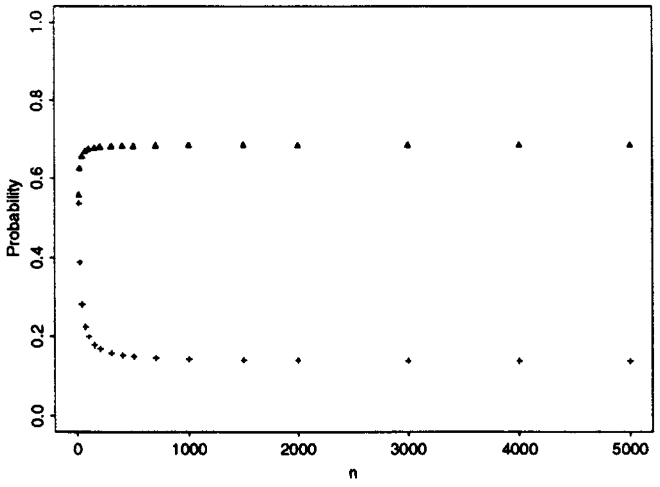

Figure 6. ($\triangle$'s) $p_n(T)$ and (+'s) $q_n(T)$ as a function of n for uniformly weighted animals in two dimensions. Both these quantities converge quickly to the constants in equation (4.1).

fraction of tree moves are plotted in figure 6 in two dimensions. The $\triangle$'s are data points for $p_n(T)$, while $q_n(T)$ is represented by the $+$'s. After some initial change, $p_n(T)$ becomes virtually independent of n. Its limiting value, as n becomes large, can be estimated by fitting the data by assuming that $p_n(T) \approx \zeta(T) + An^{-0.5}$. The acceptance fraction of tree moves remains weakly dependent on n, and we expect it to approach a constant (non-zero)

Table 1. Acceptance fractions.

	$d = 2$		$d = 3$	
Observation	Uniform	Critical	Uniform	Critical
ζ(T)	0.6881	0.2449	0.6632	0.4651
η(T)	0.1254	0.0577	0.2776	0.1630
ζ(CC)	0.0098	0.1803	0.0043	0.0637
η(CC)	1	1	1	1
ζ(CL)	0.0203	0.0692	0.0189	0.0986
η(CL)	1	0.4196	1	0.3818
ζ(LC)	0.0406	0.0599	0.0377	0.0750
η(LC)	0.5	0.5	0.5	0.5
ζ(LL)	0.0829	0.0207	0.1626	0.1120
η(LL)	1	0.6081	1	0.7938

value as $n \to \infty$†. The incidence of the other moves, and their acceptance fractions can similarly be analysed. Suppose that (as above) $p_n(X) \to \zeta(X)$ and $q_n(X) \to \eta(X)$ as n becomes large. For the other moves, our estimates‡ of $\zeta(X)$ and $\eta(X)$ are in table 1 (we ignore data-points with $n < 100$ in compiling this table).

For uniformly weighted animals in the two-dimensional case, all cycle–cycle, cycle–leaf and leaf–leaf moves proposed are accepted, while leaf–cycle moves proposed are accepted with probability 0.5, as specified in algorithm R. We note that the probability of choosing (uniformly) a cut edge in the animal is asymptotically $p(\text{cut edge}) = \zeta(T) = 0.6881$. By counting vertices of degree 1 (see later) we also conclude that the probability of selecting an edge which is a leaf is $p(\text{leaf}) = 0.2563$, so that the probability of selecting an edge in a cycle is $p(\text{cycle}) = 0.0556$. A further 15.83% of the attempts involved the attempted formation of 2-cycles, and were rejected. From these data it is possible to compute other interesting statistics. For example, given that a leaf is chosen, the probability that a move which forms a cycle is executed is 0.1189, etc.

In the case of animals at the critical percolation point in two dimensions we note that the fraction of tree moves is much less than for uniformly weighted animals, but it is still substantial. The acceptance fraction of tree moves is also reduced, but at close to 6% one should still expect it to have a substantial effect on the performance of the algorithm. Note that the acceptance fraction of leaf–cycle moves is 0.50. This is to be expected, since deleting a leaf can break at most $2d - 1$ contacts, and $\beta_c = 2d\beta_k$ (see (2.5)). The more compact nature and the larger incidence of cycles in these weighted animals are reflected by the much larger incidence of cycle–cycle and cycle–leaf moves. Similar to the case above, $p(\text{cut edge}) = 0.2449$, $p(\text{leaf}) = 0.2139$ and $p(\text{cycle}) = 0.5412$. The fraction of attempts rejected due to the formation of 2-cycles is 42.50%.

We repeated this analysis in three dimensions. The results are similarly listed in table 1. In the case of uniformly weighted animals we found that $p(\text{cut edge}) = 0.6632$, $p(\text{leaf}) = 0.3062$ and $p(\text{cycle}) = 0.0306$. The fraction of attempts rejected due to the formation of 2-cycles is 11.33%. At the critical percolation point we found that $p(\text{cut edge}) = 0.4651$, $p(\text{leaf}) = 0.3014$ and $p(\text{cycle}) = 0.2335$. The fraction of attempts rejected due to the formation of 2-cycles is 18.56%. The acceptance fraction of most of the

† This happens because there will be a density of small branches (for example, with two edges) in the limiting animal.

‡ The probabilities of proposed moves do not add up to 1. There are attempted moves which cannot be classified as one of the cases below. These include moves which attempt to create 2-cycles in the animal.

moves (and in particular, of the tree moves) is higher in three dimensions (compared with two dimensions). We expect the algorithm to perform better in three dimensions, and this should be reflected in shorter autocorrelation times for some of the observables that we will compute.

The *acceptance fraction* of algorithm R is the probability that any attempted move will be succesful. This is just the weighted sum of the acceptance fractions of the various moves. Since there will always be a density of leaves in the animal, we expect the acceptance fraction to converge to a constant as $n \to \infty$. Extrapolating to large n, by assuming that $f_n = f + An^{-0.5}$, and by ignoring all data points with $n \leqslant 100$, we estimate that the limiting acceptance fraction for uniformly weighted animals is

$$\begin{aligned} f^{(2)} &= 0.22 \qquad \text{in two dimensions} \\ f^{(3)} &= 0.39 \qquad \text{in three dimensions.} \end{aligned} \tag{4.1}$$

At the critical percolation point, we obtain

$$\begin{aligned} f^{(2)} &= 0.27 \qquad \text{in two dimensions} \\ f^{(3)} &= 0.30 \qquad \text{in three dimensions.} \end{aligned} \tag{4.2}$$

Incidentally, the above can be obtained directly from the values of $\zeta(X)$ and $\eta(X)$ above ($f = \sum_X \zeta(X)\eta(X)$). The acceptance fraction in two dimensions increases from the uniformly weighted animals to animals at the percolation point. In three dimensions the opposite holds.

