Nanoscale Digital Computation Through Percolation

Mustafa Altun, Marc D. Riedel, and Claudia Neuhauser

University of Minnesota, USA

E-mail: {altu0006, mriedel, neuha001} @umn.edu

ABSTRACT

In this study, we apply a novel synthesis technique for implementing robust digital computation in nanoscale lattices with random interconnects: *percolation theory on random graphs*. We exploit the non-linearity that occurs through percolation to produce Boolean functionality. We show that the error margins, defined in terms of the steepness of the non-linearity, translate into the degree of defect tolerance. We study the problem of mapping Boolean functions onto lattices with good error margins.

Categories and Subject Descriptors

B.7.1 **[integrated circuits]:** Types and Design Styles – *advanced technologies*.

General Terms: Design, Reliability.

Keywords: Percolation, Nanoscale Digital Computation, Logic Synthesis.

1. PERCOLATION THEORY

Percolation theory is a rich mathematical topic that forms the basis of explanations of physical phenomena such as diffusion and phase changes in materials. It tells us that in media with random local connectivity, there is a critical threshold for global connectivity: below the threshold, the probability of global connectivity quickly drops to zero; above it, the probability quickly rises to one [1]. This is illustrated in Figure 1.

Consider the lattice shown in Figure 2(a). Suppose that each square in the lattice is colored black with independent probability p_1. Let p_2 be the probability that a connected path exists between the top and bottom plates. Figure 2(b) shows the relationship between p_1 and p_2 for different square lattice sizes.
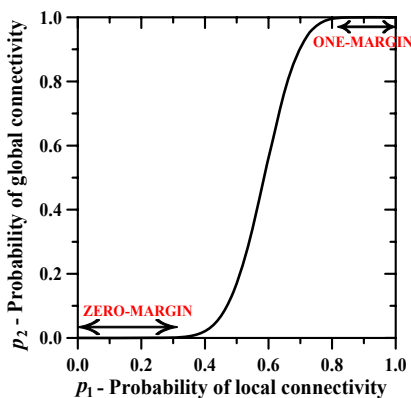

Figure 1. Non-linearity through percolation in random media.

Percolation theory tells us that with increasing lattice size, the curve steepness increases. (In the limit, an infinite lattice produces a perfect step function.) Here p_c is defined as a critical probability below which p_2 is approximately 0 and above which p_2 is approximately 1. We define the **one margin** and the **zero margin** as the corresponding ranges of p_1, i.e., values of p_1 that produce values of p_2 that we interpret as *logical* one and zero, respectively. We exploit the theory in a novel way: we use the nonlinearity produced by percolation to implement digital computation.
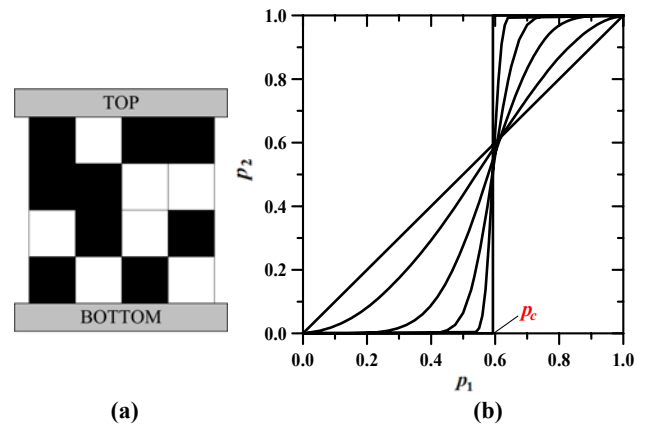

(a)	(b)

Figure 2. (a) Percolation lattice; (b) p_2 versus p_1 for 1×1, 2×2, 6×6, 24×24, 120×120 and infinite size lattices.

2. NANOWIRE CROSSBAR ARRAYS

Although not tied to any specific technology, we frame our discussion in terms of a conceptual model of nanowire arrays. Figure 3 illustrates a nanowire crossbar array with four plates: left, right, top, and bottom. Figure 4 illustrates connectivity in terms of squares: black squares represent crosspoints that are ON and white squares crosspoints that are OFF. Suppose that in this technology crosspoints between the horizontal and vertical nanowires are FET-like junctions [2]. When a high or low voltage is applied, these develop low or high impedances, respectively. Ideally, if the applied voltage is 0 (corresponding to logic zero), then all the crosspoints are OFF and so there is no connection between any of the plates. If the applied voltage is V_{DD} (corresponding to logic one), then all the crosspoints are ON and so the plates are connected. However, with defects in the lattice, not all crosspoints will respond this way [3]. This is illustrated in Figure 5.
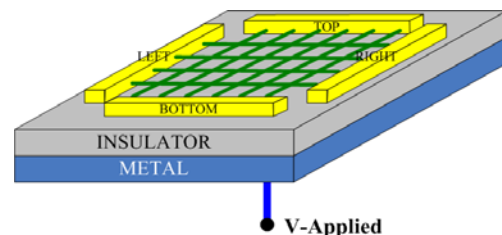

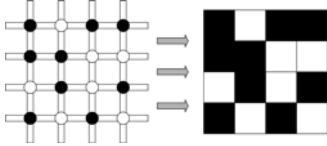**Figure 3. 3-D representation of a nanowire crossbar array.**

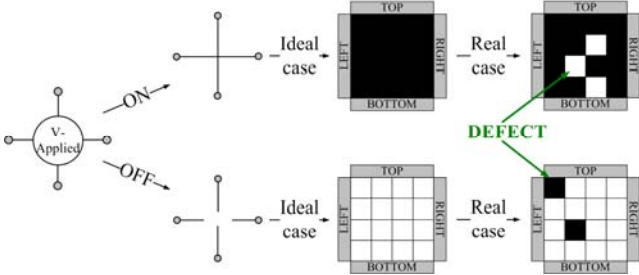**Figure 4. Nanowire crossbar array with random connections.**

Figure 5. Schematic of a nanowire crossbar based switch.

