Asynchronously parallelized percolation on distributed machines

Nicholas R. Moloney*
Blackett Laboratory, Imperial College London, Prince Consort Road, London SW7 2BW, United Kingdom

Gunnar Pruessner†
Department of Mathematics, Imperial College London, 180 Queen's Gate, London SW7 2BZ, United Kingdom

(Received 1 November 2002; revised manuscript received 3 December 2002; published 20 March 2003)

We propose a powerful method based on the Hoshen-Kopelman algorithm for simulating percolation asynchronously on distributed machines. Our method demands very little of hardware and yet we are able to make high precision measurements on very large lattices. We implement our method to calculate various cluster size distributions on large lattices of different aspect ratios spanning three orders of magnitude for two-dimensional site and bond percolation. We find that the nonuniversal constants in the scaling function for the cluster size distribution apparently satisfy a scaling relation, and that the moment ratios for the largest cluster size distribution reveal a characteristic aspect ratio at $r \approx 9$.

DOI: 10.1103/PhysRevE.67.037701 PACS number(s): 02.70.−c, 05.10.Ln, 05.70.Jk

Although an old problem [1], percolation continues to attract a steady stream of papers [2–4]. High-quality numerical data are required to corroborate the many analytical results, particularly from conformal field theory [5–8]. In this paper, we describe a method of simulating percolation that runs asynchronously in parallel on almost any hardware. In principle, the method relaxes all the standard constraints in numerical simulations of percolation, such as CPU power, memory, and network capacity. It is especially suited for calculating cluster size distributions, finite size corrections, crossing probabilities, and, by applying the corresponding boundary conditions, distributions of wrapping clusters on different topologies, e.g., cylinder, torus, or the Möbius strip.

The Hoshen-Kopelman algorithm (HKA) [9] is still the standard technique for identifying clusters in percolation, where a cluster is a set of sites connected via nearest neighbor interactions (site percolation) or active bonds (bond percolation). Strictly speaking, it is a type of data representation particularly suited for tracking clusters. Recently, Newman and Ziff [4] have shown how to exploit this data representation to monitor the change in various observables as the occupation probability p is increased. The data representation efficiently encodes the connectivity of clusters in a large percolation system. In this paper we show how to exploit this representation for different system sizes (up to $\approx 5 \times 10^{14}$ sites) and aspect ratios. The algorithm runs asynchronously in parallel over an almost arbitrarily slow network of computers. The network is hierarchically organized, and nodes on lower levels (slave nodes) can be slow and heterogeneous. In fact, the system scales like an ideal parallel computer: the overall computing time decreases linearly with the number of nodes, especially for large slave lattices (patches) where the overhead due to networking and related processing and the CPU-time at higher levels (master nodes) becomes negligible (see Table I). Other memory-efficient methods exist for constructing large clusters, for example Paul, Ziff, and Stanley [10]. In that paper a variant of the Leath algorithm is used [11], together with a data structure to record information about visited sites. As a result, memory is made available as and when it is required. In our method we can easily count the number of spanning clusters per realization, apply different boundary conditions, rearrange patches for different aspect ratios, and gather statistics at every stage of lattice construction.

We describe our method in detail for two-dimensional site percolation on a square lattice and present the overall cluster size distribution for different aspect ratios, as well as the universal moment ratios for the distribution of the order parameter in site and bond percolation. We find two surprising results. First, the nonuniversal amplitudes in the scaling function for the cluster number distribution numerically satisfy a scaling relation. Second, the moment ratios for the largest cluster size distribution all peak at $r \approx 9$, defining a characteristic aspect ratio.

The basic idea of the method is that many slave nodes independently simulate lattices of equal linear size L in parallel using the HKA. These nodes send a special representation of their lattice border to a master node, which combines m of these patches to form a superlattice. The advantage of such a decomposition is that the master node can buildup a very large superlattice while maintaining the large scale histograms. The master node can apply different boundary conditions and even reuse the same patches several times by

TABLE I. The optimal number of slaves and relative networking overhead of the slave nodes. The master node used was roughly twice as fast as the slave nodes and applied six different boundary conditions on 14 different aspect ratios from each set of 900 patches of size L^2 produced by the slaves.