4.2. Autocorrelation times

Let $\{\omega_i\}$ be the realization of a stationary Markov chain of states ω_i generated by algorithm R, and let $A_i = A(\omega_i)$ be an observable measured along the chain. We collected data along the Markov chain in two ways. In the first instance we computed *block averages* $\bar{A}_j = (\sum_{k=0}^{N-1} A_{Nj+k})/N$, and in the second case we only considered a sub-chain $\tilde{A}_j = A_{Nj}$. N was fixed at 500 for all our runs with $n > 100$. The block-average method is preferable to the sub-chain method, since less information is lost from the original chain. We used it whenever we could 'update' an observable in every successful iteration. Data were collected in the block-average method for the mean-square radius of gyration, the mean branch size, the mean number of contacts and of cycles, the mean perimeter, the mean number of edges per cycle and the fraction of vertices of degree i. The mean span was computed by the second method, updating it only every N iterations.

Let $\mathcal{A}_i$ now represent either $\bar{A}_i$ or $\tilde{A}_i$. If $\mathcal{A}_i$ is a stationary stochastic process, then its mean is

$$\mu = \langle \mathcal{A}_i \rangle = \langle \bar{A}_i \rangle = \langle \tilde{A} \rangle. \tag{4.3}$$

The unnormalized autocorrelation function is

$$C(t) = \langle \mathcal{A}_i \mathcal{A}_{i+t} \rangle - \mu^2 \tag{4.4}$$

and it is normalized as $\rho(t) = C(t)/C(0)$. The *integrated autocorrelation time* τ of the stochastic process $\mathcal{A}_t$ is defined as

$$\tau = \frac{1}{2} \sum_{t=-\infty}^{\infty} \rho(t). \tag{4.5}$$

By analysing our data, we must estimate μ and τ, from a finite sample from the stochastic process. Note that N sets a lower limit on τ: if the chain $\mathcal{A}_i$ is uncorrelated, then $\tau = \frac{1}{2}$ by

Table 2. Dynamical exponents.

	$d = 2$		$d = 3$	
Obs	Uniform $\rho(\chi_7^2)$	Critical $\rho(\chi_5^2)$	Uniform $\rho(\chi_7^2)$	Critical $\rho(\chi_5^2)$
$\langle k_n \rangle$	$0.89 \pm 0.13(7.2)$	$1.58 \pm 0.32(1.9)$	$0.92 \pm 0.10(7.3)$	$1.39 \pm 0.26(3.9)$
$\langle c_n \rangle$	$0.86 \pm 0.10(3.3)$	$0.91 \pm 0.26(2.4)$	$0.663 \pm 0.070(8.4)$	$0.95 \pm 0.17(4.7)$
$\langle R_n^2 \rangle$	$1.28 \pm 0.17(6.3)$	$1.74 \pm 0.70(1.1)$	$0.917 \pm 0.080(2.0)$	$1.48 \pm 0.28(5.2)$
$\langle S_n \rangle$	$1.27 \pm 0.16(8.3)$	$1.61 \pm 0.60(2.2)$	$0.882 \pm 0.084(8.8)$	$1.45 \pm 0.28(2.9)$
$\langle B_n \rangle$	$1.17 \pm 0.13(6.9)$	$1.46 \pm 0.36(2.8)$	$0.723 \pm 0.072(7.6)$	$1.31 \pm 0.22(3.1)$
$\langle C_n \rangle$	$0.175 \pm 0.054(7.3)$	$1.38 \pm 0.34(2.9)$	$0.160 \pm 0.050(3.3)$	$1.25 \pm 0.20(2.9)$
$\langle s_n \rangle$	$0.92 \pm 0.13(5.6)$	$1.18 \pm 0.40(1.9)$	$0.95 \pm 0.10(8.4)$	$1.34 \pm 0.28(3.3)$

(4.5), *in units of N iterations*. The autocorrelation time for the chain A_i could be smaller than $N/2$, but we cannot measure this.

The natural (unbiased) estimator for μ is the sample mean:

$$\bar{\mathcal{A}} = \frac{1}{M} \sum_{i=1}^{M} \mathcal{A}_i \tag{4.6}$$

where we have performed a run of MN iterations. The variance in $\bar{\mathcal{A}}$ is

$$\operatorname{Var} \bar{\mathcal{A}} = \frac{1}{M}(2\tau)C(0) \tag{4.7}$$

provided that $M \gg \tau$ and where we can estimate $C(t)$

$$\hat{C}(t) = \frac{1}{M - |t|} \sum_{i=1}^{M-|t|} (\mathcal{A}_i - \bar{\mathcal{A}})(\mathcal{A}_{i+|t|} - \bar{\mathcal{A}}). \tag{4.8}$$

τ is estimated over a ‘window’ of size w by

$$\hat{\tau} = \sum_{i=-w}^{w} \frac{\hat{C}(i)}{\hat{C}(0)}. \tag{4.9}$$

The variance in τ can be calculated by

$$\operatorname{Var} \tau \approx \frac{2(2w+1)}{M} \tau^2 \tag{4.10}$$

where the window is chosen such $\tau \ll w \ll M$, see [66] for details. In general we have computed $\hat{\tau}$ and $\operatorname{Var} \tau$ over windows of sizes $w = s\hat{\tau}$, for $s = 2, 5, 7, 10, 15, 20, 25, \ldots, 50$. We used a ‘flatness’ criterion to select the best estimates of $\hat{\tau}$; when it becomes insensitive to s, then we can usually assume that the value of w chosen is suitable. If it is not possible to select a value of s in this manner, then the run was deemed unsuitable for analysis.

The estimated autocorrelation times are listed in tables 1–4, and the autocorrelation times for the mean-square radius or gyration, the mean span, the mean perimeter, and the mean number of cycles are displayed in figure 7 for uniformly weighted animals, and for animals at the critical percolation threshold. All the graphs in figure 7 indicate a linear relationship between $\log(\hat{\tau})$ and $\log(n)$, for the larger values of n. Generally, the autocorrelation times for the mean-square radius of gyration, the mean span, and the mean perimeter, are comparable, and fall roughly in a band across the graphs. This indicates that these observables

