On a Form of Coordinate Percolation

Elizabeth R. Moseman[*] and Peter Winkler[†]

July 31, 2008

Abstract

Let a_i, b_i, $i = 0, 1, 2, \ldots$ be drawn uniformly and independently from the unit interval, and let t be a fixed real number. Let a site $(i, j) \in \mathbb{N}^2$ be *open* if $a_i + b_j \leq t$, and *closed* otherwise. We obtain a simple, exact expression for the probability $\Theta(t)$ that there is an infinite path (oriented or not) of open sites, containing the origin. $\Theta(t)$ is continuous and has continuous first derivative except at the critical point ($t = 1$), near which it has critical exponent $(3 - \sqrt{5})/2$.

1 Introduction

Independent percolation, in which vertices ("sites") or edges ("bonds") of a graph are destroyed randomly and independently, was introduced in the 1950s by Broadbent and Hammersley [4] as a model of porous material. Since then, it has been the object of much study (see, e.g., the books by Grimmett [9] and by Bollobás and Riordan [2]). Many important questions remain open, among them the following: What is the probability that there is an open path to infinity from the origin, when sites of the plane grid are open independently with probability p? In particular, how does this quantity behave when it first becomes positive?

In coordinate percolation, a newer and less studied variation, the life or death of a site (or bond) is determined by independent events associated with the site's coordinates. Typically, coordinate percolation arises from *scheduling* problems, in contrast to the physical motivation for independent percolation.

In the "collision" form of coordinate percolation, a uniform, independent random integer between 1 and k is assigned to each integer point on the x- and y-axes, and sites are closed if they inherit the same value from both coordinates. The resulting configuration looks quite different from what one would get from independent closures; this fact was used by Diaconis and

[*]Department of Mathematical Sciences, USMA, West Point NY 10996, USA; lizz.moseman@usma.edu.

[†]Department of Mathematics, Dartmouth College, Hanover NH 03755-3551, USA; peter.winkler@dartmouth.edu. Research supported by NSF grant DMS-0600876.

Freedman [6] in providing a counterexample to the "Julesz Conjecture," which asked whether the human eye can distinguish random textures that agree in their first- and second-order statistics.

This form arose again in a conjecture of Winkler reported in [5], to the effect that with positive probability one can "schedule" two random walks on the alphabet $\{1, \ldots, k\}$ in such a way that they never collide. Independent work in [1] and [12] established that *unoriented* percolation takes place with positive probability iff $k \geq 4$; in [7] Peter Gács showed that proving the same for oriented percolation, which is what the conjecture called for, is in a sense fundamentally difficult.

Various types of coordinate *bond* percolation were studied by Gács [8] and very recently by Gábor Pete [11], the latter in the form of "corner percolation."

In the form of coordinate percolation considered here, random reals a_i, b_i, $i = 0, 1, 2, \ldots$, independently drawn from some continuous distibution, are assigned to integer points on the x- and y-axes, in the non-negative quadrant of the plane grid. A threshold $t \in \mathcal{R}$ is chosen, and a site $(i, j) \in \mathbb{N}^2$ is declared to be open if $a_i + b_j \leq t$. We wish to determine the probability $\Theta(t)$ that there is an infinite open path containing the origin.

The problem of finding an open path can again be viewed as a scheduling issue. The two sequences $\{a_i\}$, $\{b_i\}$ may be thought of as the rate at which some resource is used at successive stages of two processes; the processes must then be scheduled in parallel so that at no time do they use this resource at a greater-than-permissible rate. Suppose, for example, that two sequences of programs must be set up to run in parallel on a computer. Each program has some random (but known) RAM requirement; the program sequences must be scheduled so that at no time does the sum of the RAM requirements for two simultaneous programs exceed the computer's capacity.

A more pressing motivation, and indeed the primary one for the authors, was to capture a Θ-function and so get a precise look at a phase transition. One reason for believing that the above form of percolation might be more tractable than independent percolation stems from the property that in the former, the oriented and unoriented cases are indistinguishable—that is, if there is an infinite open path from the origin then there is one that moves only north and east. This follows from a much more general theorem of Brightwell and Winkler [3], but is also an easy consequence of the correctness of the "percolation algorithm" presented below.

Moreover, the particular choice of distribution from which the a_i and b_i will be drawn permits a combinatorial analysis, and indeed it is this which ultimately results in an exact expression for Θ. This result and more can be found also in the first author's PhD thesis [10].

