A PROBABILISTIC-BASED DESIGN METHODOLOGY FOR NANOSCALE COMPUTATION

R. Iris Bahar, Joseph Mundy, and Jie Chen

Division of Engineering, Brown University, RI 20912, USA

ABSTRACT

As current silicon-based techniques fast approach their practical limits, the investigation of nanoscale electronics, devices and system architectures becomes a central research priority. It is expected that nanoarchitectures will confront devices and interconnections with high inherent defect rates, which motivates the search for new architectural paradigms.

In this paper, we propose a probabilistic-based design methodology for designing nanoscale computer architectures based on Markov Random Fields (MRF). The MRF can express arbitrary logic circuits and logic operation is achieved by maximizing the probability of state configurations in the logic network. Maximizing state probability is equivalent to minimizing a form of energy that depends on neighboring nodes in the network. Once we develop a library of elementary logic components, we can link them together to build desired architectures based on the *belief propagation* algorithm. Belief propagation is a way of organizing the *global* computation of marginal belief in terms of smaller *local* computations. We will illustrate the proposed design methodology with some elementary logic examples.

1. INTRODUCTION

As current silicon-based techniques fast approach their practical limits, developing nanoscale electronics, devices and system architectures becomes more important [13]. For instance, because of fundamental limitations at the nanoscale level, the past approach of using global interconnections and assuming error-free computing may no longer be possible, thereby presenting new design challenges to computer engineers. It is likely that nanoscale computing will be dominated by communication, where processing is based on redundant and adaptive pathways of error-prone connections. This framework will drive computer architecture in the direction of locally-connected, self-organizing hardware meshes that merge processing and memory [1]. In this section we provide a brief overview of various nanodevices currently being developed, their fault characteristics in the context of large networks of devices, and some proposed nanoarchitectures that take into account these characteristics in their design.

1.1. Nanodevices

A wide spectrum of nanoscale devices are being developed based on atomic and molecular phenomena. One promising structure is the carbon nanotube (CNT). Carbon nanotubes are excellent conductors and can provide wires for device interconnection. At the same time, CNT structures for diodes and field effect transistors have been demonstrated [20], [14]. CNTs can form a crosspoint

This work was supported in part by NSF grant number CCR-0304284.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ICCAD'03, November 11-13, 2003, San Jose, California, USA.
Copyright 2003 ACM 1-58113-762-1/03/0011 ...$5.00.

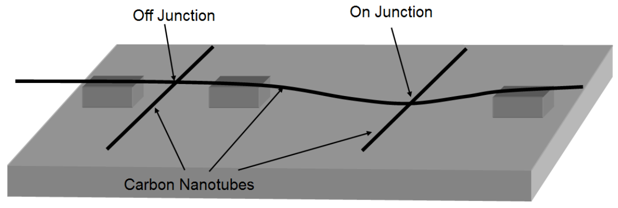


Figure 1: The principle of switching with carbon nanotubes. Tubes are joined by an attractive electric field. Molecular forces maintain the connection when the field is removed. (After Lieber [11]).

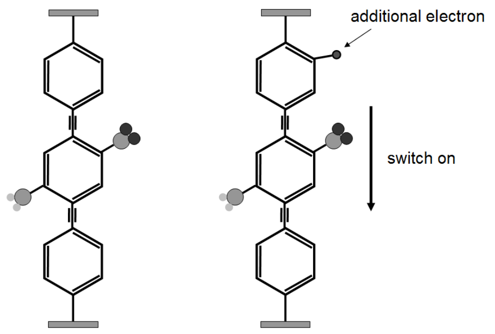


Figure 2: Switching based on altering the electronic state of a molecule. (After Lieber [11]).

switching structure as shown in Figure 1. In this device, an attractive electric field between the tubes causes the tube on the right to bend and come in contact with the tube underneath. Molecular forces between the tubes maintain the contact for an extended period of time, and thus provide a mechanism for memory [16]. A major issue is the physical placement of CNTs at the nano scale. One promising approach is to use the self-organizing properties of an alumina substrate to fabricate arrays of CNTs. Regular arrays of 50 nanometer diameter tubes on 150 nanometer centers have been demonstrated [19]. Another major nanodevice approach is the direct use of molecules and their electronic states. This concept is shown in Figure 2. The device conduction properties are achieved by changes in the physical configuration or electronic state of the molecule. Diode characteristics have been demonstrated [22] and it is possible to achieve stable memory at room temperature [15].

Quantum dot cellular automata (QCA) are based on the local interaction of quantum dots arranged in cells that enforce consistent mutual states. Logic operations can be achieved by encoding the function into spatial patterns of the cells. Information can also be propagated through chains of QCA devices in a shift register mechanism to provide interconnection between logic blocks. A major drawback to current QCA devices is that they require temperatures well below 1 degree Kelvin for successful operation.