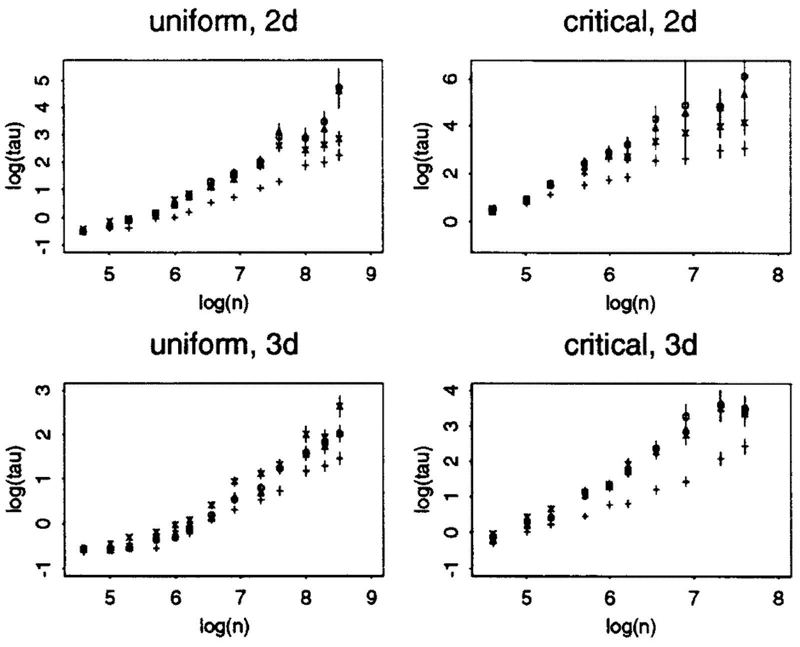

Figure 7. Measured autocorrelation times for the (○) mean-square radius of gyration, (△) mean span, (+) mean number of cycles, (×) mean perimeter. These log–log plots generally show a linear relationship for larger values of n. The first two graphs are for animals in two dimensions, uniformly weighted, and weighted as critical percolation clusters. The bottom graphs are for animals in three dimensions.

are dynamically dependent on the underlying algorithm in a similar way. It seems not unreasonable that the move which most affects these is the tree move. The autocorrelation time of the number of cycles seems to be slightly lower than those of the other, but it is not clear, from the data, that we are measuring a different dynamical property of the algorithm here. Equation (3.14) suggest the existence of a dynamical exponent ρ which characterizes the dynamical behaviour of the algorithm as a function of n: we suspect that

$$\tau \sim n^{\rho} \tag{4.11}$$

where ρ may depend on the type of observable from which τ is estimated. Our data shows that observables such as the acceptance fraction, the mean number of vertices of a given degree and the mean number of edges in cycles, have small autocorrelations, even for large values of n. This indicates a small value of ρ in those cases. On the other hand, we argued in section 3.2 that $\rho = 3 - \epsilon - \theta$, and we should observe this in the autocorrelations of some global observables along the Markov chain.

We analysed the data in figure 7, and data for other observables, by linear-least-squares analysis to obtain best estimates for ρ. The results are in table 2. Each row of table 2 corresponds to a different observable, and for each dimension we stated the nature of the simulation (uniform, or at the critical percolation point), and the χ^2_d statistics of the linear regressions (with d degrees of freedom). Error bars in table 2 are 95% statistical confidence intervals†. To minimize possible systematic errors, we discarded data obtained at smaller

† Here, and in the rest of this manuscript, we will always state 95% confidence intervals on estimates of exponents, unless we explicitly state otherwise. The confidence intervals on raw data (see the appendix) are standard deviations.

Table 3. Fraction of vertices of degree i.

n	$d = 2$		$d = 3$	
Degree	Uniform	Critical	Uniform	Critical
1	0.256 38(9)	0.2139(2)	0.306 18(8)	0.301 41(1)
2	0.491 1(2)	0.4109(2)	0.431 5(2)	0.402 9(2)3
3	0.223 9(2)	0.2979(3)	0.216 9(7)	0.222 95(9)
$\geqslant 4$	0.028 69(5)	0.0774(2)	0.051 450(5)	0.072 72(9)
Mean valence	2.026	2.238	2.074	2.14

Table 4. Best estimates for exponents.

n	$d = 2$		$d = 3$	
Exponent	Uniform	Critical	Uniform	Critical
ρ_l	1.28 ± 0.17	1.68 ± 0.70	0.900 ± 0.084	1.47 ± 0.28
ν	0.6442 ± 0.0017	0.5315 ± 0.0046	0.5016 ± 0.0013	0.4050 ± 0.0024
ϵ	0.74257 ± 0.00085	0.4215 ± 0.0077	0.6587 ± 0.0015	0.4819 ± 0.0050
σ	—	0.406 ± 0.011	—	0.4972 ± 0.0081

values of n: all data obtain at $n < 400$ were ignored in the analysis of uniformly weighted animals. For data measured at the critical percolation point, we ignored all data from values of $n < 300$. There is a variety of values for ρ, depending on the observable associated with it. As expected, the values of ρ obtained from the mean-square radius of gyration and from the mean span for uniformly weighted animals most closely agree with the prediction made in equation (3.12), and in figure 4. Generally speaking, the largest values of ρ are associated with the mean-square radius of gyration data, the mean span data and the mean contact number data. ρ seems to be somewhat smaller for the other observables, indicating that these (possibly) relax on shorter time scales.

4.3. Critical exponents and other results

In this section we present estimates of the critical exponents of animals, obtained by linear least-squares fits to our data which we assume satisfy suitable scaling relations (as expressed in section 2). These scaling relations are subject to corrections, and ignoring these introduce systematic errors in the estimates of exponents, giving instead *effective* exponents. We accept a least-squares fit as good if the χ^2_d statistic is acceptable at the 95% level. Since systematic errors seem to be most significant at the smallest value of n in a good fit, we estimate these errors by repeating our fit without the data points at the smallest value of n. The absolute difference between the computed regression coefficients is taken as the systematic error. In addition to the above, we also attempted to limit our models of scaling relations to be two-parameter models. Thus, using the simplest model, we discarded data points at the smallest value of n in the regression until a good fit was achieved. We then assumed that we had sufficiently controlled for the systematic error due to corrections to scaling (unaccounted for by our model). We changed the model only as a last resort, when all attempts at finding a good fit using a two-parameter model failed. In some cases it was necessary to discard data points which contribute anomalously to the least-square error; we called such a point an *outlier* if discarding it will produce an acceptable fit. Our raw data are in the appendix.

4.3.1. Mean-square radius of gyration. We first analyse the data obtained for uniformly weighted animals in two dimensions. By taking logarithms of equation (2.8), we obtain $\log\langle R_n^2\rangle \approx C + 2\nu\log n$. A fit with the minimum value of n, $n_{\min} = 150$, gives a good fit: $\chi^2_{10} \approx 11.1$, which is acceptable at a level of less than 75%. The fit gives $2\nu_2 = 1.2883 \pm 0.0019$†. We increase $n_{\min}$ to 200 to estimate a systematic error: This gives $2\nu_2 = 1.2869 \pm 0.0026$, and computing the absolute difference gives the following estimate: $\nu_2 = 0.6442 \pm 0.0010 \pm 0.0007$. Similar analyses were carried out in three dimensions and at the critical percolation point. At the critical percolation point we found that the data points at $n = 300$ in two dimensions, and $n = 2000$ in three dimensions, are outliers. Our results are

$$\begin{aligned}
&\nu_2 = 0.6442 \pm 0.0010 \pm 0.0007 && n_{\min} = 150 && \chi^2_{10} \approx 11.1\ (< 75\%)\\
&\nu_2^p = 0.5315 \pm 0.0034 \pm 0.0012 && n_{\min} = 150 && \chi^2_{6} \approx 10.0\ (< 90\%)\\
&\nu_3 = 0.501\,60 \pm 0.000\,61 \pm 0.000\,69 && n_{\min} = 200 && \chi^2_{9} \approx 12.9\ (< 90\%)\\
&\nu_3^p = 0.4050 \pm 0.0020 \pm 0.0004 && n_{\min} = 150 && \chi^2_{6} \approx 6.4\ (< 75\%).
\end{aligned} \tag{4.12}$$