2 An Asymmetric Percolation Algorithm

Finding an open path from the origin to infinity—that is, "percolating" in a given configuration—is in some sense (not made precise here) algorithmically much easier in our setting than in the independent case. Essentially, it is a matter of finding "good" lines, that is, columns $x = i$ with low a_i or rows $y = j$ with low b_j, then looking along these lines for even lower values.

The process is begun by checking that the origin is open, then setting $i = i' = j = 0$. Suppose at some stage a token has found its way along an open, oriented path to a site (i, j) with the following properties:

1. Values $a_0, \ldots, a_{i'}$ have been inspected, where i' is some index with $i' \geq i$, and a_i is the least of these values;

2. Values $b_0, \ldots, b_j$ have been inspected, and b_j is the least of those.

In general, we say in this situation that we are "in state A" and we proceed as follows. Values $a_{i'+1}$, $a_{i'+2}$, etc. are inspected until some $a_{i''}$ is found that is either less than a_i or greater than $t - b_j$. In the former case, we move the token to (i'', j) and repeat, with i and i' both replaced by i''.

In the latter case the site (i'', j) is closed so we must repeatedly increment j, inspecting successive values of b_j with the idea of finding a row which will enable us to get past the bad column $x = i''$. We call this procedure "state C". If a value $b_{j'} \leq t - a_{i''}$, is found, we move the token to (i, j'), replace i' by i'' and return to state A (with j now set to j').

If we first hit a closed site, i.e. a value $b_{j'} > t - a_i$, we move to state D (for "dead") and terminate the algorithm. In that case we cannot percolate, since all the sites on the north and east borders of the rectangle with corners at $(0, 0), (i'', 0), (i'', j'), (0, j')$ are closed. We thus have

Lemma 2.1. *Either there is an infinite, open, oriented path from the origin or the above algorithm terminates in state D, in which case there is no infinite open path, oriented or otherwise, containing the origin.*

The algorithm is diagrammed in Figure 1. Its inherent asymmetry with respect to the axes, which makes it tend to spend more time in state A than in state C and terminate only from the latter, may at first seem mysterious. Why not just go from state A to a state B which mimics A in the y-direction? The difficulty is that we have already inspected some columns to the east and for purposes of later analysis, we want always to be looking at fresh, independent a_i and b_j.

The percolation model and algorithm both generalize naturally to higher dimension.

3 Basics of the [0,1] Case

Henceforth we shall concentrate on the case where the distribution from which the a_i and b_i are drawn is uniform on the unit interval. This provides some attractive additional features, the most important of which is that it enables a combinatorial analysis of the percolation algorithm. The combinatorial analysis in turn stems from a connection to the "worm order" on sequences, which was introduced in [3] and will be defined later.

The main result is

Theorem 3.1. *Let a_i, b_i, $i = 0, 1, 2, \ldots$ be drawn uniformly and independently from the unit interval, with $(i, j) \in \mathbb{N}^2$ declared open iff $a_i + b_j \leq t$. Then the probability $\Theta(t)$ that there is an infinite open path containing the origin is given by*

$$\Theta(t) = \begin{cases} 0 & \textit{if } t < 1 \\ \frac{5+3\sqrt{5}}{10}\big(t-1\big)^{\frac{3-\sqrt{5}}{2}} - \frac{3\sqrt{5}-5}{10}\big(t-1\big)^{\frac{3+\sqrt{5}}{2}} & \textit{if } 1 \leq t \leq 2 \\ 1 & \textit{if } t > 2. \end{cases}$$

Corollary 3.2. *The critical exponent*

$$\beta := \lim_{t \downarrow 1} \frac{\log \Theta(t)}{\log(t-1)}$$

is given by

$$\beta = \frac{3 - \sqrt{5}}{2} \ .$$

Proof of Theorem. Let us begin with some elementary observations. If the threshold t exceeds 1, then there are entire open lines where $a_i \leq t-1$ or $b_j \leq t-1$; since the x- or y-axis may be among these it is clear that we percolate with positive probability.

On the other hand, if $t < 1$ there are closed lines where $a_i > 1-t$ or $b_j > 1-t$, and one from each will box off the origin, so $\Theta(t) = 0$ for $t < 1$. Thus the critical point is at $t = 1$.

