# Alternative criterion for two-dimensional wrapping percolation

Hongting Yang
*School of Science, Wuhan University of Technology, Wuhan 430070, P.R. China*
(Received 30 January 2012; published 24 April 2012)

Based on the difference between a spanning cluster and a wrapping cluster, an alternative criterion for testing wrapping percolation is provided for two-dimensional lattices. By following the Newman-Ziff method, the finite size scalings of estimates for percolation thresholds are given. The results are consistent with those from Machta's method.

DOI: 10.1103/PhysRevE.85.042106 PACS number(s): 05.70.Jk, 02.70.Uu, 05.10.Ln, 64.60.ah

## I. INTRODUCTION

In a square lattice each site is independently either occupied with probability $p$ or empty with probability $1 - p$. A cluster is a group of occupied (nearest) neighbor sites [1]. With more and more sites being occupied, clusters grow larger and larger. For the square lattice with free boundaries, once a cluster grows large enough to touch the two opposite boundaries, the spanning percolation occurrs. This cluster is called a spanning cluster. While for the lattice with periodic boundary conditions it is somewhat more difficult to detect a wrapping cluster, which wraps around the lattice.

Since the very early work on three-dimensional polymers [2], percolation has found a variety of uses in many fields. Typically the study of networks [3–6] has attracted much interest. Recent studies of percolation in physics mainly involve various functional materials or components, such as optical lattices [7], magnetic materials or ordinary materials [8–10], and nanocomposites [11–13]. Very recently, the study on explosive percolation generated "explosive" interest [14–17]. Many other interesting works on percolation theory and its wide applications are collected in some books [1,18,19].

In all these studies of percolation, one question is how to tell the onset of percolation. The percolation threshold (or critical probability) $p_c$ is the value of $p$ for which a spanning cluster (in the case of free boundaries) or a wrapping cluster (periodic boundary conditions) appears for the first time [20,21]. For a periodic lattice, once a wrapping cluster appears, say along the $x$ direction, if we cut the lattice along the $y$ direction to let the $x$ boundaries be open there exists (at least) one spanning cluster. In turn, when a spanning cluster first appears on a free boundaries lattice, if we connect the two opposite open boundaries to let it be a periodic lattice, a wrapping cluster does not necessarily appear. Therefore, wrapping percolation always happens later than spanning percolation for the lattices with the same linear dimension $L$ but with different boundary conditions. The difference between a wrapping cluster and a spanning cluster gives us a clue to build an alternative criterion for wrapping percolation. Once a spanning cluster occurs, we amalgamate those clusters (excluding the spanning cluster) touching the boundaries, build connected relations between the spanning cluster and the amalgamated clusters. If, finally, two ends of the spanning cluster connect to each other via an amalgamated cluster then a wrapping cluster appears. The data processing after obtaining a wrapping cluster follows the Newman-Ziff algorithm [20] exactly.

The prevalent criterion for wrapping percolation introduced by Machta *et al.* [22] has been described in detail for bond percolation in the Newman-Ziff algorithm. It can be easily extended to site percolation once the displacements of neighboring sites to the same root site differ by an amount other than zero or one lattice spacing, the cluster wrapping has occurred. Taking site percolation as an example, the core idea of the Newman-Ziff algorithm is that, starting with an empty lattice, a percolation state can be realized simply by adding sites one by one to the lattice; a sample state with $n + 1$ occupied sites is achieved by adding one extra randomly chosen site to a sample state with $n$ sites. An important technique in the Newman-Ziff algorithm is the application of binomial distribution. Taking the number of occupied sites $n$ as "energy," if a set of $\{Q_n\}$ in the microcanonical ensemble is measured then the observable $Q(p)$ in the canonical ensemble is given by

$$Q(p) = \sum_{n=0}^{N} \binom{N}{n} p^n (1-p)^{N-n} Q_n. \tag{1}$$

Such observables $Q$ can be the probability of cluster wrapping, mean cluster size, correlation length, and so on. The main advantage of this technique is that the continuous observable $Q(p)$ for all $p$ can be determined from discrete $N + 1$ values of $Q_n$.