Single electron devices have also been proposed and demonstrated. These devices operate on the principle of coulomb forces when small, isolated electrodes are occupied by an electron. The presence of the electron prevents tunneling of additional electrons and thus forms the basis for memory and switching [12]. At present these devices have to operate at sub-1 degree Kelvin temperatures. However, if electrode sizes on the order of 1*nm* can be achieved, then operation at room temperature becomes feasible.

Perhaps the most practical near-term devices are silicon nanowires which can be fabricated to widths on the order of 10*nm*. The wires exhibit semiconductor properties and so wires in contact can form PN junctions. In addition, crossing nanowires can operate as a FET device [9], making it feasible to fabricate classic silicon circuits.

1.2. Anticipated characteristics

To date, the fabrication of nanocircuits has been limited to a few devices intended to demonstrate simple logic or memory operations. There are no actual data to measure the characteristics of large networks of devices. However it is possible to pose two likely characteristics that will have to be confronted in the development of computational architectures that use these devices.

1. A high and dynamic failure process: It can be expected that a significant fraction of the devices and their interconnections will fail. These failures will occur both during fabrication and at a steady rate after fabrication, thus precluding a single test and repair strategy.
2. Operation near the thermal limit: As device sizes shrink, the energy difference between logic states will approach the thermal limit. Thus, the very nature of computation will have to be probabilistic in nature, reflecting the uncertainty inherent in thermodynamics.

The first characteristic is a simple extrapolation of current device fabrication experience. The smaller the device dimensions become, the more phenomena can interfere with correct operation. It seems likely that architectures will have to cope with device and connection failure rates of 10% or more. At the same time, the nature of connections and devices will be based on mechanisms that can easily mutate over time, such as chemical reactions or fusing and bridging of connections.

The second conclusion can be arrived at in several ways. Perhaps the most direct is to consider the evolution of current CMOS technology towards smaller and smaller device sizes, perhaps using silicon nanowire devices. Current processor chips have about 100 million transistors and dissipate over 100 Watts. It can be assumed that this power level is already near the practical limit in terms of battery life and dangerous external temperatures. The natural evolutionary forces that drive the number of gates per chip and clock rate upward will decrease logic transition energy limits to within a few orders of magnitude of k_bT. [1]

Another approach to the same conclusion is the study of the ultimate limit on the energy required for computation, starting with the paradox of Maxwell's demon [2] and ultimately clarified by the development of information theory. It can be shown that the energy cost of computation cannot be reduced below $(\ln 2)k_bT$ per bit. This basic result is derived from the necessary increase in randomness as information is lost during computation. For example, the input to an AND gate has four possible states while the output has only two. The evolution of device technology will relentlessly drive towards this limit, requiring approaches that can confront randomness in logic state.

[1] k_b is Boltzmann's constant, $1.38 \times 10^{-23} J/^\circ K$, and T is absolute temperature in $^\circ K$. At room temperature, $k_bT = 0.026$ electron-volts.

1.3. Architectural approaches

It will be necessary to evolve new architectural concepts in order to cope with the high level of device failure and randomness of logic states anticipated by the reasoning just proposed. Current architectural studies aimed at nanoscale systems are focused primarily on the first issue, device failure.

There are two basic approaches being proposed to deal with significant device failure rates: testing and routing around failures; and designing with redundant logic in the form of error correction. Illustrative of the first approach are the architectures of DeHon [5] and Goldstein and Budiu [7]. They propose designing in extra circuit elements that can be used to supplement failed devices and connections. A major issue is the testability of the network and the ability to confront continuous failures over the life of the device.

The second approach was suggested in the pioneering work by Von Neumann [18] where he used majority logic gates as a primitive building block and randomizing networks to prevent clusters of failures from overwhelming the fault tolerance of the majority logic. The architecture proposed in [8] is based on this approach.

One architectural approach that can provide continuous adaptation to errors is based on neural network structures [6]. The synaptic weights of the neural network are implemented using multiple connection paths and the summation is provided by conventional CMOS differential amplifier nodes. The connections are adaptively configured using single electron switching devices. It is proposed that useful computation could arise by training the network with a series of required input-output pairs.

The approach taken in this paper has some similarity to the architecture proposed in [6], but is based on the Markov Random Field (MRF). The MRF provides a formal probabilistic framework so that computation can be directly embedded in a network with immunity to both device and connection failures. Since logic states are computed probabilistically the computation is also robust to the logic signal fluctuations that will arise as operation approaches the thermal limit of computation.

This paper is organized as follows. In Section 2 we provide a brief overview of Markov Random Fields and the Gibbs distribution. Section 3 gives examples of how logic may be expressed using the MRF model and how this provides fault tolerance to both structural and signal based errors. Section 4 concludes the paper.