L	Slave nodes per master	Approximate overhead
100	2	4.8%
200	4	2.9%
500	10	1.4%
1000	22	1.7%

*Electronic address: n.moloney@imperial.ac.uk
†Electronic address: gunnar.pruessner@physics.org

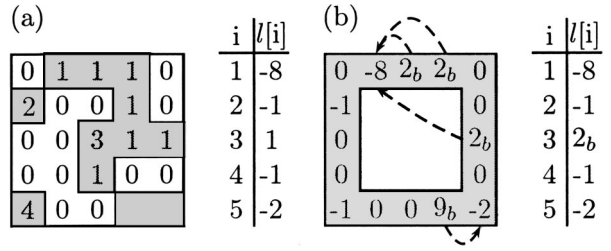

FIG. 1. (a) The lattice and the list of labels, $l[i]$, as prepared by the HKA. (b) The border configuration suitable for the master node, after a clockwise border scan starting in the upper left-hand corner, with the list of labels now being irrelevant. For the reader's convenience, labels pointing to sites i in the new border carry a suffix i_b.

rotating, mirroring, and permuting them.

The key to our algorithm is the representation of the lattice borders by the slave nodes. This is essentially a form of path compression (or Nakanishi label recycling [12]), where all border sites are considered active (i.e., possibly changing connectivity) and bulk sites are considered inactive. In this representation information about the connectivity and size of any cluster connected to the border is summarized entirely within border sites. The spatial information of clusters is neither required nor stored. Thus clusters not connected to the border are ignored, although their contribution to the cluster size histogram is recorded locally.

The HKA produces a list of labels, to which all active sites refer in order to identify their cluster, see Fig. 1(a). After the realization of a lattice, a *new* border representation is prepared by visiting each border site in succession, indexed from 1 to $4L-4$, see Fig. 1(b). The first site of a previously unscanned cluster contains the size of the cluster as a negative value in the range $[-1,-L^2]$. This site is called the root. In the list of labels of the original representation, the label of this cluster is changed to indicate the new location of the root site in the border. All other sites in the border which belong to the same cluster refer to this site. The slave nodes send the border configuration in this representation to the master node. If required, clusters in the bulk have

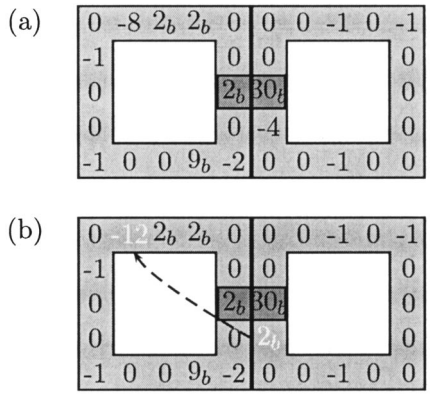

FIG. 2. (a) The configuration of the borders before two clusters merge at the marked labels. The labels in the right patch are shifted by $4L-4$ to make them unique. (b) The configuration of the borders after the merging procedure. Labels which have changed are shown in white.

their sizes recorded in a local histogram, i.e., at the slave node that produced the lattice. This histogram is stored locally for the duration of the simulation. The master node is the only component that requires enough memory to hold the large histogram(s) usually generated in large scale simulations, while the slaves only need to store a very small amount of local data.

When two patches are combined by the master (gluing) it is possible that two clusters merge at the border. This is realized by setting one of the root labels (preferably from the smaller cluster) to point to the other, as shown in Fig. 2. The master's histogram is updated by removing both cluster sizes (4 and 8 in the example) and replacing them by their sum. By adding the site-normalized histogram of the slaves (i.e., the number density of s clusters), $n_{\mathrm{slv}}(s)$, to the site-normalized histogram on the master node, $n_{\mathrm{mst}}(s)$, the total histogram, $n(s)$ is obtained,

$$n(s)=n_{\mathrm{mst}}(s)+n_{\mathrm{slv}}(s). \tag{1}$$

This result does not involve any approximation and is independent of the number of realizations. Because the superposition Eq. (1) can be postponed until postprocessing, the slaves can store these data locally. Moreover, because all relevant information is encoded in the patches, when and whence they arrive at the master node is arbitrary. Hence the algorithm is asynchronous, in contrast to standard techniques of parallelization, for example Ref. [13].

