Determination of the bond percolation threshold for the Kagomé lattice

1997 J. Phys. A: Math. Gen. 30 5351

(http://iopscience.iop.org/0305-4470/30/15/021)

J. Phys. A: Math. Gen. **30** (1997) 5351–5359. Printed in the UK PII: S0305-4470(97)82958-7

Determination of the bond percolation threshold for the Kagomé lattice

Robert M Ziff† and Paul N Suding‡

Department of Chemical Engineering, University of Michigan, Ann Arbor, MI 48109-2136, USA

Received 1 April 1997

Abstract. The hull-gradient method is used to determine the critical threshold for bond percolation on the two-dimensional Kagomé lattice (and its dual, the dice lattice). For this system, the hull walk is represented as a self-avoiding trail, or mirror-model trajectory, on the (3,4,6,4)-Archimedean tiling lattice. The result $p_c = 0.524\,4053 \pm 0.000\,0003$ (one standard deviation of error) is not consistent with previously conjectured values.

1. Introduction

The Kagomé lattice (figure 1) is one of the fundamental lattices of two-dimensional percolation, as well as many other two-dimensional lattice problems. It is one of the 11 Archimedean tiling lattices (in which all vertices are of the same type), designated as (3,6,3,6) in the notation of [1], which means that each vertex touches a triangle, hexagon, triangle, and hexagon. The Kagomé lattice is intimately related to other important lattices in percolation; its sites correspond to the bonds of the honeycomb lattice, which implies that the percolation thresholds p_c for those two systems are the same, $1 - 2\sin(\pi/18)$, and by duality they are also equal to one minus the threshold for bond percolation on the triangular lattice [2].

For bond percolation on the Kagomé lattice, no exact expression for p_c is known, although two conjectures were made a number of years ago, both within the larger context of the q-state Potts model from which percolation follows in the limit of $q = 1$. Wu [3] conjectured that p_c is the solution to

$$p^6 - 6p^5 + 12p^4 - 6p^3 - 3p^2 + 1 = 0 \tag{1}$$

which yields

$$p_c(\text{Wu}) = 0.524\,429\,718. \tag{2}$$

Subsequently, Enting and Wu [4] showed that the general conjecture is not valid for $q = 3$, thereby casting doubt on its validity for $q = 1$. Tsallis [5] conjectured that p_c satisfies

$$-p^3 + p^2 + p = 1 - 2\sin(\pi/18) \tag{3}$$

which yields

$$p_c(\text{Tsallis}) = 0.522\,372\,078. \tag{4}$$

† E-mail address: rziff@umich.edu
‡ E-mail address: psuding@umich.edu

0305-4470/97/155351+09$19.50

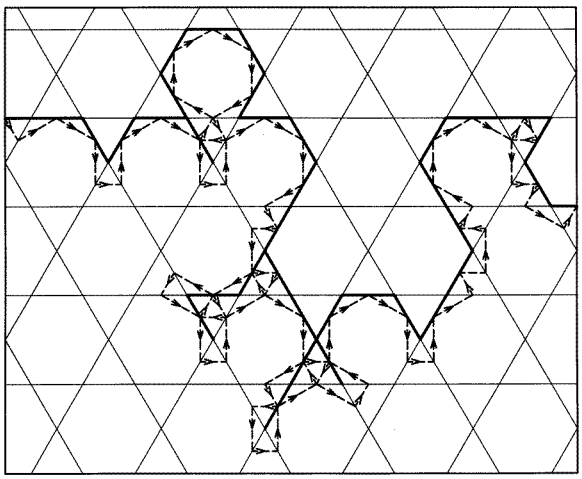

Figure 1. The Kagomé lattice, showing a path of bonds that represent the occupied bonds of the frontier of the percolating region, which is above it (gradient increases in the vertical direction). The broken curve shows the hull walk, which 'bounces' back and forth between centres of the occupied and vacant bonds of the hull.