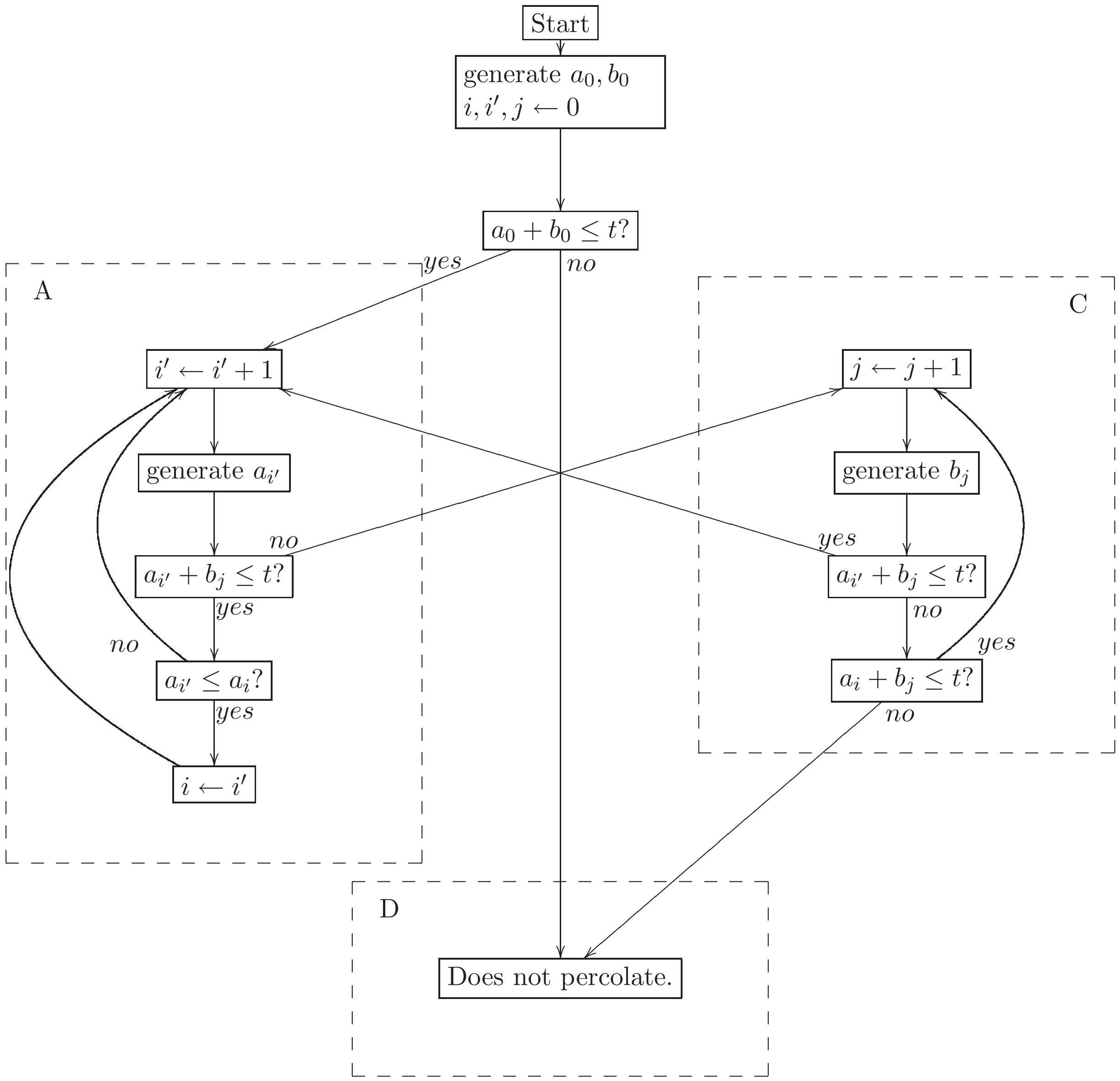

Figure 1: A percolation algorithm.