3. IMPLEMENTING BOOLEAN FUNCTIONS THROUGH PERCOLATION

Our approach for implementing Boolean functions is illustrated in Figure 6. The input values, $X_{11},..., X_{rc}$, are applied to distinct regions in the array. In each region, the probability of local connectivity p_1 is ideally only dependent on the corresponding Boolean input value. Connectivity through the entire array is now a Boolean function of the input values via the relationship between the input values and the probability of local connectivity. Call the Boolean functions that are implemented according to the top-to-bottom and left-to-right plate connectivities f and g, respectively. As shown in Figure 7, each Boolean function evaluates to one if there exists a path between corresponding plates, and evaluates to zero otherwise. In this way, digital computation is achieved without switches or logic elements; it occurs due to the random connectivity of the fabric.
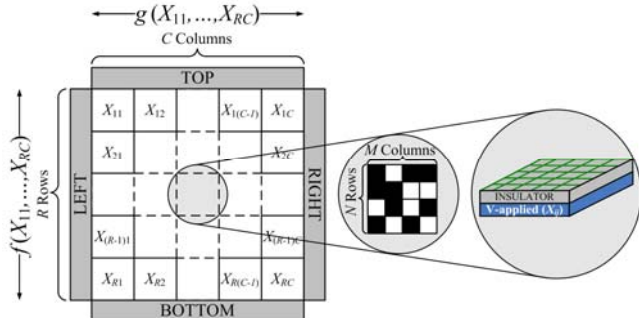

Figure 6. Boolean computation in a random network.

A path exists between top and bottom, $f = 1$
No path exists between left and right, $g = 0$

A path exists between top and bottom, $f = 1$
A path exists between left and right, $g = 1$

Figure 7. Relation between Boolean functionality and paths.

An important consideration in the synthesis methodology is the quality of the margins. Suppose that the zero and one margins are the range of values for p_1 for which p_2 is always below ε and above 1-ε, respectively, where ε is a very small number. We selected $\varepsilon = 0.001$ in this study. Accordingly, these margins correlate with the degree of **defect tolerance**. For instance a 10%

one margin means that even though we expect that one out of ten crosspoints will not operate correctly – because these are defective, or because of transient errors, or because of noise – the circuit still evaluates to one with high probability ($p_2 > 0.999$). The higher the margins, the higher the defect tolerance that we achieve.

Different assignments of input variables to the regions of the network affect the margins. A 4-input 2×2 lattice is analyzed in Figure 8. As can be seen, the row highlighted in grey has very low margins – indeed, these are nearly zero –so the circuit is likely to produce erroneous values for this input computation. Let's examine why.

X_{11}	X_{21}	X_{12}	X_{22}	f	Margin	g	Margin
0	0	0	0	0	40%	0	40%
0	0	0	1	0	25%	0	25%
0	0	1	1	1	14%	0	23%
0	1	0	1	0	23%	1	14%
0	1	1	0	0	0%	0	0%
0	1	1	1	1	14%	1	14%
1	1	1	1	1	25%	1	25%
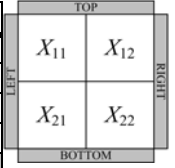
$f = X_{11}X_{21}+X_{12}X_{22}$

$g = X_{11}X_{12}+X_{21}X_{22}$

Figure 8. Four-input lattice and possible 0/1 assignments to the inputs (up to symmetries) and the corresponding margins.

Assignments that evaluate to zero but have diagonally adjacent assignments of blocks of one's result in poor zero margins because there is a good chance that a weak connection from top to bottom will form through stray, random connections across the diagonal. This is illustrated in Figure 9. Note that each 4-connected path corresponds to the AND of the inputs; the paths taken together correspond to the OR of these AND terms, so implement a sum-of-products expression. This interpretation gives rise to an interesting problem in logic optimization: given a Boolean sum-of-products expression as the target function, how should one assign the variables to the regions of the fabric such that there are no diagonally adjacent variables that are one in any assignment that evaluates to zero? This question takes us to the concept of **lattice duality**. A necessary and sufficient condition for good error margins is that the Boolean functions corresponding to the top-to-bottom and left-to-right plate connectivities f and g are dual functions.
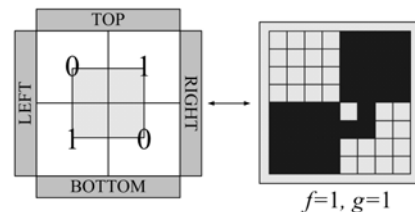

$f=1, g=1$

Figure 9. An input assignment with a low zero margin.

4. REFERENCES

[1] G. Grimmet, *Percolation*, Springer, New York, NY, 1989.

[2] A. DeHon, "Nanowire-based programmable architectures," *ACM Journal on Emerging Technologies in Computing Systems*, Vol. 1, No. 2, pp. 109–162, 2005.

[3] J. Huang, M. Tahoori, and F. Lombardi, "On the defect tolerance of nano-scale two-dimensional crossbars," *Proc. IEEE Int'l Symp. on Defect and Fault Tolerance of VLSI systems,* pp. 96 - 104, 2004.