Note that the quantity on the right-hand side of (3) is identical to the threshold for site percolation on the Kagomé lattice and is the solution to the cubic equation $y^3-3y^2+1=0$, so it follows that (4) is the solution to the ninth-order equation

$$p^9 - 3p^8 + 8p^6 - 6p^5 - 6p^4 + 5p^3 + 3p^2 - 1 = 0. \tag{5}$$

The only existing numerical values of p_c of relatively high precision appear to be those of Yonezawa *et al* [6], who found 0.5244 ± 0.0002, and van der Marck [7], who found 0.5243 ± 0.0004. These results clearly favour Wu's value over Tsallis'. Note that Hu *et al* [8] have also recently presented numerical evidence that Wu's conjecture still works quite well (and better than Tsallis') for the Potts model of various q. In order to investigate the validity of these conjectures further, and to provide an accurate value of p_c for use by others [9], we have carried out a new numerical study to determine the percolation threshold for bond percolation on the Kagomé lattice.

2. Method

The method we employ is the hull-gradient method [10], in which the gradient-percolation frontier is created by a hull-generating walk. In gradient percolation [11, 12], a linear gradient in p is imposed on the lattice in the vertical direction; as the height increases, the occupied bond density p also increases. The estimate of p_c is related to the average position of the frontier of the percolating region. To simultaneously create and measure that frontier, a hull-generating walk is employed [13, 14]. In this walk, the status of a bond (whether occupied or vacant) is determined when the bond is visited by generating a random number and comparing that number with the occupation probability for that height.

The efficiency of this method derives from the fact that in the hull-generating method, the entire lattice is not full before the walk begins. Rather, the lattice is initialized with all bonds undetermined, and the state of the bonds are decided only when they are visited. If the walk does not reach a given bond, then the status of that bond remains undetermined,

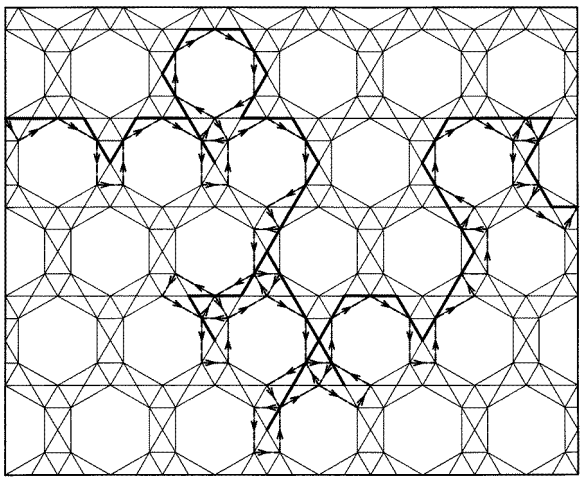

Figure 2. The (6,4,3,4)-Archimedean lattice, on which the hull walk of figure 2 effectively takes place. The trajectory of the walk is equivalent to a mirror-model trajectory [21, 22] on the (6,4,3,4)-lattice, in which the bonds on the underlying Kagomé lattice act as the mirrors.

and no random number need be generated. The error of this method is at the statistical limit $0.5N^{-1/2}$, where N is the total quantity of random numbers generated. Previously, we used the hull-gradient method to find p_c for site percolation on the square lattice to six significant figures, $0.592\,7460 \pm 0.000\,0005$ [10, 15]. This value was confirmed using a different method (also implemented using hulls) [16], and is consistent with the most precise value $0.592\,77 \pm 0.000\,05$ [17] obtained using the more traditional average crossing-probability method [18]. In [10], we also applied the hull-gradient method to determine p_c for site percolation on the Kagomé lattice, and found a value ($0.652\,704 \pm 0.000\,009$) in agreement with the theoretical prediction mentioned above.

