Computation of Boolean Matrix Chain Products in 3D ReRAM

Alvaro Velasquez
Department of Computer Science
University of Central Florida, Orlando, FL
Email: velasquez@eecs.ucf.edu

Sumit Kumar Jha
Department of Computer Science
University of Central Florida, Orlando, FL
Email: http://www.sumitkumarjha.com/contact.html

***Abstract*—Energy concerns, the infamous memory wall, and the enormous data deluge of the current big-data age have made the integration of processing and memory elements into a very appealing paradigm. In this paper, we focus on a computation-in-memory solution to the problem of multiplying a set of Boolean matrices, also known as Boolean matrix chain multiplication (BMCM). This is a fundamental computational task with applications in graph theory, group testing, data compression, and digital signal processing. In particular, we propose a framework for mapping arbitrary instances of BMCM to a 3-dimensional (3D) crossbar memory architecture consisting of 1-diode 1-resistor (1D1R) structures.**

I. Introduction

In recent years, there has been increased interest in unifying the processor and memory in order to alleviate the performance overheads imposed by the memory-processor bottleneck [1]. In this paper, we leverage this unified model of computation with recent advances in crossbar memory technology in order to tackle the problem of multiplying a chain of Boolean matrices within memory efficiently.

The paper is organized as follows. Section II defines a model for 3D crossbars and provides background information on Boolean matrix chain multiplication (BMCM). In Section III, we propose a mapping of the BMCM problem to a 3D crossbar and establish mathematical guarantees on the space complexity and correctness of the procedure. Section IV follows with experimental results and we finalize with concluding remarks in Section V.

II. Background

Traditional Resistive Random Access Memory (ReRAM) architectures are crossbars, or cross-point memories, consisting of two sets of parallel wires, with each wire from one set placed perpendicularly to every wire in the other set. At every junction of wires, there is an interconnect acting as a switch between two wires. These interconnects typically consist of 1-transistor 1-resistor (1T1R) or 1-diode 1-resistor (1D1R) structures. However, it has been shown that in the context of 3-dimensional memory stacking, the usage of 1D1R cells is a more effective [2] and cost-efficient solution [3] than 1T1R. These 3D ReRAM memories are simply crossbars with more than one layer of 1D1R interconnects. An example can be seen in Figure 1. An abstraction of these memories can be found in Definition 1.

Definition 1. 3D Crossbar *A $p \times n \times L$ 3D crossbar is a 3-tuple $\mathbb{C} = (M, R, C)$ where*

- $\mathbb{M} = \{M^1, \ldots, M^L\}$ *is a set of $p \times n$ Boolean matrices, where* $M^k = \begin{pmatrix} m^k_{11} & m^k_{12} & \ldots & m^k_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ m^k_{p1} & m^k_{p2} & \ldots & m^k_{pn} \end{pmatrix}$ *represents a matrix of interconnects with p rows and n columns. Each $m^k_{ij} \in \{0, 1\}$ denotes the state of the device connecting row i with column j in layer k;*
- $\mathbb{R} = \{R^1, \ldots, R^{L_R}\}$ *is the set of row wire vectors, where $R^k = \{r^k_1, \ldots, r^k_p\}$ and $r^k_i \in \{0, 1\}$ provides the same input voltage to every interconnect in row i of layer $2k-1$.*
- $\mathbb{C} = \{C^1, \ldots, C^{L_C}\}$ *is the set of column wire vectors, where $C^k = \{c^k_1, \ldots, c^k_n\}$ and $c^k_j \in \{0, 1\}$ provides the same input voltage to every interconnect in column j of layer $2k$.*

Definition 1 provides a high-level abstraction that can model an electrical system. For example, $r_1 = 1$ denotes that wire r_1 has a significant flow of electric current while $r_1 = 0$ denotes negligible flow. It follows then that the physical meaning of *Axiom 1* is that of unidirectional flow of current. This means that the interconnects m_{ij} act as unidirectional switches, similar to a diode or a rectifying memristor [4] [5]. A survey of such interconnects can be found in [2]. In our abstraction, $m_{ij} = 1$ acts like a closed switch which redirects current from one terminal to the other, while $m_{ij} = 0$ acts like an open switch that does not allow current to flow from one terminal to the other. For the rest of this paper, we assume square matrices for the sake of simplicity; however, the proposed approach applies to matrices of arbitrary sizes.

Axiom 1 (Unidirectional Flow)**.** *Let $\mathbb{C} = (M, R, C)$ be an $n \times n \times L$ crossbar. Then* $\forall i, j, k; 1 \leq i, j \leq n; 1 \leq k \leq L-1 :$
$\left[\left(r^k_i \wedge m^{2k-1}_{ij}\right) \implies c^k_j\right] \wedge \left[\left(c^k_j \wedge m^{2k}_{ij}\right) \implies r^{k+1}_i\right]$

While the problem of performing logic computations using crossbars [6], [7], [8], [9], [10], [11], [12], [13] has been extensively studied in the literature, leveraging the structure of a 3-dimensional crossbar for computation has largely gone unexplored. In this paper, we take a step in this direction by computing Boolean matrix chain products within 3D ReRAM. We use the terms 3D ReRAM and crossbar interchangeably.