In running our percolation algorithm in the $t > 1$ case, our objective is to survive until we hit an open line, which will occur when we inspect our (n+1)st value for some random geometrically-distributed n. At the moment when we do find an a_i or b_j below $t-1$, the corresponding line has been reached and percolation achieved. Thus,

$$\Theta(t) = \sum_{n=0}^{\infty}(2-t)^n(t-1)Q_n(t)$$

where $Q_n(t)$ is the probability that the algorithm escapes death for n steps, given that all the values it inspects lie between $t-1$ and 1. Note that for all t, $Q_0(t) = 1$ (vacuously), $Q_1(t) = 1$, and $Q_2(t) = \frac{1}{2}$, reflecting the $\frac{1}{2}$ probability that the origin is open given that neither the line $x = 0$ nor the line $y = 0$ is completely open. In fact, we will see that $Q_n(t)$ never depends on t.

First we must convert the openness criterion to one involving order, instead of summation. Fix $t > 0$, and for each $j \geq 0$, let

$$c_j = \begin{cases} t - b_j & \text{if } b_j > t-1, \text{ and} \\ b_j & \text{otherwise.} \end{cases}$$

Then the c_j are independent and uniform in $[0, 1]$, just as the b_j were, and the site (i, j) is open exactly when either $c_j \leq t-1$, or $a_i \leq c_j$.

Suppose $a_0, \ldots, a_i$ and $c_0, \ldots, c_j$ are all greater than $t-1$. Then all are independent and uniforrm in $(t-1, 1]$ and the configuration of open and closed sites in $\{0, 1, \ldots, i\} \times \{0, 1, \ldots, j\}$ is determined only by the *order* in which these $i+j+2$ values fall as real numbers. Since this order is (with probability 1) strict and uniformly random, $Q_n = Q_n(t)$ is a combinatorial quantity (indeed, an integer multiple of $1/n!$) which does not depend on t.

Our objective now is to calculate Q_n, using a combinatorial version of the algorithm of Figure 1, and ultimately obtain an expression for $\Theta(t)$. In determining Q_n we can and will assume that $t = 1$, so that (i, j) is open iff $a_i \leq c_j$.

If the site (i, j) is reachable by oriented open path from the origin, the sequence $a_0, \ldots, a_i$ is said to precede $c_0, \ldots, c_j$ in the "worm order", which is easily seen to be transitive and reflexive. If two sequences are defined to be equivalent when each precedes the other in this pre-order, then each equivalence class will contain a unique shortest word, called a "worm"; see [3, 10] for more information about worms.

The computation of Q_n amounts to determining the probability that one random sequence precedes another in the worm order. For the infinite random sequences under consideration here, the worm is an alternating subsequence of new "records", and it is these which our percolation algorithm compares. However, we will run the algorithm without observing any actual values of the a_i and c_j, but only their order as real numbers.

Thus, the algorithm now proceeds as follows. We begin with an order consisting only of the single point a_0. At the kth step we examine the kth new value, thus it is in the first two steps that we verify that $a_0 < c_0$ and conclude that the origin is open. In any state, prior to the kth step the information on hand consists only of indices i' and j (with $i' + j + 2 = k - 1$) and an ordering $\sigma = (\sigma_1 < \sigma_2 < \cdots < \sigma_{k-1})$ of the $k-1$ values $\{a_0, \ldots, a_{i'}\} \cup \{c_0, \ldots, c_j\}$. In states A and C, this ordering must have $a_i = \sigma_1$ for some index $i \leq i'$ and $c_j = \sigma_{k-1}$, representing the fact that $x = i$ is the best (lowest-valued) column and $y = j$ the best (highest-valued) row seen so far.

In state A, the next value to be considered, namely a_{i+1}, is now equally likely to be found in any of the k slots in the order. Successive a's are inserted into σ until we hit an $a_{i''}$ at the top of the order. When this value is found, we move to state C.

In state C, we insert $c_{j+1}, c_{j+2}, \ldots$ into the order σ, looking for a $c_{j'}$ to appear on bottom or top. The former puts us in state D, the latter back into state A. See Figure 2 for a flowchart of this now-discrete algorithm.

The discrete algorithm spawns a simple time-dependent Markov chain. At time $k > 2$, the transition probabilities among the states A, C, and D (taken in that order) are given by the matrix

$$M_k = \begin{bmatrix} 1 - 1/k & 1/k & 0 \\ 1/k & 1 - 2/k & 1/k \\ 0 & 0 & 1 \end{bmatrix}.$$

Let $\vec{p_k}$ be the state probability distribution after stage k, so that $\vec{p_k} = \vec{p_{k-1}} M_k$. We may consider that the process begins at time $k = 1$ in state A—that is, $\vec{p_0} = (1, 0, 0)$. Then $\vec{p_1} = \vec{p_0} M_1 = (0, 1, 0)$ and $\vec{p_2} = \vec{p_1} M_2 = (\frac{1}{2}, 0, \frac{1}{2})$, reflecting the $\frac{1}{2}$ probability that the origin is open (so $Q_2 = \frac{1}{2}$).