4.3.2. Mean span. The mean-span data seem to contain strong corrections to scaling, and we could only obtain acceptable fits with large values of $n_{\min}$ if we assume the two parameter model $\log\langle S_n\rangle C + \sim \nu\log n$‡. Our best estimates are

$$\begin{aligned}
&\nu_2 = 0.6481 \pm 0.0016 \pm 0.0008 && n_{\min} = 400 && \chi^2_{7} \approx 8.6\ (< 75\%)\\
&\nu_2^p = 0.5429 \pm 0.0024 \pm 0.0018 && n_{\min} = 300 && \chi^2_{5} \approx 8.6\ (< 90\%)\\
&\nu_3 = 0.5100 \pm 0.0011 \pm 0.0013 && n_{\min} = 700 && \chi^2_{5} \approx 8.1\ (< 90\%)\\
&\nu_3^p = 0.4255 \pm 0.0021 \pm 0.0006 && n_{\min} = 500 && \chi^2_{3} \approx 4.2\ (< 90\%).
\end{aligned} \tag{4.13}$$

The exponent ν is systematically larger when computed from span data, compared with its values when computed from the mean-square radius of gyration data. In addition, we obtained a better fit to our model from the mean-square radius of gyration data in each of the cases above. We therefore accept the values of ν estimated from the mean-square radius of gyration data as more reliable.

† We will add a subscript to all exponents to indicate dimension. In addition, a superscript p will indicate that an exponent was computed for animals at the critical percolation point. For example, ν_2^p will be the metric exponent for animals at the critical percolation point in two dimensions. The format will always be *best estimate* ± 95% *statistical confidence interval* ± *estimated systematic error*.

‡ An assumption that $\log\langle S_n\rangle \sim \nu\log n + An^{-\Delta}$, where Δ is an (effective) correction to the scaling exponent, proved successful in reducing the size of the least-square error, but ν turned out to be sensitive to the value of Δ. Thus, it seemed prudent to stay with a two-parameter fit, even if we had to discard a lot of data for an acceptable fit.

4.3.3. Mean branch size. An assumption that $\log\langle B_n\rangle = C + \epsilon \log n$ for some constant A proved suitable for good fits to our data. Our best estimates are

$$\begin{aligned}
&\epsilon_2 = 0.742\,57 \pm 0.000\,62 \pm 0.000\,23 \qquad n_{\min} = 70 \qquad \chi^2_{12} \approx 17.7\ (< 90\%)\\
&\epsilon^p_2 = 0.4215 \pm 0.0042 \pm 0.0035 \qquad n_{\min} = 70 \qquad \chi^2_9 \approx 13.8\ (< 90\%)\\
&\epsilon_3 = 0.658\,74 \pm 0.000\,80 \pm 0.000\,69 \qquad n_{\min} = 200 \qquad \chi^2_9 \approx 8.1\ (< 75\%)\\
&\epsilon^p_3 = 0.4819 \pm 0.0034 \pm 0.0016 \qquad n_{\min} = 150 \qquad \chi^2_7 \approx 12.6\ (< 95\%).
\end{aligned} \tag{4.14}$$

4.3.4. Mean number of contacts. The mean number of contacts per vertex is expected to converge to a constant as n increases. Examining (2.12), and our data, we observe that $K = 0.315\,55 \pm 0.000\,38$ for expanded animals in two dimensions†. If we assume this as a good approximation of K, then we can estimate the rate at which $\langle k_n\rangle/n$ converges, assuming that it is a power law: $\langle k_n\rangle \approx 0.315\,55n + An^{1-x}$. We therefore tried to fit $\log(0.315\,55 - \langle k_n\rangle/n)$ against $\log n$. There may be other corrections to this, apart from the term An^{-x}; we assume that these will be dominated by the power-law correction for all $n \geqslant n_{\min}$ (and where $n_{\min}$ is some cut-off). If n is large, then the correction will be dominated by the statistical errors in our simulations, so we will also take $n \leqslant n_{\max}$ in our analysis. A (weighted) least-squares fit with $n_{\min} = 100$ and $n_{\max} = 1000$ gives a fit with a large least squares error, and with $x \approx 1.070$. Additional fits with other values for $n_{\min}$ and $n_{\max}$ indicate that x is not very sensitive to the values of $n_{\min}$ and $n_{\max}$. Thus, while the uncertainty in x is large, our estimate, that it is close to 1, indicates that the corrections to $\langle k_n\rangle/n$ go to zero fast with increasing n. If we repeat this analysis in the other cases, we obtain

$$\begin{aligned}
&K_2 = 0.315\,55 \pm 0.000\,38 \qquad x \approx 1\\
&K^p_2 = 0.5397 \pm 0.0017 \qquad x \approx 0.9\\
&K_3 = 0.3660 \pm 0.0052 \qquad x \approx 1\\
&K^p_3 = 0.6972 \pm 0.0024 \qquad x \approx 0.85.
\end{aligned} \tag{4.15}$$

4.3.5. Mean number of cycles. The mean number of cycles per vertex is also expected to converge to a constant with increasing n (see equation (2.13)). We analysed our data in the same manner as for the mean number of contacts. We were able to estimate the limit of $\langle c_n\rangle/n$ by plotting it against n. For large n this is converged to about three or four digits. We obtain

$$\begin{aligned}
&C_2 = 0.012\,47 \pm 0.000\,07\\
&C^p_2 = 0.107\,05 \pm 0.000\,40\\
&C_3 = 0.006\,515 \pm 0.000\,029\\
&C^p_3 = 0.037\,60 \pm 0.000\,19.
\end{aligned} \tag{4.16}$$

4.3.6. Mean perimeter. For expanded animals, the arguments preceding and following (2.14) indicate that $\langle s_n\rangle$ should scale proportionally to n. We test this expectation by

† In this case, we must estimate the limit of $\langle k_n\rangle/n$ as $n \to \infty$. By plotting $\langle k_n\rangle/n$ against n, we observe that the ratio is constant to about three or four digits, in which case we assume that the our data are converged to about three or four digits. We can thus just take the results at the largest value of n as our best estimate for the limit. Implicitly we are assuming that the convergent corrections to the limit all goes to zero quite quickly with n, and that there are no slow corrections which take longer to settle down.