The master node can itself be considered a slave node and prepare a border configuration for another master node, so that one obtains a treelike structure of master and slave nodes, where statistics can be obtained on every level. We have used this scheme to produce a single lattice of size $(22.2\times10^6)^2$ sites, and have calculated its cluster size distribution. For large L the CPU-time required for networking becomes negligible, as shown in Table I. The complexity of the master gluing algorithm is $\mathcal{O}(mL\log L)$, while the slaves need $\mathcal{O}(L^2\log L)$ time to produce a patch, which is represented in $\mathcal{O}(L)$ memory. Therefore, the optimal number of slaves per master in which the master fully utilizes its resources, while not blocking any slaves, scales like L. At the same time, the relative networking overhead per slave scales like $1/L$. Table I shows the corresponding measurements.

Debarring correlations introduced by the random number generator, all patches arriving at the master are statistically independent. However, it is possible to recycle incoming patches by arranging them in different configurations (e.g., boundary conditions or aspect ratios). The results for these different configurations are *not* statistically independent. An upper bound can be calculated for the error introduced by this procedure. Rather than recycling all patches q times (for example, for $q=14$ different aspect ratios), one could distribute them evenly among q bins, now all statistically independent. The error in the estimator for the mean of an observable in the q bins would be a factor $\sqrt{q}$ larger than that for the complete sample. Therefore, when considering results for q bins while using the same complete sample in each bin, the upper bound for the error is $\sqrt{q}$ times the error for the complete set. When patches are recycled it is possible to reduce

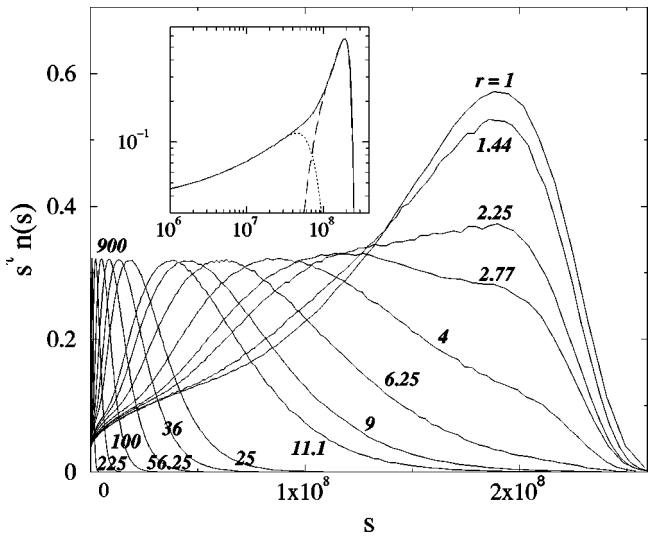

FIG. 3. The rescaled and binned distribution $s^\tau n_s(s)$ for systems containing $N=30\,000^2$ sites. The inset shows $s^\tau n_s(s)$ (solid line), $s^\tau n_{\max;s}(s)/N$, (long-dashed line), and their difference (dotted line), for $r=1$. Evidently, the bump in the distribution is derived mainly from the largest clusters.

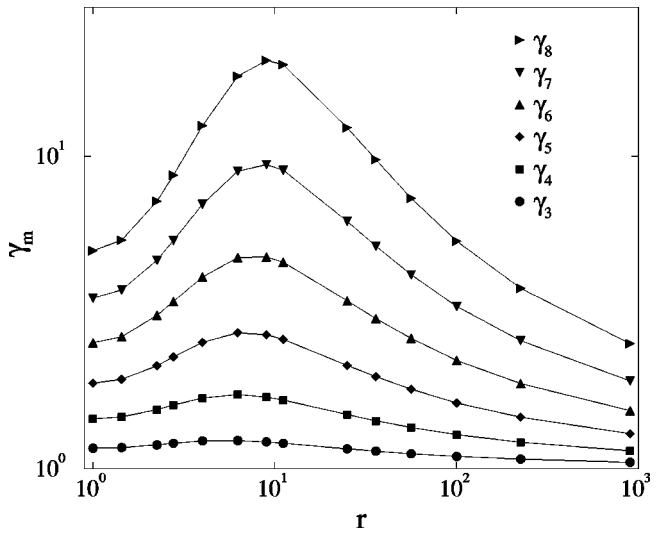

FIG. 4. Universal moment ratios $\gamma_{m;s,b}(r)$ for different aspect ratios and system sizes.

the correlations by randomly rotating, mirroring and permuting them. We have done this in all simulations.