There are four types of probabilities $R_L(p)$ of cluster wrapping on the periodic square lattice of $L \times L$ sites. $R_L^{(e)}$ is the probability of cluster wrapping along either the horizontal or vertical directions, or both; $R_L^{(1)}$ around one specified axis but not the other axis; $R_L^{(b)}$ around both the horizontal and vertical directions; $R_L^{(h)}$ and $R_L^{(v)}$ around the horizontal and vertical directions, respectively. For the square lattice $R_L^{(h)} = R_L^{(v)}$. The four wrapping probabilities satisfy the equations

$$R_L^{(b)} = R_L^{(e)} - 2R_L^{(1)}, \tag{2}$$

$$R_L^{(h)} = R_L^{(e)} - R_L^{(1)}, \tag{3}$$

from which the values of $R_L^{(b)}$ and $R_L^{(h)}$ can be obtained by measuring only the values of $R_L^{(e)}$ and $R_L^{(1)}$, or vice versa. Given the exact value of $R_\infty(p_c)$, the solution $p$ of the equation

$$R_L(p) = R_\infty(p_c) \tag{4}$$

gives a very good estimator for $p_c$. The solution $p$ converges to $p_c$ according to

$$p - p_c \sim L^{-11/4}. \tag{5}$$

In this way the corresponding values of $p_c$ from $R_\infty^{(e)}(p_c)$, $R_\infty^{(b)}(p_c)$, and $R_\infty^{(h)}(p_c)$ can be calculated, respectively. Since the wrapping probability $R_L^{(1)}(p)$ is nonmonotonic, the position of its maximum can be used to estimate $p_c$, instead of the value of $R_\infty^{(1)}(p_c)$.

The CPU time $T_L$ is related to the statistical errors $\sigma_{p_c}$ according to $\sigma_{p_c} \sim T_L^{-1/2} L^{1/4}$. For finite $L$ and a specific algorithm, $T_L$ depends only on $n$, the number of runs of the algorithm. To keep the same statistical error on systems of a different size, proper values of $n$ are taken to fulfill $T_L \sim \sqrt{L}$.

The results explained above are from Ref. [20] and will be followed in this Brief Report.

## II. THE ALGORITHM

Since our criterion for wrapping percolation originates from the difference in wrapping cluster and spanning cluster, the central task of the method is how to handle those clusters touching the boundaries. Each run of the algorithm starts from a lattice with periodic boundary conditions. In the following, we take the site wrapping percolation along the $x$ direction as an example. The whole algorithm can be separated into two parts, the spanning process and the wrapping process. In the spanning process, we check the states of a newly occupied site and its (nearest) neighbor sites previously occupied. If one site and one of its neighbors happen to be on the left boundary and right boundary, respectively, we call them as a pair of occupied "quasineighbor" sites. Along the $x$ direction, if we shift the periodic boundary conditions to the open boundary conditions, two sites of any pair of quasineighbor sites do not neighbor each other. In Fig. 1(a), there are six pairs of quasineighbor sites on the square lattice $6 \times 6$: (0,5), (6,11), (12,17), (18,23), (24,29), and (30,35). Two occupied quasineighbor sites belong to their respective clusters. Therefore, pairs of occupied quasineighbor sites correspond to pairs of their respective clusters. In the spanning process no implementations are required to these pairs of clusters or these occupied quasineighbor sites. For non-quasineighbor sites, if the two sites point to the same root site (belong to the same cluster), we need do nothing; otherwise, if the two sites belong to different clusters, we must merge them into a single cluster. We do exactly what we usually do [23] before the appearance of a spanning cluster[1] on the square lattice. In other words, the periodic boundary conditions along the $x$ direction are suppressed temporarily in the spanning process.

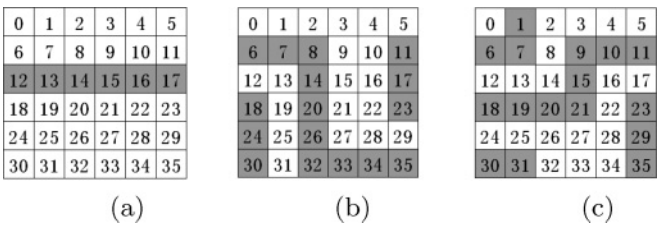



FIG. 1. Examples of wrapping clusters with occupied sites shaded. Pairs of clusters (excluding the spanning cluster) are merged in the first step of scanning the occupied quasineighbor sites [e.g., (18, 23) in (b) and (30, 35) in (c)]. In (a), the occupied quasineighbor sites (12, 17) belong to the same cluster. In (b) or (c) two ends of the spanning cluster are connected via a merged cluster.