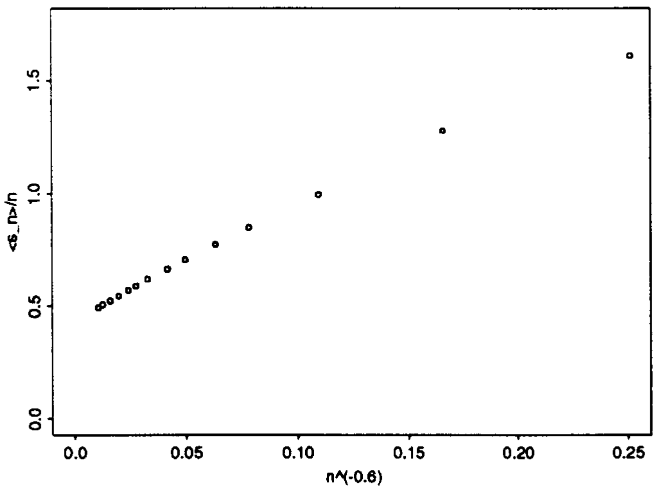

Figure 8. A plot of $\langle s_n\rangle/n$ against $n^{-0.6}$ for critical percolation clusters in two dimensions. This plot can be interpreted as good support for (2.17).

assuming (2.14) and by doing least-squares fits to our data. Our best estimates for A are

$$\begin{aligned} &A_2 = 1.318\,24 \pm 0.000\,34 \qquad n_{\min} = 70 \qquad \chi^2_{12} = 6.2\ (< 10\%) \\ &A_3 = 3.227\,58 \pm 0.000\,34 \qquad n_{\min} = 100 \qquad \chi^2_{11} = 15.1\ (< 90\%). \end{aligned} \tag{4.17}$$

At the critical percolation point, we expect instead the behaviour in (2.15), where there is a surface correction to the linear behaviour of the mean perimeter. Simulation of percolation clusters indicate that $\sigma \approx 0.40$ in both two and three dimensions (see [28]), so we can test (2.15) by plotting $\langle s_n\rangle/n$ against $n^{-0.6}$. In figure 8 we display the plot for our data in two dimensions, and we interpret this again as strong support for (2.15) if $\sigma \approx 0.4$.

We can use the accepted values of p_c to estimate the exponent σ in (2.15). Since the excess perimeter t'_n at the critical percolation point is expected to scale as n^σ, we assume that

$$\frac{\langle s_n + k_n\rangle}{n} = \frac{1-p_c}{p_c} + Cn^{\sigma-1} \tag{4.18}$$

where C is a constant, and where we argue as in the fourth footnote of section 2. A weighted linear least-squares analysis of our data gives

$$\begin{aligned} &\sigma_2 = 0.4061 \pm 0.0063 \pm 0.0048 \qquad n_{\min} = 40 \qquad \chi^2_{10} = 3.3\ (< 10\%) \\ &\sigma_3 = 0.4972 \pm 0.0029 \pm 0.0052 \qquad n_{\min} = 20 \qquad \chi^2_{11} = 10.6\ (< 75\%). \end{aligned} \tag{4.19}$$

The result in two dimensions is close to the expected value of 0.4, but the exponent is slightly larger in three dimensions (we used the values $p_c = 0.5$ in two dimensions and $p_c = 0.249$ in three dimensions in equation (4.18)).

We can now estimate the value of A in equation (2.15), assuming the values of σ estimated above. An assumption that $\langle s_n\rangle/n = A + Bn^{\sigma-1}$ gives acceptable fits:

$$\begin{aligned} &A_2^p \approx 0.4408 \pm 0.0043 \qquad n_{\min} = 150 \qquad \chi^2_7 = 4.7\ (< 50\%) \\ &A_3^p \approx 2.2640 \pm 0.0017 \qquad n_{\min} = 150 \qquad \chi^2_7 = 12.5\ (< 95\%) \end{aligned} \tag{4.20}$$

where the point at $n = 500$ was an outlier in three dimensions. If we add A and K (which we obtained for the mean contact number, and stated in equation (2.15)), then by the argument following (2.15) we should be able to estimate p_c†. In two dimensions we found that $A + K = 0.9809 \pm 0.0045$ which implies that $p_c = 0.5048 \pm 0.0011$, and in three dimensions, $A + K = 2.9953 \pm 0.0077$ which gives $p_c = 0.250\,29 \pm 0.000\,49$. These estimates for the critical percolation probability are very close to their accepted values of $\frac{1}{2}$ in two dimensions and 0.249 in three dimensions [28].

4.3.7. Mean number of edges in a cycle. The mean numbers of edges in a cycle is shown in table A.7. For expanded animals we can say with some certainty that $\langle C_n \rangle \to 0.3835 \pm 0.0022$ in two dimensions, and $\langle C_n \rangle \to 0.2534 \pm 0.0015$ in three dimensions. At the critical percolation point, the convergence of the mean number of edges in a cycle is slow, and we cannot plot it against n to obtain a limit. It seems possible that $\langle C_n \rangle \to \infty$ with n, and we tried to explore this by assuming that $\langle C_n \rangle = \omega \log n$. In two dimensions a least squares analysis gives $\omega = 0.9261 \pm 0.0052$ with $\chi_8^2 \approx 13.2$ ($< 90\%$) and in three dimensions, $\omega = 0.5190 \pm 0.0035$ with $\chi_7^2 \approx 10.1$ ($< 90\%$). We do not take this as strong evidence that our assumed model gives the correct n-dependence for $\langle C_n \rangle$, but we do accept this as evidence that the mean number of edges in a cycle diverges with n.

4.3.8. The fraction of vertices of degree i. In all our simulations the fraction of vertices of degree i have converged to four or five digits for $n > 1000$. We therefore state the results at the largest values of n as our best estimates of these fractions for infinite animals. Our results are listed in table 3. The mean valence of sites can be computed from the data in table 3 as we explained in section 2. These are also listed in table 3.

5. Discussion of results

In this paper we have proposed and tested a new metropolis Monte Carlo algorithm for the simulation of lattice animals. We implemented the algorithm and tested it by measuring some the observables associated with lattice animals. In particular, our data allows us to estimate the exponents ρ_l, ν and ϵ.

The dynamical exponents associated with various observables seems to adopt many different values, as is apparent in table 1. Despite this, it seems that the exponents associated with the mean span and mean-square radius of gyration are the largest, and are close in value to each other in each of the cases we considered. Assuming that they do indeed measure the same underlying dynamics, we can take the average to give our best estimate of the dynamical exponent associated with the length scale of the animals. We call this exponent ρ_l and we record our best values in table 4. For uniform lattice trees, the same exponent was measured for the non-local algorithm in [44], giving $\rho_l^2(\text{Trees}) = 1.45 \pm 0.04$ and $\rho_l^3(\text{Trees}) = 1.12 \pm 0.03$ (the superscript indicates the number of dimensions). In the case of animals we obtain here $\rho_l^2(\text{Animal}) = 1.28 \pm 0.17$ and $\rho_l^3(\text{Animal}) = 0.900 \pm 0.084$. The two-dimensional results are not inconsistent with the results for the tree algorithm, but we do measure a smaller autocorrelation time in the case of three-dimensional animals. Equation (3.12) suggests that $\rho_l = 3 - \epsilon - \theta$; if we substitute our best estimates for ϵ and the accepted values for θ we do indeed find a value close to the measured value for expanded animals.