For bond percolation, the hull walk simplifies to a trajectory that 'bounces' back and forth between the centres of the occupied and vacant bonds of the hull, as first noted by Grassberger [19] for the case of bond percolation on the square lattice. For the Kagomé lattice, a similar method can be used. As shown in figure 1, the walk moves along line segments that connect centres of adjacent bonds. These line segments produce a new lattice whose topology is the Archimedean (6,4,3,4)-lattice [1] shown in figure 2. Here, the walker turns clockwise when an occupied bond is hit, and anticlockwise when a vacant bond is hit, so that the bonds are effectively rotators [20] or mirrors [21, 22] on the vertices of the (6,4,3,4)-lattice. An occupied bond (probability p) on the Kagomé lattice corresponds to a mirror placed tangent to the vertex of the hexagon on the (6,4,3,4)-lattice, while a vacant bond (probability $1 - p$) corresponds to a mirror that intersects the hexagon. The hull walk is then a mirror-model trajectory [21, 22] on the (6,4,3,4)-lattice.

Many other representations of the hull walk can also be made. The vacant bonds on the Kagomé lattice can be associated with occupied bonds on the dice (or 'diced') lattice shown in figure 3, which is dual to the Kagomé lattice, and the walk creates a hull on that lattice also. The hull trajectory is also a self-avoiding trail on the directed (6,4,3,4)-lattice, with opposing direction vectors at each vertex. The hulls on this lattice can also be produced by a random tiling similar to [23], with 'kite'-shaped tiles having weights p and $1 - p$ as shown in figure 4.

In gradient percolation, the hull of the percolating region resides on bonds whose average

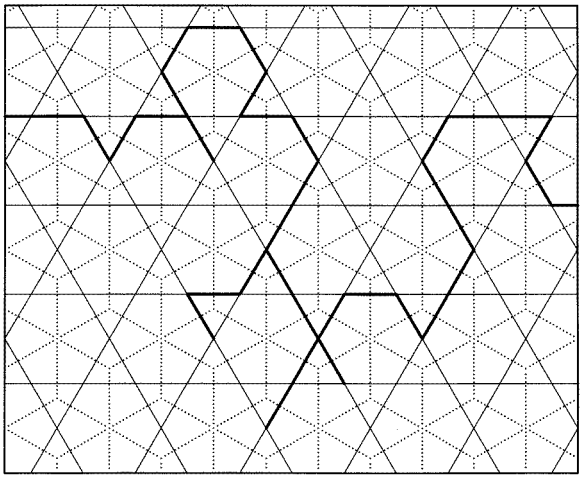

Figure 3. The same hull as in figure 2 on the dice (or 'diced') lattice, the dual to the Kagomé lattice. Vacant bonds on the Kagomé lattice correspond to occupied bonds on the dice, and $p_c(\text{dice}) = 1 - p_c(\text{Kagomé})$.

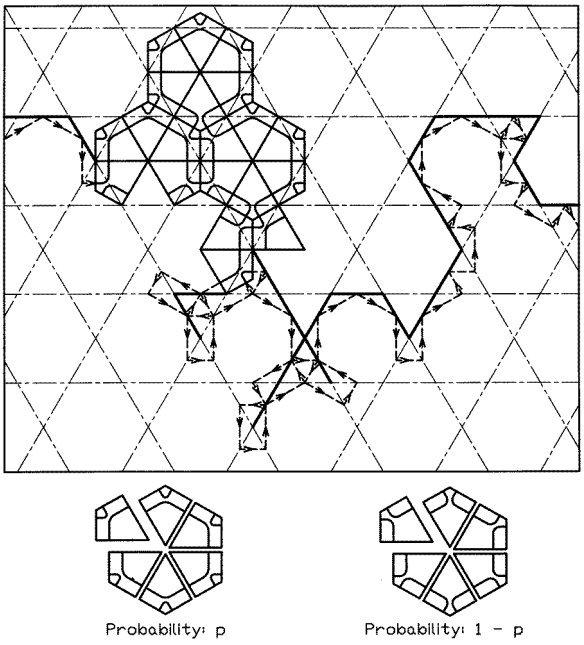

Figure 4. A tiling representation of the hull walk on the Kagomé lattice, shown in the central part of the figure. The tile marked with probability p corresponds to an occupied bond on the underlying Kagomé lattice, while the one marked $1 - p$ corresponds to a vacant bond.