2. OUR PROPOSED APPROACH

2.1. The Markov Random Network

The basis for our architectural approach is the Markov random network, a network embodiment of the mathematical concept, the Markov random field (MRF) [10]. The Markov random field defines a set of random variables, $\mathbf{\Lambda} = \{\lambda_1, \lambda_2 \ldots, \lambda_k\}$. Each variable λ_i can take on various values, e.g. state labels. Associated with each variable, λ_i, is a neighborhood, $\mathcal{N}_i$, which is a set of variables from $\{\mathbf{\Lambda} - \lambda_i\}$. Simply put, the probability of a given variable depends only on a (typically small) neighborhood of other

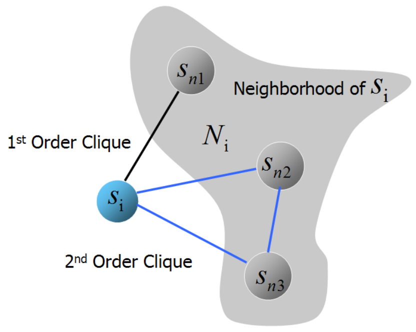


Figure 3: The neighborhood of a network node. The transition probabilities can be expressed in terms of node cliques.

variables. In our model, the variables represent states of nodes in a network. The arcs or edges in the network convey the conditional probabilities with respect to the neighboring nodes.

The definition of the MRF is as follows:

$$P(\lambda) > 0, \quad \forall\lambda \in \mathbf{\Lambda} \qquad (Positivity) \tag{1}$$

$$P(\lambda_i|\{\mathbf{\Lambda} - \lambda_i\}) = P(\lambda_i|\mathcal{N}_i) \quad (Markovianity) \tag{2}$$

The conditional probability of a node state in terms of its neighborhood can be formulated in terms of cliques. A neighborhood and two clique examples are shown in Figure 3. It can be shown, due to the the Hammersley-Clifford theorem [3], that

$$P(\lambda_i|\{\mathbf{\Lambda} - \lambda_i\}) = \frac{1}{Z} e^{-\frac{1}{k_b T}\Sigma_{c\in\mathcal{C}} U_c(\lambda)}. \tag{3}$$

This form for the probability in equation 3 is called the Gibbs distribution. The normalizing constant Z is called the partition function and insures that P is in the range $[0, 1]$. The set $\mathcal{C}$ is the set of cliques for a given node, i. The function U_c is called the clique energy function and depends only on the nodes in the clique. Note that the probability of states depends on the ratio of clique energy of the MRF to the thermal energy k_bT. For instance, the probability are uniform at high values of k_bT and becomes sharply peaked at low values of k_bT. This mimics the annealing behavior of physical systems. This Gibbs formulation of the Markov random field is an attractive representation for computation, since the physical interpretation of the probabilities in terms of entropy of computation is likely to find ready interpretation in the physical device characteristics.

2.2. Markov Random Field Computation

The previous discussion has suggested an approach to embedding a Markov random network in CNT circuits. We now briefly describe the nature of computation in the MRF framework. The general algorithm for finding individual site labels that maximize the probability of the overall network is called *belief propagation* (BP) [21] and provides an efficient means of solving inference problems by propagating marginal probabilities through the network. There are three essential probability functions:

Joint probability:

$$p(x_0, x_1, \cdots, x_{n-1})$$

Marginal probability:

$$p(x_i) = \sum_{x_0}\sum_{x_1}\cdots\sum_{x_j}\cdots\sum_{x_{n-1}} p(x_0, x_1, \cdots, x_{n-1}), j \neq i$$

Conditional probability:

$$p(x_0, x_1, \cdots, x_{i-1}, x_{i+1}, \cdots|x_i) = \frac{p(x_0, x_1, \cdots, x_{n-1})}{p(x_i)}.$$

We can classify the nodes in the network into those that have defined label probabilities, and those whose values must be determined by the propagation algorithm. The first node type would correspond to a computational input whose value is constrained by the problem setup. Such nodes are called *observable* nodes. The other nodes are called *hidden* nodes. In a logic circuit, we can think the input/output as observable nodes and the others as hidden nodes.

The basic idea of belief propagation is that the probability of state labels at a given node in the network can be determined by marginalizing (summing) over the joint probabilities for the node state given just the probabilities for site labels in the Markov neighborhood, $\mathcal{N}_i$. This marginalization establishes the label probabilities for the next propagation step. It can be shown that this propagation algorithm will converge to the maximum probability site label assignment for the entire network, provided there are no loops [21].

This incremental algorithm has computational complexity on the order of the number of nodes in the network, with a weighting term proportional to the size of the neighborhood. In the case of loops, the marginalization must be done combinatorially over a region of the network that bounds the loops in order to guarantee maximum probability solutions. That is, one would partition the network into a loop-free network of blocks which internally contain loops. However it has been demonstrated that the belief propagation algorithm usually converges to the maximum probability state even in the presence of loops [21]. We will give examples of the belief propagation process in a later section.