† This observation is circular: we need to know p_c in order to estimate A and K in the first place. At best this will give a self-consistent check on the algorithm and our (initial) value for p_c. We do not hope to compute an improved value for the critical percolation probability in this manner.

The exponent ν seems to be best estimated by the mean-square radius of gyration data. There seemed to be significant corrections to scaling in the mean span, and we had to discard many data points to obtain satisfactory fits in our analysis. Our best estimates for ν are stated in table 4, where we add the statistical and systematic errors to state a single error bar. For expanded animals, the values of ν in table 4 compares very well with estimates for lattice trees (see [44] and references therein). Indeed, our error bars on ν are indeed smaller than those obtained for trees (see [44, table 9]) in a large number of studies, no doubt due to the fact that we simulated animals of size upto 5000. ν has also been estimated for lattice animals in two dimensions in [63]. They obtain 0.628 ± 0.023, which barely excludes our estimate. For animals at the critical percolation point the accepted estimated of ν is approximately 0.53 in two dimensions and 0.40 in three dimensions†. Our results are consistent with these estimates. Along the θ-line we expect ν to assume the value of trees in a θ-solvent; this is estimated to be 0.54 ± 0.03 in two dimensions and 0.400 ± 0.005 in three dimensions [45], also close to the value of ν at the percolation point. There seems to be no reason to believe that these values should be different at the percolation point, or along the θ' line in figure 2 (see for example [55]).

Best estimates for exponent ϵ are also listed in table 4. For expanded animals, our results are within the error bars of the branch exponent of lattice trees [44] in both two and three dimensions, but the error bars in this paper are smaller, improving the previous estimates for this exponent. At the critical percolation point we have newly computed values for ϵ.

We were able to compute the crossover exponent σ from the excess perimeter data, assuming that $p_c = 0.5$ in two dimensions and $p_c = 0.249$ in three dimensions. Our result in two dimensions is consistent with the estimate of 0.4 for σ in two dimensions [28], but in three dimensions we found a slightly larger value. As a check on the consistency of our *a priori* assumptions that $p_c = 0.249$ in three dimensions, we estimated the critical edge probabilities for percolation from our data. The results were self-consistent, and supported our belief that we simulated critical percolation clusters.

The fraction of vertices of degree i were computed for bond percolation in the square lattice in [30] at $p = p_c$ (see [30, table 1]). They found that $\delta_1 = 0.215$, $\delta_2 = 0.410$, $\delta_3 = 0.297$ and $\delta_4 = 0.078$, in each case very close to our estimates in table 3. The mean valence of a lattice tree is 2, and our computed mean valences in table 3 are in every case close to 2. In fact, it is known that the mean valence of animals and critical percolation clusters is equal to 2 in the scaling limit [30].

Acknowledgments

EJJvR and NM are supported by operating and equipment grants from NSERC (Canada). We also acknowledge many fruitful discussions with MC Tesi, E Orlandini and SG Whittington.

† These estimates can be made from the relation $R \sim n^{(1+1/\delta)/d}$ for the root-mean-square radius of gyration R of a percolation cluster at $p = p_c$. δ is a critical exponent (in the classical description associated with the scaling of the magnetization with the external field at the critical point of a ferromagnet), and d is the spatial dimension. It is thought that $\delta = 18.00 \pm 0.75$ [68] in two dimensions, and $\delta = 5.0 \pm 0.8$ [69] in three dimensions (see also [28]). We can use our best values for ν at the critical percolation point to compute estimates for δ. We obtain $\delta = 15.9 \pm 2.4$ in two dimensions, and $\delta = 4.66 \pm 0.16$ in three dimensions (the error bars include a 95% confidence interval as well as an estimated systematic error).

Appendix. Raw data

The following tables show the raw data from our Monte Carlo simulations. All confidence intervals are standard deviations, stated as two digits in parentheses indicating the uncertainty in the last two digits of the measured quantities.

Table A1. Mean-square radius of gyration.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	1.866 72(81)	1.4264(11)	1.326 96(52)	1.204 21(62)
20	5.079 8(32)	3.4359(38)	3.006 9(12)	2.514 2(14)
40	13.005 2(89)	7.7088(84)	6.348 6(22)	4.807 9(26)
70	27.305(17)	14.441(18)	11.339 9(52)	7.862 1(62)
100	43.428(34)	21.318(37)	16.328 3(88)	10.635(12)
150	73.662(75)	33.188(73)	24.632(16)	14.899(24)
200	106.97(14)	45.12(14)	32.918(25)	18.806(34)
300	180.11(22)	70.79(35)	49.579(47)	26.213(73)
400	261.35(52)	94.23(57)	66.079(70)	32.91(11)
500	348.32(88)	120.68(89)	82.71(11)	39.77(17)
700	537.6(17)	172.7(21)	115.97(18)	51.61(28)
1000	846.9(32)	249.5(42)	166.03(34)	69.48(58)
1500	1436.2(7)	369.0(56)	248.07(58)	95.21(95)
2000	2074(16)	526(14)	331.5(11)	110.8(12)
3000	3439(24)	—	496.3(19)	—
4000	5087(51)	—	667.2(29)	—
5000	6680(130)	—	827.9(39)	—

Table A2. Mean span.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	3.338 19(57)	2.9104(12)	2.230 01(34)	2.116 52(52)
20	5.730 1(14)	4.7778(24)	3.630 24(52)	3.347 35(77)
40	9.425 5(24)	7.4758(35)	5.575 13(70)	4.965 5(11)
70	13.862 9(27)	10.5019(49)	7.694 2(13)	6.630 8(19)
100	17.618 6(49)	12.9390(79)	9.379 5(19)	7.898 5(30)
150	23.097 3(86)	16.438(13)	11.691 8(26)	9.566 7(51)
200	27.943(13)	19.225(23)	13.637 0(36)	10.909 4(69)
300	36.420(16)	24.239(42)	16.901 1(56)	13.098(12)
400	43.968(32)	28.197(61)	19.640 9(74)	14.855(16)
500	50.854(44)	32.016(70)	22.051 8(94)	16.417(22)
700	63.295(67)	38.51(15)	26.257(14)	18.932(34)
1000	79.68(11)	46.71(28)	31.562(22)	22.145(52)
1500	103.88(18)	57.46(35)	38.741(29)	26.212(90)
2000	124.89(38)	68.07(47)	44.925(47)	29.531(93)
3000	161.50(44)	—	55.188(68)	—
4000	195.69(63)	—	63.890(87)	—
5000	224.9(15)	—	71.49(13)	—