value of p gives an estimate $p_c(g)$, which approaches p_c as the gradient $g \equiv |\nabla p|$ goes to zero [11, 12]. Equivalent to taking the average value of p on bonds of the hull, one can

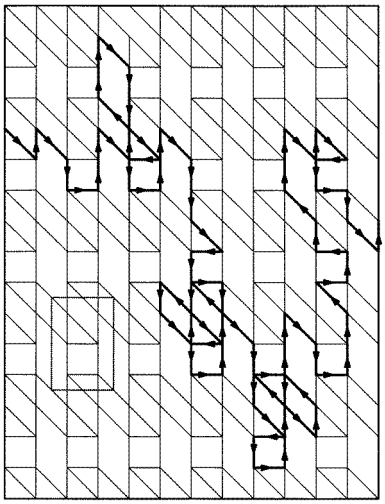

Figure 5. The representation of the (6,4,3,4)-lattice on a rectilinear grid, as utilized in the computer program's two-dimensional array.

simply take as the estimate [12]

$$p_c(g) = \frac{n_{\rm occ}}{n_{\rm occ} + n_{\rm vac}} \tag{6}$$

since the expected fraction of occupied bonds in the hull equals the average value of p of the vertices that belong to the hull. Here, $n_{\rm occ}$ is the number of vertices corresponding to occupied bonds and $n_{\rm vac}$ is the number of vertices corresponding to vacant bonds in the hull.

To represent the (6,4,3,4)-lattice on the computer efficiently, it is necessary to transform it to align on a rectilinear grid. One way to do this is shown in figure 5, where the basic rectangle of six sites is repeated in every second column and every third row. While we have distorted the lattice laterally to accommodate the square lattice periodicity, we have not shifted any vertices vertically with respect to each other, so as not to affect the gradient in the vertical direction. In producing that gradient, we made the change in p proportional to the actual height, so that changes between the wider rows in figure 2 equal twice the change of the narrower ones. The gradient g is defined here by $g = \Delta p/\ell$ where Δp is the change of p between the wider rows, and ℓ is the bond length, taken to be unity. (We also considered two alternate representations where the gradient was not precisely uniform on a local scale; the behaviour of these systems is discussed in the appendix.) A three-dimensional array was used to store the six possible outgoing directions based upon the two incoming directions, the six types of vertices, and the status (occupied or vacant) of the vertex.

The lattice was initialized by filling the first column halfway with occupied bonds and the rest with vacant bonds, which prevents the walk from closing on itself at the start. With the gradient in the vertical direction, the walk naturally drifts to the right. Periodic boundary conditions were applied in the horizontal direction, and each new column to the right was cleared off as it was first visited. This allows the simulation to run indefinitely and have essentially no boundary effects from the horizontal ends of the system. We tracked the maximum distance the walk wandered to the left of the moving front in order to confirm that the system width was sufficient to preclude wraparound errors.

To rule out systematic errors related to random number generation, three different generators were tried. For most of the runs we used the shift-register sequence generator

$R_7(9689)$ [16, 24] defined by

$$x_n = x_{n-471} \hat{} x_{n-1586} \hat{} x_{n-6988} \hat{} x_{n-9689} \tag{7}$$

where $\hat{}$ is the bitwise exclusive or operation. This 'four-tap' generator is equivalent to decimating by 7 (taking every seventh term) of the sequence generated by the two-tap rule R(9689), $x_n = x_{n-471} \hat{} x_{n-9689}$, which follows from [25]. This decimation has the effect of vastly reducing the three- and four-point correlations of the two-tap generator, and was previously found to yield good behaviour for problems of this type [26]. For the system with the second smallest gradient, we considered two additional random number generators. The second generator, $R_{21}(9689)$, was obtained by further decimating $R_7(9689)$ three times, simply by using every third number, which may yield better statistical properties. For the third generator, we used a traditional congruential generator CONG, but with a very large modulus of 64 bits [27]:

$$x_n = (5081\,641\,266\,417\,562\,522 x_{n-1} + 11) \bmod 2^{64}. \tag{8}$$

First, to study the general finite-size behaviour, we considered lattices of height $H = 128, 256, 512, 1024, 2048$ and 4096, with widths sufficient to avoid wraparound error, and p ranging all the way from 0 to 1. For these systems, $g = 1.5/H$, the factor of 1.5 resulting from the changing increment of p between the wide and narrow rows as described above. Approximately 10^{11} occupied plus vacant vertices were generated for each of these lattices. Then, to obtain a precise final value, we used systems with very small gradients $g = 2.564 \times 10^{-5}$ and 7.324×10^{-6} by using lattices of height 4096 and 8192, with p ranging from 0.49 to 0.56 and from 0.505 to 0.545, respectively; 2×10^{12} steps were carried out for each of these systems. In all, several months of workstation computer time were used to obtain the final results.

In the simulations, we kept track of the maximum and minimum heights of the hull. As g decreases, the relative width of the walk decreases as $g^{3/7}$ [12], allowing us to expand the gradient as mentioned above. Note that we also carried out runs on a system of height 64, but ran into difficulty because the walk wandered all the way to a top or bottom boundary where it got stuck in a dead end. Presumably, this could be averted by constructing those boundaries more carefully, but we did not attempt to do it.

3. Results and discussion

First we compare the three random number generators. Table 1 gives the results for runs for $g = 2.564 \times 10^{-5}$, where all generators were used. Error bars represent one standard deviation, and follow from the statistical formula $[p_c(1 - p_c)/N]^{1/2} \approx 0.5N^{-1/2}$ where $N = n_{\mathrm{occ}} + n_{\mathrm{vac}}$, since the occupancy of each vertex is achieved with complete statistical

Table 1. Results for $p_c(g)$ given by (6) for runs with height $H = 4096$ and gradient $g = 0.000\,025\,64$, using three different random number generators (RNG). N is the total number of occupied and vacant bonds generated, and σ represents one standard deviation of error (68% confidence interval).

RNG	N	$p_c(g)$	$\sigma = 0.5N^{-0.5}$
$R_7(9689)$	1.0×10^{12}	0.524 4048	±0.000 0005
$R_{21}(9689)$	0.5×10^{12}	0.524 4053	±0.000 0007
CONG(64-bit)	0.5×10^{12}	0.524 4059	±0.000 0007
Average	2.0×10^{12}	0.524 4052	±0.000 0004

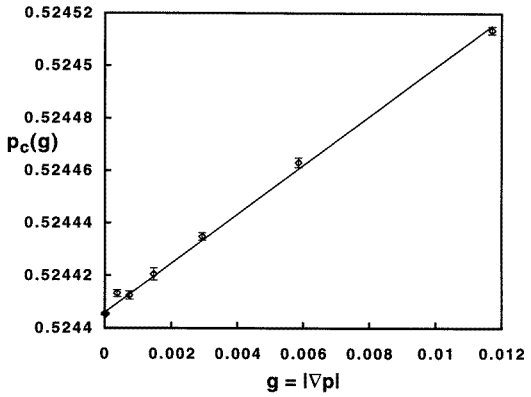

Figure 6. A plot of the estimate $p_c(g)$ determined by (6), versus the gradient g, implying the linear relation (9). The error bars represent one standard deviation of statistical error.

independence. Clearly, the three random number generators give statistically consistent results, and we thus averaged their results to get the final data point for this g.