3. DESIGN EXAMPLES

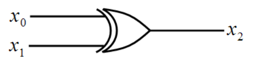
So far, the type of computation to be performed by the network has not been specified. The MRF is a completely general computational framework and in principle any type of computation could be mapped onto the model. In order to concretely illustrate the operation of the model, we will use combinatorial logic as an example. The *programming* of the MRF is straightforward in this case, and will permit some analysis of the fault tolerance of the architecture.

Combinatorial logic can be implemented using a simple, yet powerful, form for the clique energy, called the *auto-model*. For cliques up to order three, the energy function is given by:

$$\begin{aligned} U_c(\lambda) \quad &= \quad \kappa + \Sigma_{i\in\mathcal{C}_0}\alpha_i\lambda_i + \Sigma_{i,j\in\mathcal{C}_1}\beta_{ij}\lambda_i\lambda_j + \\ &\quad\;\; \Sigma_{i,j,k\in\mathcal{C}_2}\gamma_{ijk}\lambda_i\lambda_j\lambda_k. \end{aligned} \tag{4}$$

The constants, α_i, β_{ij} and γ_{ijk} are called *interaction coefficients*. The constant κ acts an energy offset. This form for $U_c(\lambda)$ has been used in many MRF applications including image segmentation, texture classification and object recognition [4] [17].

There are two aspects of fault tolerance that must be considered. Our implementation does not distinguish between devices and connections. Instead, we have structure-based and signal-based faults. Nanoscale devices contain a large number of defects or structural errors, which fluctuate on time scales comparable to the computation cycle. The error will result in variation in the clique energy coefficients in the auto-model in Equation 4. The



i	x_0	x_1	x_2	f
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

Figure 4: The logic compatibility function for an exclusive-or gate with all possible states.

signal-based type of error is directly accounted for in the probability maximization process inherent in the MRF processing. Next, we will illustrate the behavior of the model for each type of errors.

3.1. Structure-based Errors

The effect of structure-based errors, or errors on the coefficients in the clique energy, is illustrated using an XOR example. There are three nodes in the network: the inputs x_0, x_1, and the output x_2 of the gate. Successful operation of the gate is designated by the compatibility function, $f(x_0, x_1, x_2)$ as shown in Figure 4. Here we list all possible states (valid states with $f = 1$ and invalid state with $f = 0$) because our approach *adapts* to errors and we make no assumption about the occurrence of errors. In order to relate the logic compatibility function to a Gibbs energy form it is necessary to use the axioms of the *Boolean ring*. The Boolean ring expresses the rules of symbolic Boolean logic in terms of algebraic manipulations as follows:

$$\begin{aligned} X' &\rightarrow (1 - X) \\ X_1 \wedge X_2 &\rightarrow X_1 X_2 \quad \text{(multiplication)} \\ X_1 \vee X_2 &\rightarrow X_1 + X_2 + X_1 X_2. \end{aligned}$$

The logic variables are treated as real valued algebraic quantities and logic operations are transformed into arithmetic operations. Additionally, it is desired that valid input/output states should have lower clique energies than invalid states. Thus, the clique energy expression is obtained by a negative sum over minterms from the valid states,

$$U(x_0, x_1, x_2) = -\sum_i f_i(x_0, x_1, x_2),$$

where $f_i = 1$, and the minterms are transformed using the Boolean ring rewrite rules. Note that this form exploits the simplification that cross-products of minterms vanish. The Boolean ring conversion for the minterm $(x_0, x_1, x_2) = 000$ is,

$$\begin{aligned} x_0' \wedge x_1' \wedge x_2' &= (1 - x_0)(1 - x_1)(1 - x_2) \\ &= (1 - x_0 - x_1 + x_0 x_1)(1 - x_2) \\ &= 1 - x_0 - x_1 - x_2 + x_0 x_1 + x_0 x_2 + \\ &\quad x_1 x_2 - x_0 x_1 x_2. \end{aligned}$$

For the exclusive-or example, by summing over the valid states, $000 = (1 - x_0)(1 - x_1)(1 - x_2), 011 = (1 - x_0)x_1 x_2, 101 = x_0(1 - x_1)x_2$, and $110 = x_0 x_1 (1 - x_2)$, we can compute the clique energy as following:

$$\begin{aligned} U &= -1 - (1 - x_0)(1 - x_1)x_2 - (1 - x_0)x_1(1 - x_2) - \\ &\quad x_0(1 - x_1)(1 - x_2) - x_0 x_1 x_2 \\ &= -1 + x_0 + x_1 + x_2 - 2x_0 x_1 - 2x_0 x_2 - 2x_1 x_2 + \\ &\quad 4x_0 x_1 x_2. \end{aligned} \tag{5}$$

If we take the structural errors into consideration in our design, the clique energy in Eqn. 4 can be rewritten as:

$$\begin{aligned} U &= \kappa + Ax_0 + Bx_1 + Cx_2 + Dx_0 x_1 + \\ &\quad Ex_0 x_2 + Fx_1 x_2 + Gx_0 x_1 x_2. \end{aligned} \tag{6}$$

where κ is a constant, and the nominal *weight* values for the coefficients are:$A = B = C = D = E = F = G = 1$, as derived above for the error free case.