Table A3. Mean branch size.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	1.7240(11)	1.2282(22)	1.658 44(96)	1.5191(13)
20	3.0241(21)	1.7947(45)	2.820 1(12)	2.3956(20)
40	5.1240(31)	2.4927(59)	4.576 2(15)	3.5575(24)
70	7.7927(36)	3.2165(72)	6.678 4(27)	4.7948(43)
100	10.1543(52)	3.748(11)	8.469 4(38)	5.7663(65)
150	13.7372(88)	4.463(14)	11.088 3(56)	7.057(12)
200	17.001(13)	5.021(21)	13.424 1(77)	8.116(14)
300	22.990(14)	6.026(32)	17.554(12)	9.889(24)
400	28.419(31)	6.706(48)	21.231(17)	11.327(31)
500	33.587(44)	7.358(61)	24.571(21)	12.717(46)
700	43.115(65)	8.45(11)	30.683(32)	14.793(67)
1000	56.10(13)	9.76(15)	38.848(48)	17.73(11)
1500	75.94(20)	11.32(22)	50.679(76)	21.11(18)
2000	94.03(28)	12.85(28)	61.13(11)	24.24(25)
3000	125.58(42)	—	79.75(17)	—
4000	156.97(81)	—	96.58(23)	—
5000	184.1(13)	—	111.69(34)	—

Table A4. Mean number of contacts.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	1.6234(13)	1.9704(23)	1.4890(13)	1.7942(17)
20	4.5653(40)	5.7366(69)	4.6276(27)	5.9440(40)
40	10.7779(77)	14.443(12)	11.5819(45)	16.1468(74)
70	20.1692(96)	28.634(14)	22.3902(97)	33.353(19)
100	29.633(14)	43.536(25)	33.298(14)	51.607(39)
150	45.407(26)	68.873(42)	51.606(24)	83.319(73)
200	61.184(36)	94.798(61)	69.875(34)	116.16(11)
300	92.900(54)	147.26(11)	106.570(52)	183.24(20)
400	124.435(82)	200.99(17)	143.211(71)	251.53(27)
500	155.98(11)	254.56(23)	179.872(90)	320.10(43)
700	219.26(14)	363.16(38)	253.30(13)	460.86(67)
1000	313.77(20)	527.89(78)	363.02(21)	672.4(11)
1500	471.88(34)	803.5(12)	547.07(30)	1033.3(19)
2000	629.41(55)	1079.3(17)	730.33(40)	1394.4(24)
3000	946.62(63)	—	1095.49(68)	—
4000	1262.05(85)	—	1464.22(78)	—
5000	1577.73(95)	—	1830.0(13)	—

Table A5. Mean number of cycles.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	0.069 45(35)	0.4909(20)	0.031 91(19)	0.142 39(84)
20	0.194 14(93)	1.3799(52)	0.091 37(37)	0.446 4(15)
40	0.445 7(15)	3.3138(87)	0.227 52(55)	1.119 5(23)
70	0.817 8(19)	6.334(12)	0.424 97(95)	2.184 2(43)
100	1.195 0(27)	9.416(19)	0.622 0(14)	3.267 6(63)
150	1.826 9(41)	14.653(28)	0.949 3(22)	5.130(11)
200	2.450 3(52)	19.957(42)	1.277 2(30)	6.988(14)
300	3.689 6(82)	30.485(65)	1.922 1(41)	10.773(23)
400	4.947(11)	41.164(86)	2.577 6(62)	14.546(32)
500	6.169(14)	51.98(11)	3.249 0(72)	18.233(38)
700	8.716(21)	73.445(18)	4.555(11)	25.880(57)
1000	12.415(28)	105.66(23)	6.491(16)	37.262(78)
1500	18.611(43)	159.08(33)	9.783(23)	56.50(14)
2000	24.925(57)	214.10(40)	13.042(29)	75.19(19)
3000	37.373(94)	—	19.498(47)	—
4000	49.72(11)	—	26.054(59)	—
5000	62.35(16)	—	32.574(73)	—

Table A6. Mean perimeter.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	18.4754(28)	16.0957(55)	38.8306(28)	37.5573(55)
20	32.0929(91)	25.607(14)	72.1680(59)	67.434(12)
40	58.661(19)	39.859(27)	137.471(10)	122.989(22)
70	98.391(22)	59.398(44)	234.670(22)	202.190(50)
100	137.954(35)	77.265(74)	331.672(32)	279.181(93)
150	203.879(59)	105.64(13)	493.092(52)	404.58(19)
200	269.830(79)	132.58(22)	654.586(74)	527.76(27)
300	401.44(13)	185.53(37)	977.33(11)	770.89(47)
400	533.34(18)	235.36(62)	1 300.11(15)	1011.67(61)
500	665.36(23)	284.97(71)	1 622.76(20)	1252.4(10)
700	928.62(31)	381.9(12)	2 268.08(28)	1725.0(16)
1000	1324.81(45)	523.6(18)	3 237.02(47)	2433.6(26)
1500	1983.81(72)	758.7(26)	4 849.16(65)	3596.5(46)
2000	2643.5(13)	986.9(34)	6 463.10(86)	4768.0(52)
3000	3959.3(15)	—	9 694.0(15)	—
4000	5279.0(18)	—	12 917.3(17)	—
5000	6597.2(23)	—	16 146.6(27)	—

Table A7. Mean number of edges per cycle.

	$d = 2$		$d = 3$	
n	Uniform	Critical	Uniform	Critical
10	0.140 39(7)	0.9917(37)	0.065 55(37)	0.2965(17)
20	0.218 03(10)	1.5336(49)	0.113 44(37)	0.5417(18)
40	0.276 25(10)	2.1005(45)	0.154 71(34)	0.8161(15)
70	0.309 13(68)	2.5733(39)	0.181 78(39)	1.0588(20)
100	0.325 55(72)	2.8957(49)	0.195 71(42)	1.2199(24)
150	0.341 93(78)	3.2599(59)	0.209 30(45)	1.4210(32)
200	0.349 73(79)	3.5278(74)	0.216 72(47)	1.5679(39)
300	0.358 93(85)	3.880(11)	0.226 15(53)	1.7772(52)
400	0.365 28(92)	4.182(12)	0.232 08(56)	1.9220(57)
500	0.368 4(11)	4.383(17)	0.236 73(56)	2.0371(74)
700	0.372 8(10)	4.692(21)	0.241 15(59)	2.2220(99)
1000	0.375 7(11)	5.051(31)	0.243 29(59)	2.392(12)
1500	0.377 6(12)	5.463(39)	0.247 83(62)	2.654(18)
2000	0.381 3(13)	5.695(45)	0.249 70(65)	2.792(22)
3000	0.383 1(11)	—	0.250 49(69)	—
4000	0.383 2(11)	—	0.252 46(70)	—
5000	0.383 5(11)	—	0.253 41(71)	—