Our complete results are shown in figure 6, where we plot $p_c(g)$ versus g for all the lattices we considered. This plot provides good evidence that the dependence of $p_c(g)$ upon g is linear (as observed by Rosso *et al* [12] for the case of site percolation on the square lattice) with the behaviour

$$p_c(g) = p_c + 0.010g. \tag{9}$$

The slope 0.010 implies that the average value of p differs from p_c by an amount corresponding to one hundredth of a lattice spacing, independent of the gradient. For $g = 2.564 \times 10^{-5}$, this implies a finite-size correction $p_c(g) - p_c = 2.6 \times 10^{-7}$ that is somewhat smaller than the error bars given in table 1. For $g = 7.324 \times 10^{-6}$, the simulations of 2.2×10^{12} steps using $R_7(9689)$ yielded $0.524\,4055 \pm 3.4 \times 10^{-7}$. Here, the finite-size correction is insignificant compared with the statistical error. Combining these results, we obtain our final result

$$p_c = 0.524\,4053 \pm 0.000\,0003. \tag{10}$$

This result is consistent with—but nearly 1000 times more precise than—previous values [6, 7]. It evidently agrees with neither Tsallis' nor Wu's predictions, although the difference with Wu's approximate conjecture is remarkably small, only 47 parts per million (but still much larger than our error bars of less than one part per million). Note that it would be quite difficult to observe this small difference in p_c using conventional methods, (e.g. [6–8, 17, 18, 28, 29]).

If neither Wu's nor Tsallis' conjecture is valid, is there perhaps some other simple polynomial that yields p_c? In the absence of a theory, we can search numerically for possible candidates consistent with our numerical result. However, if we allow the maximum order of the polynomial to be six, and the integer coefficients to be as large as say ± 24 (except for the leading coefficient, which we restrict to unity), then we find literally thousands of polynomials with roots within two standard deviations of (10). Some examples are:

$$p^4 + 7p^3 + 17p - 10 = 0 \qquad p_c = 0.524\,405\,335 \tag{11a}$$

$$p^4 - 24p^2 + p + 6 = 0 \qquad p_c = 0.524\,405\,671 \tag{11b}$$

$$p^5 - 2p^4 - 2p^3 + 16p^2 - 4 = 0 \qquad p_c = 0.524\,405\,424 \tag{11c}$$

$$p^5 + 5p^3 - 8p^2 + 18p - 8 = 0 \qquad p_c = 0.524\,404\,863 \tag{11d}$$

$$p^6 + 3p^5 - 3p^3 + 12p - 6 = 0 \qquad p_c = 0.524\,405\,290 \tag{11e}$$

$$p^6 + 5p^5 + 7p^3 - 5p^2 + 6p - 3 = 0 \qquad p_c = 0.524\,405\,306 \tag{11f}$$

$$p^6 + 3p^5 + 9p^4 - p^3 + p^2 + 2p - 2 = 0 \qquad p_c = 0.524\,405\,134. \tag{11g}$$

Note also that $(\frac{11}{40})^{1/2} = 0.524\,404\,424$ is only slightly low. Unfortunately, it is not possible by numerical means to determine p_c with sufficient accuracy to distinguish which of these many polynomials is the correct one (if indeed one is!).

Note finally that (10) implies that the bond threshold of the dice lattice shown in figure 3 is given by

$$p_c(\text{dice}) = 1 - p_c(\text{Kagomé}) = 0.475\,5947 \pm 0.000\,0003. \tag{12}$$

Appendix

Besides the system described above with the gradient applied completely uniformly, we also considered two systems in which gradient was not constructed so precisely on a local scale, and it is instructive to see their effects on the finite-size behaviour. First, we squared-off the (6,3,4,3)-lattice by 'stretching' it horizontally, leading to a lattice similar to figure 5 but rotated by 90°. Thus, we effectively pushed up and down alternating columns in the original lattice, and the gradient was applied equally between all the rows in this distorted lattice. The idea was that these local variations should have little effect on the behaviour when the gradient is small. However, the deviations turned out to be rather large, until the gradient dropped to about 0.002, as shown in figure 7 (case A). As a second test (case B in figure 7), we represented the lattice as in figure 5, but applied the gradient equally between all rows, whether 'wide' or 'narrow'. Again, the behaviour of the finite-size corrections to $p_c(g)$ differed from the uniform case, but not as much here. In the limit of g small, where p changes very little from row to row, all systems followed the same limiting finite-size behaviour as (9). These results show, however, that for the linear behaviour to remain valid for moderately large gradients, a uniform gradient must be applied to the lattice in its actual configuration.