In the modified equation above, the energy coefficients have been replaced by variables to indicate that their values can deviate from the ideal setting due to failures. The variables A, B, C stand for the first-order clique energy coefficients, and D, E, F are second coefficients. The third order coefficient, G, constrains the values of all the lower order coefficients as will be shown shortly. In the nanoarchitecture being described here, the coefficient values are determined by a set of gate connections, so coefficient error is caused by structure-based failure. For successful operation of the logic, it is necessary that the energy of correct logic state configurations always be *less* than invalid state configurations.

valid state	energy relation to invalid states		
000	κ	$<$	κ +C
	κ	$<$	κ +B
	κ	$<$	κ +A
	κ	$<$	κ +A+B+C-2D-2E-2F+4G
011	κ+B+C-2F	$<$	κ +C
	κ+B+C-2F	$<$	κ +B
	κ+B+C-2F	$<$	κ +A
	κ+B+C-2F	$<$	κ +A+B+C-2D-2E-2F+4G
101	κ+A+C-2E	$<$	κ +C
	κ+A+C-2E	$<$	κ +B
	κ+A+C-2E	$<$	κ +A
	κ+A+C-2E	$<$	κ +A+B+C-2D-2E-2F+4G
110	κ+A+B-2D	$<$	κ +C
	κ+A+B-2D	$<$	κ +B
	κ+A+B-2D	$<$	κ +A
	κ+A+B-2D	$<$	κ +A+B+C-2D-2E-2F+4G

Table 1: The inequalities that must hold among the energy coefficients for successful gate operation.

For our example, the set of inequalities that must hold is given in Table 1. Here we relate a valid state to all possible invalid states. For example, for valid state $(x_0, x_1, x_2) = 000$ the clique energy in Eqn.6 must evaluate to a lower energy state than all possible invalid state. These relations are given in the 16 inequalities listed in Table 1. These inequalities can be solved using a proposed algorithm similar to Gaussian elimination where a variable that appears with opposite signs in two equations can be eliminated. Applying this procedure to the inequalities in Table 1 the following constraints on the clique coefficients are obtained:

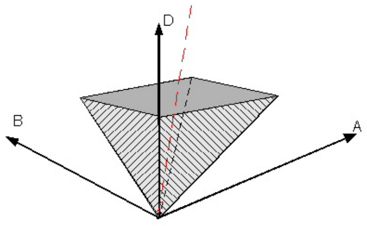


Figure 5: The constraints on the clique coefficients form a cone in the space of coefficient values for $2D > B$ and $2D > A$. The dotted line indicates the nominal coefficient values.

2G>D	2F>C	2E>A	2D>B
2G>F	2F>B	2E>C	2D>A
2G>E			

The constraints should be viewed as being driven by coefficient G which can take on any positive value. A selected value for $G > 0$ then determines bounds on coefficients, D, E, F in terms of ($2G > D, 2G > F, 2G > E$). They in turn bound A, B, C. The bounds are linear, and so the constraints form a *polytope* in the space of energy coefficients. This concept is illustrated in Figure 5 where a projection onto the D, A, B subspace is depicted. In general, the polytope will be a cone whose cross-section increases linearly with the highest order clique coefficient. The nominal values for the coefficients of Figure 5 are $A_0 = D_0$ and $B_0 = D_0$.

We assume a fixed error rate in the connections leads to a coefficient error proportional to its value. For example, for some error rate α, if coefficient D deviates from its nominal value by $D' = D_0 \pm \epsilon$, then $\epsilon = \alpha D_0$. The inequality relating $2D > A$ requires that,

$$\begin{aligned} 2D_0(1 \pm \alpha) &> D_0(1 \pm \alpha) \quad \text{or} \\ 2 * (1 - \alpha) &> (1 + \alpha), \end{aligned}$$

when the worst case condition is used. Thus, α can be as large as 1/3 without causing a failure of the inequality. The constraint on the D coefficient also permits $\alpha < 1/3$. Similar conditions arise from considering the remaining constraints. Thus for the XOR circuit, up to one third of the connections can be bad and the correct logic state will still be achieved.

From this example, we observe that complex logic can be decomposed into simple designs by exploiting properties embedded in a circuit. In general, the highest order clique coefficient can be increased until the lowest order coefficient has sufficient connection redundancy to be guaranteed to attain the average error rate. This policy guards against catastrophic failure, where a few bad connections affect a large percentage of the coefficient values. The conical structure of the constraint surface insures that this strategy is always possible.