References

[1] Harary F 1960 *MagyarTud. Akad. Mat. Kutató Intezetenek Kozlemenyei* **5** 63
[2] Lubensky T C and Isaacson J 1979 *Phys. Rev.* **20** 2130
[3] Gaunt D S, Sykes M F and Ruskin H 1976 *J. Phys. A: Math. Gen.* **9** 1899
[4] Domb C 1976 *J. Phys. A: Math. Gen.* **9** L141
[5] Stauffer D 1976 *J. Stat. Phys.* **18** 125
[6] Lubensky T C and Isaacson J 1978 *Phys. Rev. Lett.* **41** 829
[7] Klein D J 1981 *J. Chem. Phys.* **75** 5186
[8] Whittington S G, Torrie G M and Gaunt D S 1983 *J. Phys. A: Math. Gen.* **16** 1695
[9] Zimm B H and Stockmayer W H 1949 *J. Chem. Phys.* **17** 1301
[10] Dobson G R and Gordon M 1964 *J. Chem. Phys.* **41** 2389
[11] Fisher M E and Essam J W 1961 *J. Math. Phys.* **2** 609
[12] Hara T and Slade G 1991 *J. Stat. Phys.* **59** 1469
[13] Flesia S, Gaunt D S, Soteros C E and Whittington S G 1992 *J. Phys. A: Math. Gen.* **25** L1169
[14] Flesia S, Gaunt D S, Soteros C E and Whittington S G 1993 *J. Phys. A: Math. Gen.* **26** L993
[15] Flesia S, Gaunt D S, Soteros C E and Whittington S G 1994 *J. Phys. A: Math. Gen.* **27** 5831
[16] Parisi G and Sourlas N 1981 *Phys. Rev. Lett.* **46** 871
[17] Derrida B and Herrmann H J 1983 *J. Physique* **44** 1365
[18] Dickman R and Shieve W C 1984 *J. Physique* **45** 1727
[19] Lam P M 1988 *Phys. Rev.* B **38** 2813
[20] Chang I S and Shapir Y 1988 *Phys. Rev.* **38** 6736
[21] Gaunt D S and Flesia S 1990 *Physica* **168A** 602
[22] Dhar D 1987 *J. Phys. A: Math. Gen.* **20** L847
[23] Whittington S G, Soteros C E and Madras N 1991 *J. Math. Chem.* **7** 87
[24] Madras N, Soteros C E and Whittington S G 1988 *J. Phys. A: Math. Gen.* **21** 4617
[25] Behringer R E 1958 *J. Chem. Phys.* **29** 537
[26] de Gennes P G, Lafore P and Millot J 1959 *J. Chem. Phys.* **11** 105
[27] Domb C 1959 *Nature* **184** 509
[28] Stauffer D 1979 *Phys. Rep.* **54** 1
[29] Gaunt D S, Middlemiss K M and Whittington S G 1980 *J. Phys. A: Math. Gen.* **13** 3029
[30] Whittington S G, Middlemiss K M and Gaunt D S 1981 *J. Phys. A: Math. Gen.* **14** 2415
[31] Peters H P, Stauffer D, Hölters H P and Loewenich K 1979 *Z. Phys.* B **34** 399

[32] Gaunt D S, Sykes M F, Torrie G M and Whittington S G 1982 *J. Phys. A: Math. Gen.* **15** 3209
[33] Margolina A, Djordjevic Z V, Stauffer D and Stanley H E 1983 *Phys. Rev.* B **28** 1652
[34] Margolina A, Family F and Privman V 1984 *Z. Phys.* B **54** 321
[35] Flesia S, Gaunt D S, Soteros C E and Whittington S G 1992 *J. Phys. A: Math. Gen.* **25** 3515
[36] de Queiroz S L A 1995 *J. Phys. A: Math. Gen.* **28** 6315
[37] Flesia S and Gaunt D S 1992 *J. Phys. A: Math. Gen.* **25** 2127
[38] Leath P L 1976 *Phys. Rev. Lett.* **36** 921
[39] Leath P L 1976 *Phys. Rev.* B **14** 5046
[40] Leath P L and Reich G R 1978 *J. Phys. C: Solid State Phys.* **11** 4017
[41] Stauffer D 1978 *Phys. Rev. Lett.* **41** 1333
[42] Herrmann H J 1979 *Z. Phys.* B **32** 335
[43] Stratychuk L M and Soteros C E 1996 *J. Phys. A: Math. Gen.* **29** 7067
[44] Janse van Rensburg E J and Madras N 1992 *J. Phys. A: Math. Gen.* **25** 303
[45] Madras N and Janse van Rensburg E J 1997 *J. Stat. Phys.* **86** 1
[46] Redner S 1979 *J. Phys. A: Math. Gen.* **12** L239
[47] Seitz W A and Klein D J 1981 *J. Chem. Phys.* **75** 5190
[48] Meirovitch H 1987 *J. Phys. A: Math. Gen.* **20** 6059
[49] Grimmett G 1989 *Percolation* (New York: Springer)
[50] Klarner D A 1967 *Can. J. Math.* **19** 851
[51] Whittington S G and Gaunt D S 1978 *J. Phys. A: Math. Gen.* **11** 1449
[52] Hille E 1948 *Functional Analysis and Semi-Groups (AMS Colloq. Publ. 31)* (New York: American Mathematical Society)
[53] Hardy G H, Littlewood J E and Pólya G 1952 *Inequalities* 2nd edn (Cambridge: Cambridge University Press)
Rockafellar R T 1970 *Convex Analysis* (Princeton, NJ: Princeton University Press)
[54] Janse van Rensburg E J and Madras N 1996 *Numerical Methods for Polymeric Systems, Proc. 1995/96 IMA program on Mathematical Methods in Material Science Workshop 7 (May, 1996)* ed S G Whittington
[55] Seno F and Vanderzande C 1994 *J. Phys. A: Math. Gen.* **27** 5813
Seno F and Vanderzande C 1994 *J. Phys. A: Math. Gen.* **27** 7937
[56] Vanderzande C 1996 Private communication
[57] Sykes M F and Glen M 1976 *J. Phys. A: Math. Gen.* **9** 87
[58] Sykes M F, Gaunt D S and Glen M 1976 *J. Phys. A: Math. Gen.* **9** 715
[59] Rands B M I and Welsh D J A 1981 *IMA J. Appl. Math.* **27** 1
[60] Vyssotsky V A, Gordon S B, Frisch H L and Hammersley J M 1961 *Phys. Rev.* **123** 1566
[61] Dean P and Bird N F 1967 *Proc. Camb. Phil. Soc.* **63** 477
[62] Stoll E and Domb C 1978 *J. Phys. A: Math. Gen.* **11** L57
[63] Soteros C E and Paulhus M M 1996 *IMA Preprint*
[64] Hammersley J M and Handscomb D C 1964 *Monte Carlo Methods* (London: Chapman and Hall)
[65] Knuth D E 1973 *The Art of Computer Programming* vol 3 (Reading, MA: Addison-Wesley)
[66] Madras N and Sokal A 1988 *J. Stat. Phys.* **50** 109
[67] Wilson R J and Watkins J J 1990 *Graphs: An Introductory Approach* (New York: Wiley)
[68] Gaunt D S and Sykes M F 1976 *J. Phys. A: Math. Gen.* **9** 1109
[69] Gaunt D S 1977 *J. Phys. A: Math. Gen.* **10** 807