Once a spanning cluster occurs, we turn to the wrapping process immediately, which aims to find a wrapping cluster. Obviously, this process is the core of the present algorithm. A spanning cluster has at least two ends (left end and right end), which touch the left boundary and the right boundary, respectively. When the periodic boundary conditions along the $x$ direction are recovered, across the boundaries, the two ends of the spanning cluster could be connected, which implies the appearance of a wrapping cluster. This process can be finished in three steps.

In the first step, we check the pairs of clusters touching the boundaries by scanning the occupied quasineighbor sites row by row. If two clusters of a pair of occupied quasineighbor sites, respectively, belong to different root sites and neither of them is the spanning cluster, we merge them into a single cluster. The root site of any of them can be chosen as the root site of the merged cluster. After the implementations on all the nonspanning clusters touch the boundaries, what we do next is to build relations between the spanning cluster and the (merged) nonspanning clusters touching the boundaries. We may meet three interesting cases when we scan pairs of clusters touching the boundaries once more. The simplest case is that the two clusters are the same spanning cluster [see Fig. 1(a) as an example]. We simply add a pointer from the spanning cluster to itself. The second case is that a nonspanning cluster does not possess a pointer pointing to the spanning cluster, we could add a pointer from the former to the latter, and label the boundary touched by the nonspanning cluster. For the third case, a nonspanning cluster already has a pointer pointing to the spanning cluster and previously touches the opposite boundary, we add a pointer from the spanning cluster to the nonspanning cluster. Two examples for the third case are shown in Figs. 1(b) and 1(c), respectively. For other cases we need do nothing further. In the final step, a simple statement is used to check whether the spanning cluster points to itself or not by following the pointers added in the second step.

Thus the wrapping process of our algorithm can be summarized as follows.

(1) Amalgamate pairs of nonspanning touching-boundary clusters.

(2) Add pointers from the amalgamated nonspanning touching-boundary clusters to the spanning cluster, or in turn, add a pointer from the spanning cluster to a nonspanning touching-boundary cluster.

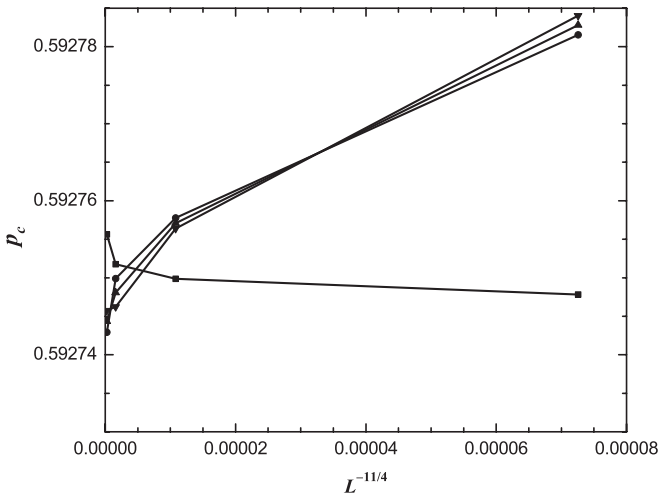
(3) By following a succession of pointers added above, check if we can get from the spanning cluster to itself.

[1]For the lattice with open boundary conditions along the $x$ direction, but with periodic boundary conditions along the $y$ direction, it is possible there are two or more spanning clusters along the $x$ direction. In the text, the spanning cluster always indicates that one is right visited; all other touching-boundary clusters are grouped into nonspanning clusters whether they are spanning or not.

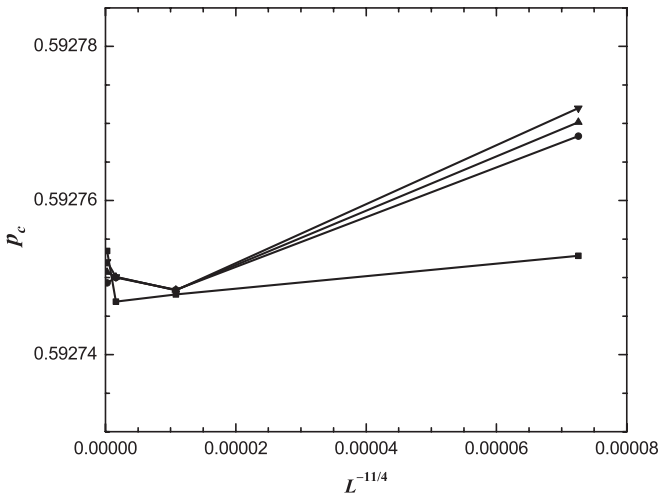
TABLE I. The values of $p_c$ for infinite lattice. Here, P and M represent the present method and Machta's method, respectively; $p_c^{(1)}$, $p_c^{(b)}$, $p_c^{(h)}$, and $p_c^{(e)}$ correspond, respectively, to the wrapping probabilities $R_L^{(1)}$, $R_L^{(b)}$, $R_L^{(h)}$, and $R_L^{(e)}$.