3.2. Errors in Signal

3.2.1. State probability

The use of a probabilistic approach to logic has the advantage that the process is inherently fault tolerant to errors in the value of logic state variables. The behavior of a simple inverter circuit will be used to illustrate this aspect of the Markov random network approach.

The Gibbs distribution for an inverter is given by:

$$p(x_0, x_1) = \frac{1}{Z} e^{-\frac{1}{k_b T}(2x_0x_1 - x_0 - x_1)},$$

here x_0 is the input and x_1 is the output of an inverter. $2x_0x_1 - x_0 - x_1$ is the clique energy or auto-model of an inverter. The partition function Z normalizes the expression as required for a probability. Suppose the input, x_0, takes on values from $\{0, 1\}$. The dependence on the input x_0 can be marginalized away by summing over its possible values, i.e,

$$\begin{aligned} p(x_1) &= \frac{1}{Z} \sum_{x_0=\{0,1\}} e^{-\frac{1}{k_b T}(2x_0x_1 - x_0 - x_1)} \\ &= \frac{e^{\frac{x_1}{k_b T}} + e^{\frac{(1-x_1)}{k_b T}}}{2(1 + e^{\frac{1}{k_b T}})}. \end{aligned} \tag{7}$$

In the marginalization it is assumed that the input to the inverter is equally likely to be a zero or one and that the inverter has exact clique energy weights. These assumptions are somewhat idealized, since in practice the inverter will have variable clique coefficients and the input will range over a continuous set of values near zero or one according to the distribution of signal noise and device error.

The marginalized inverter output distribution function is plotted in Figure 6 for various values of k_bT. Both output 0 and 1 are equally likely, but note that the most likely outputs are 0 or 1 with the likelihood of any intermediate values becoming vanishingly small as $k_bT \to 0$. This behavior is characteristic of Markov random network processing. As long as the energy balance is favorable to the correct logic state, decreasing k_bT will lock in the valid configurations.

Most applications of the MRF model treat k_bT as a variable that can be manipulated in solving for the maximum probability state. For the purposes of the following examples, k_bT is expressed in normalized units relative to the logic state energy. That is, thermal energy is expressed as a fraction of unit logic state energy. For example, the value $k_bT = 0.1$ means that the unit logic energy is ten times the thermal energy, so at room temperature, the unit logic energy in some physical realization of the Markov network would be 0.26 electron-volts.

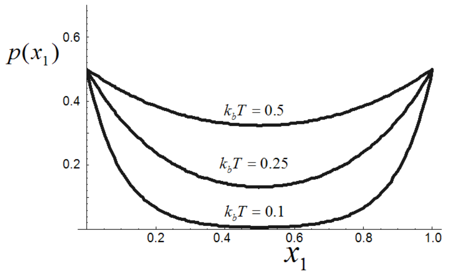


Figure 6: The probability of an inverter output for different values of k_bT.

In actual operation of a logic circuit, the input states would not be equally likely but would have higher probability of being in a given state, as required by deterministic behavior. For example, suppose the input to the inverter has $p(1) = 0.7, p(0) = 0.3$ then the Gibbs distribution of Figure 6 is as shown in Figure 7. As the computing entropy increases, this probability margin asymptotically approaches zero. Based on the Maxwell's demon discussion

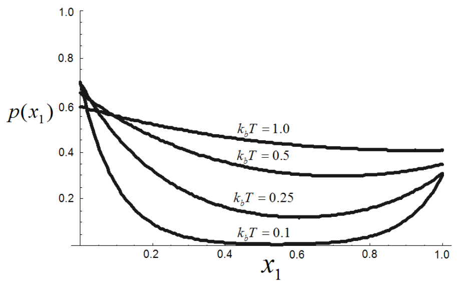


Figure 7: The probability of an inverter output for different values of k_bT when the input is one, with probability 0.7.

in [2], the minimum energy required for bit-operation is $k_bT \ln 2$. Similarly, a nano logic device will by necessity operate with logic energies within a few tens of k_bT in order to achieve the expected reduction in power afforded by the small scale of nanodevices. For finite temperatures, the policy of choosing the output state with the highest probability always yields the correct logic operation. However, it can be expected that errors will result if $|p(x) - 0.5|$ is small, since any physical realization of the Markov network will have significant fluctuation of the logic levels.

3.2.2. Logic processing

In order to consider the error behavior of more complex circuits, it is necessary to describe the processing of logic signals through the Markov random network. This process is carried out by the chaining of conditional probabilities by *belief propagation* [21]. As explained in Section 2.2, the probability of logic variables can be determined by summing (marginalizing) over the set of possible values of clique neighborhood states except for the variable in question. What remains is the probability for the single variable. This probability can be propagated to the next node in the network and used for the next summation. An example of this basic algorithm has already been given in the case of Equation 7.