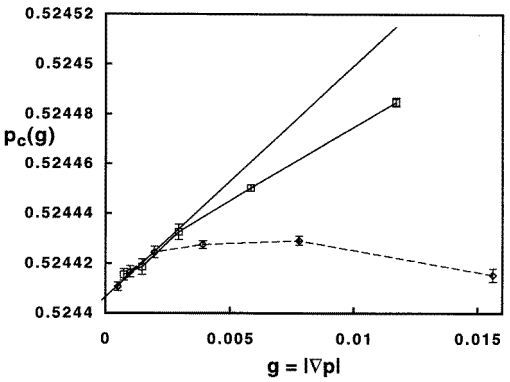

Figure 7. A similar plot as in figure 6, with data from two systems (case A, broken curve through data points, and case B, full curve through data points) in which the gradient is not locally uniform, as described in the appendix. The full line represents the fit of the data of figure 6 for the system the uniform gradient. For small g, all systems follow the same asymptotic behaviour.

Acknowledgments

The authors thank A J Guttmann for interesting them in this problem, and thank F Y Wu for comments. This material is based upon work supported by the US National Science Foundation under grant no DMR-9520700.

References

[1] Grünbaum B and Shephard G C 1987 *Tilings and Patterns* (New York: Freeman)
[2] Sykes M F and Essam J W 1964 *J. Math. Phys.* **5** 1117
[3] Wu F Y 1979 *J. Phys. C: Solid State Phys.* **12** L645
[4] Enting I G and Wu F Y 1982 *J. Stat. Phys.* **28** 351
[5] Tsallis C 1982 *J. Phys. C: Solid State Phys.* **15** L757
[6] Yonezawa F, Sakamoto S and Hori M 1989 *Phys. Rev.* B **40** 636
[7] van der Marck S C 1997 *Phys. Rev.* E **55** 1514
[8] Hu C-K, Chen J-A and Wu F Y 1994 *Mod. Phys. Lett.* B **8** 455
[9] Guttmann A J and Jensen I 1997 to be published
[10] Ziff R M and Sapoval B 1986 *J. Phys. A: Math. Gen.* **18** L1169
[11] Sapoval B, Rosso M and Gouyet J F 1985 *J. Physique Lett.* **46** L149
[12] Rosso M, Gouyet J F and Sapoval B 1986 *Phys. Rev.* B **32** 6053
[13] Ziff R M, Cummings P T and Stell G 1984 *J. Phys. A: Math. Gen.* **17** 3009
[14] Weinrib A and Trugman S 1985 *Phys. Rev.* B **31** 2993
[15] Ziff R M and Stell G 1988 *UM LaSC Report* 88-4, unpublished
[16] Ziff R M 1992 *Phys. Rev. Lett.* **69** 2670
[17] Gebele T 1984 *J. Phys. A: Math. Gen.* **17** L51
[18] Stauffer D and Aharony A 1994 *An Introduction to Percolation Theory* 2nd revised edn; (London: Taylor and Francis)
[19] Grassberger P 1984 *J. Phys. A: Math. Gen.* **19** 2675
[20] Gunn J M F and Ortuño M 1985 *J. Phys. A: Math. Gen.* **18** L1096
[21] Cao M-S and Cohen E G D 1997 *J. Stat. Phys.* to be published
[22] Wang F and Cohen E G D 1995 *J. Stat. Phys.* **81** 467
[23] Roux S, Guyon E and Sornette D 1988 *J. Phys. A: Math. Gen.* **21** L475
[24] Golomb S W 1967 *Shift Register Sequences* (San Francisco, CA: Holden-Day)
[25] Zierler N 1969 *Inf. Control* **15** 67
[26] Ziff R M unpublished
[27] Wood W W 1995 Private communication
[28] Reynolds P J, Stanley H E and Klein W 1980 *Phys. Rev.* B **21** 1223
[29] Hu C-K 1992 *Phys. Rev. Lett.* **69** 2739