| | $p_c^{(1)}$ | $p_c^{(b)}$ | $p_c^{(h)}$ | $p_c^{(e)}$ |
|---|---|---|---|---|
| P | 0.592 752 8(16) | 0.592 748 0(31) | 0.592 747 5(23) | 0.592 747 1(19) |
| M | 0.592 749 3(23) | 0.592 748 2(14) | 0.592 748 7(18) | 0.592 749 1(22) |

If no wrapping cluster appears, we occupy one additional site on the previous lattice and repeat the spanning process and the wrapping process. Since we aim to check the reliability of our algorithm via the calculation of the percolation threshold only, we halt the algorithm once the percolation along both the $x$ and $y$ directions are detected. The program for our algorithm written in C is available online [24].




(a)




(b)

FIG. 2. The values of $p_c$ on square lattices of $L \times L$ obtained from the probabilities of cluster wrapping along one axis but not the other (solid squares), both axes (solid circles), one axis (solid upward-pointing triangles), and either axis (solid downward-pointing triangles). (a) Our method. (b) Machta's method.

## III. RESULTS

All computations are implemented on a desktop PC with a CPU clock speed 2.6 GHz and memory 1.96 GB. We fix $n$ runs of the algorithm to $2 \times 10^6$ for the lattice with $L = 256$. The computation takes the CPU time of about 37 hours. The other values of $n$ for $L$ equal to 128, 64, and 32 are, respectively, chosen to ensure $T_L \sim \sqrt{L}$. The total $n$ is about $2.7 \times 10^8$. The finite size scaling of $p_c$ for square lattice $L \times L$ is shown in Fig. 2(a), and the values of $p_c$ obtained from four different probabilities are listed in Table I.

To check the reliability of our method, a similar computation within the framework of Machta's method is implemented. For $L = 256$, $n$ is also taken as $2 \times 10^6$, CPU time is about 4.5 hours, which is only one-eighth of that in our method. In other words, the computation time in units of hours is $T_M = 0.28\sqrt{L}$ for Machta's method, while for our method it is $T_P = 2.3\sqrt{L}$. The other values of $n$ corresponding to $L$ are chosen by reference to this short CPU time, and the total $n$ in Machta's method is about $8 \times 10^7$. In comparison with the simulation of more than $7 \times 10^9$ separate samples in the work of Newman and Ziff, the statistical errors in this work are larger. The finite size scaling and the results of $p_c$ are, respectively, shown in Fig. 2(b) and Table I.

Besides the difference in computation time, there is little difference in the results. Obviously, from Table I, one can see that the data of $p_c^{(1)}$ and $p_c^{(e)}$ are better (smaller errors) than the data of $p_c^{(b)}$ and $p_c^{(h)}$ in our method, while in Machta's method, on the contrary, the data of $p_c^{(b)}$ and $p_c^{(h)}$ are better. This difference possibly arises from the larger values of $p_c^{(e)}$ (and therefore larger values of $p_c^{(b)}$ and $p_c^{(h)}$) at small $L$ in our method, which can be seen in Fig. 2. With the exception of this difference, our results coincide with that from Machta's method quite well.

## IV. CONCLUSION

A wrapping cluster appears if two ends (touching two opposite boundaries, respectively) of a spanning cluster are connected to each other across the boundaries. The results of percolation threshold in our method are as good as those from Machta's method, and are consistent with the published estimates of the square site percolation threshold [25–27]. The better result of $p_c^{(1)}$ obtained with our method implies that it might be appropriate to choose the present method for the further study of the spatially correlated percolation model [28], where the percolation thresholds have not yet been well determined.

The lengthy computation time in our method mainly comes from two aspects of our specific algorithm. In the spanning process, the clusters are amalgamated by suppressing the

periodic boundary conditions along the $x$ and $y$ directions, respectively; after a spanning cluster appears (either along the $x$ or $y$ direction), a wrapping process for testing a wrapping cluster is implemented. While in Machta's method, a wrapping cluster could be tested in the process of merging clusters on a square lattice with full periodic boundary conditions.