The Gibbs joint and conditional probability distributions for a NAND gate are given by,

$$\begin{aligned} p(x_a, x_b, x_c) &= \frac{1}{Z} e^{-\frac{1}{k_bT}(2x_ax_bx_c - x_ax_b - x_c)} \\ p(x_c|x_a, x_b) &= \frac{1}{(e^{\frac{x_ax_b}{k_bT}} + e^{\frac{1-x_ax_b}{k_bT}})} e^{-\frac{1}{k_bT}(2x_ax_bx_c - x_ax_b - x_c)}, \end{aligned}$$

where x_a, x_b are the inputs and x_c is the output. Assuming independent inputs, $p(x_a) = p(x_b) = 0.5$, we can obtain the probability of a one at the output by marginalizing over all input combinations,

$$p(x_c) = \sum_{x_a, x_b \in \{0,1\}} \frac{1}{(1 + e^{\frac{1}{k_bT}})} e^{-\frac{1}{k_bT}(2x_ax_bx_c - x_ax_b - x_c)}.$$

This probability distribution is shown in Figure 8. Note that the NAND gate is *asymmetrical* in its probability distribution and for a uniform distribution of inputs, the probability of a one output state is three times that of a zero state. This should be expected, since only one input combination produces a zero output. However, this asymmetry is detrimental to logic processing as shown in Figure 9. Note that it is necessary to have $p(x_a) = p(x_b) > 0.7$ in order to achieve any logic margin. This margin is reduced at higher input

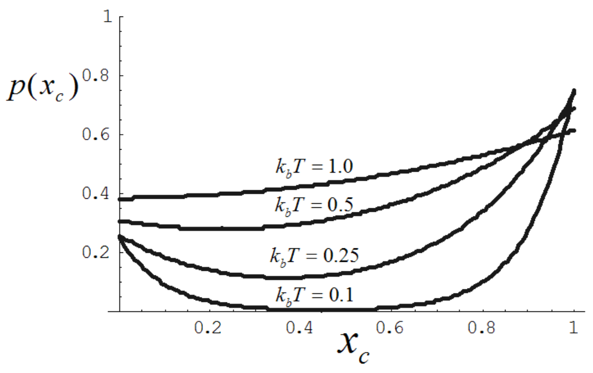


Figure 8: The marginal distribution for the output of a NAND gate for different values of k_bT. The inputs are assumed to have uniform state probabilities.

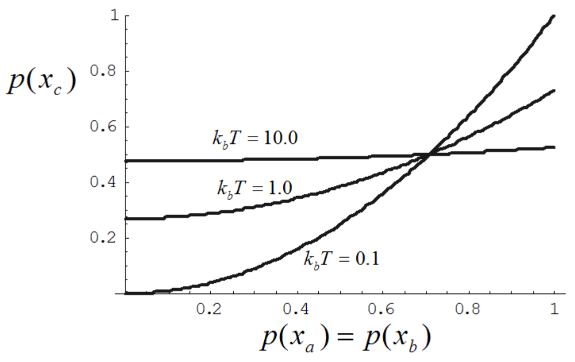


Figure 9: The probability, $p(x_c = 0)$, for a zero output state for a NAND gate as a function of the input state probabilities, $p(x_a) = p(x_b)$.

probabilities as the entropy increases. This result shows that logic structures should be as symmetrical as possible in order to operate close to the thermal limit of $k_bT \ln 2$.

3.2.3. Equivalent logic

To further illustrate the approach, consider the equivalent logic circuits shown in Figure 10. These circuits have identical logic functions but differ in configuration. The marginal distribution of the output state, $p(x_6)$ is shown in Figure 11. Circuit b) has a perfectly symmetrical output distribution and thus can be expected to have significantly better failure tolerance than circuit a), even though they have the same logic function.

This example demonstrates that the Markov framework can be used to optimize circuit configurations for the best signal reliability. However, it should be noted that this treatment assumed perfect device operation. A complete analysis would include both device error and signal error by integrating the device analysis from Section 3.1. Efforts are underway to carry out this integration.

4. CONCLUSION

In this paper, we propose a probabilistic-based design methodology for nanoscale computer architecture. The main reason for se-

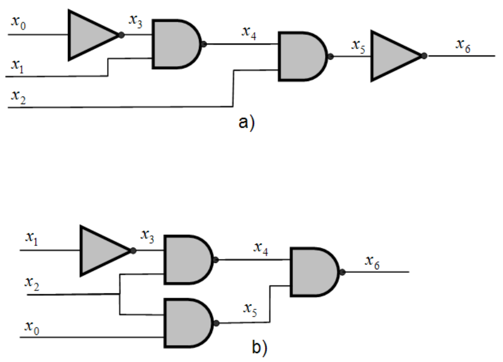


Figure 10: a) The logic function $x_6 = x_2 \wedge (x_0 \vee x_1')$. b) An equivalent logic circuit.