As an application of the algorithm, various cluster size distributions for site and bond percolation for $q=14$ different aspect ratios, $r=$ width/height, between 1 and 900 were calculated. The slaves produced square patches of three different sizes, $L=10,100,1000$, of which $m=900$ were glued at the master node to form q superlattices with $N=300^2,3000^2,30\,000^2$ sites. The simulations were performed at critical density $p_c=0.592\,746\,21$ for site percolation [4], and $p_c=1/2$ for bond percolation [14]. All numerical results are based on at least 10^6 independent realizations (i.e., roughly 10^9 realizations at the slave nodes). Free boundary conditions have been applied everywhere. The random number generator used was the so-called Mersenne-Twister [15], which is highly suitable for parallel simulations.

The site-normalized cluster size distribution $n_{s,b}(s;r)$ is the number density of s-clusters for aspect ratio r. Henceforth, subscripts s and b refer to site and bond percolation, respectively. For large cluster sizes near p_c, $n_{s,b}(s;r)$ is expected to behave like

$$n_{s,b}(s;r)=a_{s,b}(r)s^{-\tau}\mathcal{G}(s/s^0_{s,b};r), \tag{2}$$

where, in a finite system of effective size $\tilde{L}$, $s^0_{s,b}=b_{s,b}(r)\tilde{L}^D$, and $\mathcal{G}$ is the scaling function. The effective size can be taken as anything that scales linearly in $\sqrt{N}$. The universal critical exponents are τ and D, while the amplitudes $a_{s,b}(r)$ and $b_{s,b}(r)$ are nonuniversal, and set by two arbitrary conditions on $\mathcal{G}$. Figure 3 shows $s^\tau n_s(s;r)$ for different values of r, using $\tau=187/91$ [16].

Two interesting features emerge. Independently of L, the shape of the distribution changes abruptly at around $r=2.25$ and the maximum of the rescaled distribution is seemingly constant for larger r. Therefore, there is no possible choice of $\tilde{L}$ that can collapse the scaling function for different aspect ratios, and $\mathcal{G}$ explicitly depends on r. The inset in Fig. 3 shows $n_s(s;r)-n_{\max;s}(s;r)/N$ at $r=1$, where $n_{\max;s,b}(s;r)$ denotes the distribution of the size of the largest cluster. It seems that the sudden change in the shape of $n_{s,b}(s;r)$ is caused by a change in $n_{\max;s,b}(s;r)$, but what happens at this particular value of r remains an open question.

If we define the moment ratios as

$$V_{m;s,b}(r)\equiv\frac{\langle s^m\rangle_{s,b}(r)N}{\langle s^2\rangle^{m/2}_{s,b}(r)N^{m/2}} \tag{3}$$

with $\langle s^k\rangle_{s,b}(r)\equiv\int s^k n_{s,b}(s;r)ds$, then site and bond percolation should differ by powers of the factor

$$\frac{a_s(r)/a_b(r)}{(b_s(r)/b_b(r))^{\tau-1}}, \tag{4}$$

which is obtained by calculating the moments with the help of Eq. (2). However, we find numerically that this factor is unity, i.e., that the ratio $a(r)/b(r)^{\tau-1}$ is the same for site and bond percolation. This ratio is not a universal function, because its value depends on the conditions imposed on $\mathcal{G}$ for determining $a(r)$ and $b(r)$. However, numerics suggests strongly that, once these conditions are given, this ratio is independent of the lattice type, i.e., Eq. (3) represents a *universal* moment ratio. Therefore, it is possible to write

$$a_{s,b}(r)=b^{\tau-1}_{s,b}(r)q(r), \tag{5}$$

where $q(r)$ depends only on the choice of the two conditions imposed on $\mathcal{G}$, but not on the lattice type. As mentioned above, $\mathcal{G}$ is necessarily an explicit function of r, so that it can absorb $q(r)$ defined in Eq. (5), thereby fixing one of the two conditions on $\mathcal{G}$. The remaining condition determines (together with the choice of $\tilde{L}$) the remaining free parameter. Consequently, we conclude that Eq. (2) can be replaced by

$$n_{s,b}(s;r) = b_{s,b}^{\tau-1}(r) s^{-\tau} \tilde{\mathcal{G}}(s/(N^{D/2} b_{s,b}); r).$$

Of course, $b_{s,b}(r)^{\tau-1}$ cannot be absorbed into $\mathcal{G}$ in the same way as $q(r)$ because it depends on the lattice type. Thus all the characteristics of the lattice enter solely through b. For completeness we note that numerically the ratios $a_s(r)/a_b(r)$ and $b_s(r)/b_b(r)$ are independent of r, no matter what conditions are imposed on $\mathcal{G}$.