Although the values of percolation threshold could be obtained from spanning percolation on a lattice with open boundary conditions or from wrapping percolation on a lattice with periodic boundary conditions, different boundary effects are definitely covered in the results. With our method it is possible to give a direct computation of different boundary effects.

The present algorithm differs from the Newman-Ziff algorithm only in the criterion for wrapping percolation. Without question, the former can be extended to the calculation of $p_c$ for three-dimensional percolation on a cubic lattice in the same way as that of the latter.

## ACKNOWLEDGMENTS

The author would like to thank Professor Stephan Haas for useful comments. The recommendations and criticisms of a referee, which helped to improve this report, are highly appreciated.

[1] D. Stauffer and A. Aharony, *Introduction to Percolation Theory*, 2nd ed. (Taylor & Francis, London, 1992).
[2] P. J. Flory, J. Am. Chem. Soc. **63**, 3083 (1941).
[3] S.-W. Son, G. Bizhani, C. Christensen, P. Grassberger, and M. Paczuski, Europhys. Lett. **97**, 16006 (2012).
[4] G. Bizhani, P. Grassberger, and M. Paczuski, Phys. Rev. E **84**, 066111 (2011).
[5] E. Agliari, C. Cioli, and E. Guadagnini, Phys. Rev. E **84**, 031120 (2011).
[6] S. V. Buldyrev, R. Parshani, G. Paul, H. E. Stanley, and S. Havlin, Nature (London) **464**, 1025 (2010).
[7] Y. V. Kartashov, V. A. Vysloukh, and L. Torner, Opt. Express **15**, 12409 (2007).
[8] J. A. Hoyos and T. Vojta, Phys. Rev. B **74**, 140401 (2006).
[9] M. Ortuño, A. M. Somoza, V. V. Mkhitaryan, and M. E. Raikh, Phys. Rev. B **84**, 165314 (2011).
[10] G. Lois, J. Blawzdziewicz, and C. S. O'Hern, Phys. Rev. Lett. **102**, 015702 (2009).
[11] A. Ofir, S. Dor, L. Grinis, A. Zaban, T. Dittrich, and J. Bisquert, J. Chem. Phys. **128**, 064703 (2008).
[12] C. Lu and Y.-W. Mai, J. Mater. Sci. **43**, 6012 (2008).
[13] D. R. Stevens, L. N. Downen, and L. I. Clarke, Phys. Rev. B **78**, 235425 (2008).
[14] N. A. M. Aráujo, Jose S. Andrade, R. M. Ziff, and H. J. Herrmann, Phys. Rev. Lett. **106**, 095703 (2011).
[15] R. A. da Costa, S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes, Phys. Rev. Lett. **105**, 255701 (2010).
[16] N. A. M. Araújo and H. J. Herrmann, Phys. Rev. Lett. **105**, 035701 (2010).
[17] F. Radicchi and S. Fortunato, Phys. Rev. E **81**, 036110 (2010).
[18] M. Sahimi, *Applications of Percolation Theory* (Taylor & Francis, Bristol, MA, 1994).
[19] A. G. Hunt and R. Ewing, *Percolation Theory for Flow in Porous Media*, Lecture Notes in Physics 771, 2nd ed. (Springer, Berlin, 2009).
[20] M. E. J. Newman and R. M. Ziff, Phys. Rev. E **64**, 016706 (2001).
[21] G. Pruessner and N. R. Moloney, J. Phys. A **36**, 11213 (2003).
[22] J. Machta, Y. S. Choi, A. Lucke, T. Schweizer, and L. M. Chayes, Phys. Rev. E **54**, 1332 (1996).
[23] H. Gould, J. Tobochnik, and W. Christian, *An Introduction to Computer Simulation Methods*, 3rd ed. (Addison-Wesley, Reading, MA, 2006), p. 468.
[24] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevE.85.042106 for the program of our wrapping percolation algorithm.
[25] Y. Deng and H. W. J. Blöte, Phys. Rev. E **72**, 016126 (2005).
[26] X. Feng, Y. Deng, and H. W. J. Blöte, Phys. Rev. E **78**, 031136 (2008).
[27] M. J. Lee, Phys. Rev. E **78**, 031131 (2008).
[28] H. Yang, W. Zhang, N. Bray-Ali, and S. Haas, e-print arXiv:0908.0104.