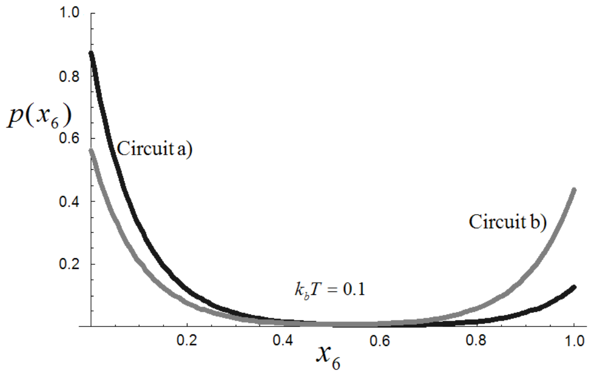


Figure 11: The marginal output distributions for the circuits in Figure 10. Note that circuit a) is highly asymmetrical while circuit b) is perfectly symmetrical.

lecting the Markov random network, belief propagation algorithm, and Gibbs energy distribution as the basis for our nanoarchitectural approach is that its operation does not depend on perfect devices or perfect connections. In operation, the Markov network is updated by iteratively changing the state of nodes and propagating those changes through the network. Ultimately the network converges to a stable set of state probabilities reflecting the required result. A computational result is determined by selecting those visible output states that maximize the probability of their respective nodes. Successful operation only requires that the energy of correct states is lower than the energy of errors.

5. REFERENCES

[1] P. Beckett and A. Jennings. Towards nanocomputer architecture. In *Asia-Pacific Computer Systems Architecture Conference*, 2002.

[2] C. H. Bennett. The thermodynamics of computation—a review. *Intl. Journal of Theoretical Physics*, 21(12):905–940, 1982.

[3] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B*, 36(3):192–236, 1994.

[4] R. Chellappa. *Markov random fields: theory and applications*. Academic Press, 1993.

[5] A. DeHon. Array-based architecture for fet-based, nanoscale electronics. *IEEE Transactions on Nanotechnology*, 2(1):23–32, March 2003.

[6] S. Folling, O. Turel, and K. Likharv. Single-electron latching switches as nanoscale synapses. In *Proc. of Int. Joint Conf. on Neural Networks*, pages 216–221, July 2001.

[7] S. C. Goldstein and M. Budiu. Nanofabrics: Spatial computing using molecular electronics. In *The 28th Annual International Symposium on Computer Architecture*, pages 178–189, 2001.

[8] J. Han and P. Jonker. A system architecture solution for unreliable nanoelectronic devices. *IEEE Transactions on Nanotechnology*, 1(4):201–208, December 2002.

[9] Y. Huang, X. Duan, Y. Cui, L. Lauhon, K. Kim, and C. M. Lieber. Logic gates and computation from assembled nanowire building blocks. *Science*, 294:1313–1317, 2001.

[10] S. Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer, 1995.

[11] C. Lieber. The incredible shrinking circuit. *Scientific American*, pages 59–64, September 2001.

[12] K. K. Likharev. Single-electron devices and their applications. *Proceedings of the IEEE*, 87(4):606–632, April 1999.

[13] S. Luryi, J. M. Xu, and A. Zaslavsky. *Future Trends in Microelectronics: The Road Ahead*. John Wiley & Son, 1999.

[14] R. Martel, V. Derycke, J. Appenzeller, S. Wind, and P. Avouris. Carbon nanotube field-effect transistors and logic circuits. In *Proceedings of the 39th Conference on Design Automation*, pages 94–98, 2002.

[15] M. A. Reed, J. Chen, A. M. Rawlett, D. W. Price, and J. M. Tour. Molecular random access memory cell. *Applied Physics Letters*, 78:3735, 2001.

[16] T. Rueckes, K. Kim, E. Joselevich, G. Tseng, C.-L. Cheung, and C. Lieber. Carbon nanotube-based nonvolatile random access memory for molecular computing. *Science*, 289:94–97, 2000.

[17] R. R. Schulz and R. L. Stevenson. A Bayesian approach to image expansion for improved definition. *IEEE Transactions on Image Processing*, 3(3):233–242, 1994.

[18] J. von Neumann. *Probabilistic Logic and the Synthesis of Reliable Organisms from Unreliable Components*. Automata Studies. Princeton University Press, 1956.

[19] J. M. Xu. Highly ordered carbon nanotube arrays and IR detection. *Infraread Physics & Technology*, 42:485–491, 2001.

[20] Z. Yao, H. W. Postma, L. Balents, and C. Dekker. Carbon nanotube intramolecular junctions. *Nature*, 402:273–276, 1999.

[21] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In *International Joint Conference on AI*, 2001. Distinguished Lecture.

[22] C. Zhou, M. R. Deshpande, M. A. Reed, L. J. II, and J. . M. Tour. Nanoscale metal/self-assembled monolayer/metal heterostructures. *Applied Physics Letters*, 71:611, 1997.