The eigenvalues for M_k are $\lambda_k = 1 + \frac{1}{k}(\varphi - 2)$, $\bar{\lambda}_k = 1 + \frac{1}{k}(\bar{\varphi} - 2)$, and 1, where $\varphi = \frac{1+\sqrt{5}}{2}$ is the golden ratio and $\bar{\varphi} = 1 - \varphi$. Conveniently, the corresponding eigenvectors do not depend on k; they are, respectively, $\vec{u} = (\varphi, 1, \frac{1}{\varphi - 2})$, $\vec{v} = (\bar{\varphi}, 1, \frac{1}{\bar{\varphi} - 2})$, and $\vec{w} = (0, 0, 1)$.

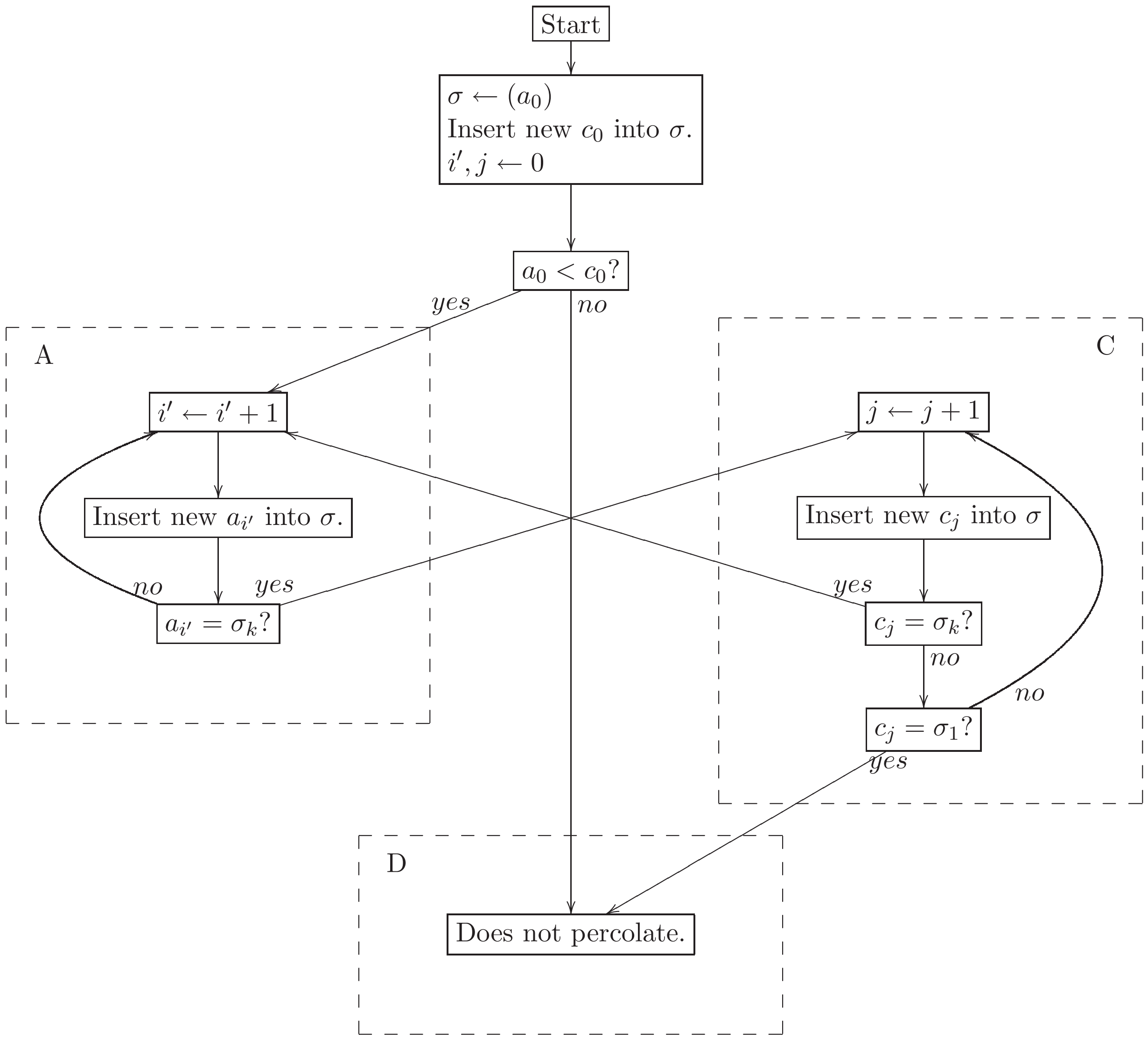

Figure 2: A discrete percolation algorithm.

Define

$$\begin{aligned}\Lambda_n &= \prod_{k=1}^{n} \lambda_k = (-1)^n \binom{\bar{\varphi}}{n} \\ \bar{\Lambda}_n &= \prod_{k=1}^{n} \bar{\lambda}_i = (-1)^n \binom{\varphi}{n}\end{aligned}$$

where $\binom{r}{n} = \prod_{k=1}^{n} \frac{r-k+1}{k}$ is the generalized binomial coefficient. Noting that

$$\vec{p}_0 = \frac{1}{\sqrt{5}}\vec{u} - \frac{1}{\sqrt{5}}\vec{v} + \vec{w}$$

we see that

$$\vec{p}_n = \frac{1}{\sqrt{5}}\Lambda_n\vec{u} - \frac{1}{\sqrt{5}}\bar{\Lambda}_n\vec{v} + \vec{w}$$

for all n, including $n = 0$.

The probability Q_n that we are not in state D at time n, thus in state A or C, is the sum of the first two coordinates of the vector $\vec{p}_k$. Therefore

$$Q_n = (\varphi{+}1)\frac{1}{\sqrt{5}}\Lambda_k - (\bar{\varphi}{+}1)\frac{1}{\sqrt{5}}\bar{\Lambda}_k$$

$$= \frac{(-1)^n}{\sqrt{5}}\binom{\bar{\varphi}}{n}(\varphi + 1) - \frac{(-1)^n}{\sqrt{5}}\binom{\varphi}{n}(\bar{\varphi} + 1)$$