The order parameter of percolation is the fraction of sites belonging to the spanning (or largest) cluster. Thus, one expects the moment ratios

$$\gamma_m \equiv \frac{\langle s^m \rangle_{\max;s,b}(r)}{\langle s^2 \rangle_{\max;s,b}^{m/2}(r)} \tag{6}$$

with $\langle s^k \rangle_{\max;s,b}(r) \equiv \int s^k n_{\max;s,b}(s;r) ds$ to be universal. Figure 4 shows the behavior of this ratio for different aspect ratios r. A pronounced bump appears at around $r=9$. The origin of the bump remains unclear, and can be used to define a characteristic aspect ratio.

In conclusion, the method proposed in this paper permits the use of resources usually considered too slow, small, or badly connected. At the same time, it takes advantage of parallelization by providing a very flexible framework for simulating different boundary conditions and aspect ratios. By way of illustration, we have increased Tiggemann's former world record [13] for the largest simulated system by a factor of 30. The new record was set by an undergraduate computer cluster (as opposed to a Cray T3E) when idle. The data presented have remarkable numerical accuracy and are from systems of unprecedented size. They give rise to a number of urgent questions, namely, how to reconsider the nonuniversal amplitudes in Eq. (2), and how to account for a characteristic aspect ratio as provided by the moment ratios of the largest cluster size distribution.

The authors wish to thank Andy Thomas for his fantastic technical support. Without his help and dedication, this project would not have been possible. The authors are grateful for the generous donation of ''I-D Media AG, Application Servers & Distributed Applications Architectures, Berlin.'' We especially thank Matthias Kaulke and Oliver Kilian. The authors also thank Dan Moore, Brendan Maguire, and Phil Mayers for their continuous support, as well as Kim Christensen for his helpful comments. N.R.M. is very grateful to the Beit Foundation, and to the Zamkow family. G.P. gratefully acknowledges the support of the EPSRC.

[1] P.J. Flory, J. Am. Chem. Soc. **63**, 3091 (1941).
[2] B. Duplantier, Phys. Rev. Lett. **82**, 3940 (1999).
[3] M. Aizenman, B. Duplantier, and A. Aharony, Phys. Rev. Lett. **83**, 1359 (1999).
[4] M.E.J. Newman and R.M. Ziff, Phys. Rev. Lett. **85**, 4104 (2000).
[5] J. Cardy, J. Phys. A **25**, L201 (1992).
[6] R. Langlands, C. Pichet, P. Pouliot, and Y. Saint-Aubin, J. Stat. Phys. **67**, 553 (1992).
[7] H.T. Pinson, J. Stat. Phys. **75**, 1167 (1994).
[8] J. Cardy, J. Phys. A **31**, L105 (1998).
[9] J. Hoshen and R. Kopelman, Phys. Rev. B **14**, 3438 (1976).
[10] G. Paul, R.M. Ziff, and H.E. Stanley, Phys. Rev. E **64**, 026115 (2001).
[11] P.L. Leath, Phys. Rev. B **14**, 5046 (1976).
[12] K. Binder and D. Stauffer, in *Applications of the Monte Carlo Method in Statistical Physics*, edited by K. Binder, Topics in Current Physics, Vol. 36, 2nd ed. (Springer-Verlag, Berlin, 1987), pp. 241–275.
[13] D. Tiggemann, Int. J. Mod. Phys. C **12**, 871 (2001); e-print cond-mat/0106354.
[14] H. Kesten, Commun. Math. Phys. **74**, 41 (1980).
[15] M. Matsumoto and T. Nishimura, in *Monte Carlo and Quasi-Monte Carlo Methods 1998* (Springer-Verlag, Berlin, 1998), e-print from http://www.math.h.kyoto-u.ac.jp/matumoto/RAND/DC/dgene.ps.
[16] D. Stauffer and A. Aharony, *Introduction to Percolation Theory* (Taylor & Francis, London, 1994).