for all $n \geq 0$.

Recalling our formula for the probability of percolation, we have

$$\Theta(t) = \sum_{n=0}^{\infty}(2 - t)^n(t{-}1)Q_n(t)$$

$$= \frac{t{-}1}{\sqrt{5}}\left((\varphi{+}1)\sum_{n=0}^{\infty}(-1)^n\binom{\bar{\varphi}}{n}(2{-}t)^n - (\bar{\varphi}{+}1)\sum_{n=0}^{\infty}(-1)^n\binom{\varphi}{n}(2{-}t)^n\right)$$

$$= \frac{t{-}1}{\sqrt{5}}\left((\varphi{+}1)\sum_{n=0}^{\infty}\binom{\bar{\varphi}}{n}(t{-}2)^n 1^{\bar{\varphi}-n} - (\bar{\varphi}{+}1)\sum_{n=0}^{\infty}\binom{\varphi}{n}(t{-}2)^n 1^{\varphi-n}\right)$$

$$= \frac{t{-}1}{\sqrt{5}}\left((\varphi{+}1)((t{-}2) + 1)^{\bar{\varphi}} - (\bar{\varphi}{+}1)((t{-}2) + 1)^{\varphi}\right)$$

(via the generalized binomial theorem)

$$= \frac{\varphi{+}1}{\sqrt{5}}(t{-}1)^{1+\bar{\varphi}} - \frac{\bar{\varphi}{+}1}{\sqrt{5}}(t{-}1)^{1+\varphi}$$

$$= \frac{5 + 3\sqrt{5}}{10}(t{-}1)^{\frac{3-\sqrt{5}}{2}} - \frac{3\sqrt{5} - 5}{10}(t{-}1)^{\frac{3+\sqrt{5}}{2}}$$

as claimed. Notice that this expression does take the value 0 at the critical point $t = 1$, rising there with the claimed critical exponent $(3 - \sqrt{5})/2 \sim 0.381966011$. At $t = 2$, the expression is equal to 1 with (left) derivative 0; of course $\Theta(t) = 1$ for all $t \geq 2$ since, in that case, all grid points are open. Thus Θ is continuous, has continuous first derivative except at $t = 1$, and is analytic except at $t = 1$ and $t = 2$. A plot of Θ, compliments of *Mathematica*$^{\text{TM}}$, is shown in Figure 3.

$\square$

4 Concluding Comments

The general shape, continuity, and first-derivative behavior of $\Theta(t)$ is consistent with what is known and suspected about the function $\Theta_{\text{ip}}(p)$, describing the probability of independent percolation (oriented or not) on the plane grid—except that in the latter case, the critical exponent is believed to be rational. There is of course no reason to draw any inferences regarding $\Theta_{\text{ip}}(p)$ from the results above; coordinate percolation is simply not the same process as independent percolation.

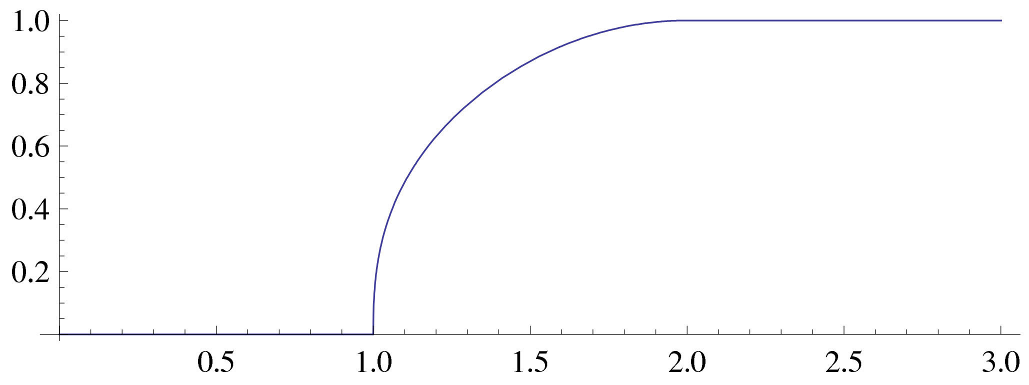

Figure 3: Probability of coordinate percolation for thresholds t between 0 and 3.

Coordinate percolation does have some independent interest, however, and we hope that the results here will inspire others to study other, and more general, versions. The analytic "percolation algorithm" described above generalizes to higher dimension, but its combinatorial cousin does not (as far as we know). Versions on other lattices even on the plane, such as the triangular lattice (in which sites will live or die based on values given to the *three* lines crossing there), may behave quite differently.

Acknowledgments

The authors have greatly benefited from conversations with Graham Brightwell (London School of Economics) and Dartmouth colleague Peter Doyle.

References

[1] P.N. Balister, B. Bollobás, and A.M. Stacey, Dependent percolation in two dimensions, *Probab. Theory & Related Fields* **117** #4 (2000), 495–513.

[2] B. Bollobás and O. Riordan, *Percolation*, Cambridge 2006.

[3] G.R. Brightwell and P. Winkler, Submodular Percolation, preprint (2007).

[4] S.R. Broadbent and J.M Hammersley, Percolation processes: I. Crystals and mazes, *Proc. Cambridge Philos. Soc.* **53** (1957), 629–641.

[5] D. Coppersmith, P. Tetali and P. Winkler, Collisions among random walks on a graph, *SIAM J. Discrete Math.* **6** #3 (1993), 363–374.

[6] P. Diaconis and D. Freedman, On the statistics of vision: the Julesz conjecture, *J. Math. Psych.* **24** #2 (1981), 112–138.

[7] P. Gács, The clairvoyant demon has a hard task, *Combin. Probab. Comput.* **9** #5 (2000), 421–424.

[8] P. Gács, Compatible sequences and a slow Winkler percolation, *Combin. Probab. Comput.* **13** #6 (2004), 815–856.

[9] G. Grimmett, *Percolation*, Second Edition, Springer 1999.

[10] E. Moseman, *The Combinatorics of Coordinate Percolation*, PhD thesis, Dartmouth College 2007.

[11] G. Pete, Corner percolation on Z2 and the square root of 17, preprint, [arXiv:math.PR/0507457v4], `http://arxiv.org/abs/math.PR/0507457`.

[12] P. Winkler, Dependent percolation and colliding random walks, *Random Structures & Algorithms* **16** #1 (2000), 58–84.