978-1-4673-6853-7/17/$31.00 ©2017 IEEE

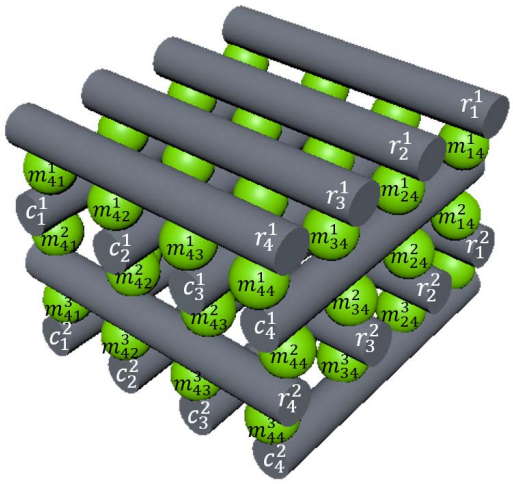

Fig. 1: $4 \times 4 \times 3$ 3D crossbar. There are two sets of row and column wires and three layers of interconnects.

The Boolean matrix multiplication (BMM) problem may be defined as follows. Given two Boolean matrices $X^1 = (x^1_{ij}) \in \{0,1\}^{n\times n}$ and $X^2 = (x^2_{ij}) \in \{0,1\}^{n\times n}$, we wish to compute their product $S = X^1X^2 = (s_{ij})$, where $s_{ij} = \bigvee_{k=1}^{n} (x^1_{ik} \wedge x^2_{kj})$.

$$X^1 = \begin{pmatrix} x^1_{11} & \dots & x^1_{1n} \\ \vdots & \ddots & \vdots \\ X^1_{n1} & \dots & x^2_{nn} \end{pmatrix}, \quad X^2 = \begin{pmatrix} x^2_{11} & \dots & x^2_{1n} \\ \vdots & \ddots & \vdots \\ x^2_{n1} & \dots & x^2_{nn} \end{pmatrix}$$

$$S = \begin{pmatrix} \bigvee_{i=1}^{n} (x^1_{1i} \wedge x^2_{i1}) & \bigvee_{i=1}^{n} (x^1_{1i} \wedge x^2_{i2}) & \dots & \bigvee_{i=1}^{n} (x^1_{1i} \wedge x^2_{in}) \\ \bigvee_{i=1}^{n} (x^1_{2i} \wedge x^2_{i1}) & \bigvee_{i=1}^{n} (x^1_{2i} \wedge x^2_{i2}) & \dots & \bigvee_{i=1}^{n} (x^1_{2i} \wedge x^2_{in}) \\ \vdots & \vdots & \ddots & \vdots \\ \bigvee_{i=1}^{n} (x^1_{ni} \wedge x^2_{i1}) & \bigvee_{i=1}^{n} (x^1_{ni} \wedge x^2_{i2}) & \dots & \bigvee_{i=1}^{n} (x^1_{ni} \wedge x^2_{in}) \end{pmatrix}$$

Given Boolean matrices $X^1, \dots, X^\alpha$, where $X^k = (x^k_{ij}) \in \{0,1\}^{n\times n}$, we define the k-chain product of these matrices as $S^k = (s^k_{ij}) = X^1X^2 \cdots X^{k+1}$ as defined by equation (3).

III. Methodology

The basis of our method consists of redirecting the flow of information through the crossbar based on the values of its interconnects. In the case of an electrical system, this can be achieved by applying a high voltage with respect to ground on some row wires in the first layer of the crossbar, grounding all of the bottommost wires in said crossbar, and configuring the interconnects based on variables of the formula ϕ that we wish to compute in such a way that electrical current will flow into the grounded wires if and only if $\phi = 1$. In the context of our abstraction, this means that, given a 3D crossbar $\mathbb{C} = \{M, \{R^1, \dots, R^{L_R}\}, \{C^1, \dots, C^{L_C}\}\}$, a voltage is applied at some row wires $r^1_{i_1} = r^1_{i_2} = \cdots = r^1_{i_k} = 1$, which will generate a flow of current such that, by successively applying *Axiom 1*, will result in some rows $r^{L_R}_{i_1} = r^{L_R}_{i_2} = \cdots = r^{L_R}_{i_k} = 1$ having flow as well (or $c^{L_C}_{i_1} = c^{L_C}_{i_2} = \cdots = c^{L_C}_{i_k} = 1$).

For any layer k, we can define the values of r^k_i and c^k_j as specified in (1) and (2), respectively.

$$r^k_i = \bigvee_{j=1}^{n} \left(c^{k-1}_j \wedge m^{2(k-1)}_{ij} \right), r^1_i \text{ is the } i^{th} \text{ input} \tag{1}$$

$$c^k_j = \bigvee_{i=1}^{n} \left(r^k_i \wedge m^{2k-1}_{ij} \right) \tag{2}$$

$$s^k_{ij} = \bigvee_{t=1}^{n} \left(s^{k-1}_{it} \wedge x^{k+1}_{tj} \right), s^1_{ij} = \bigvee_{t=1}^{n} \left(x^1_{it} \wedge x^2_{tj} \right) \tag{3}$$

We can expand equations (1) and (2) to obtain (4) and (5). Note the striking similarity between the structure of these two formulas and (6), which corresponds to the expansion of equation (3). Recall that (3) is an arbitrary entry in the matrix corresponding the product of some Boolean matrices $X^1, \dots, X^{k+1}$. This similarity provides some intuition as to the feasibility of mapping the BMCM problem onto a 3-dimensional crossbar. In fact, the mapping is rather simple, as specified in *Theorem 1*. The crux of the procedure lies in configuring the layers of interconnects according to (7). A pictorial representation can be found in Figure 2.

$$M^k = \begin{cases} X^{k+1} & , k \text{ odd} \\ (X^{k+1})^T & , k \text{ even} \end{cases} \tag{7}$$

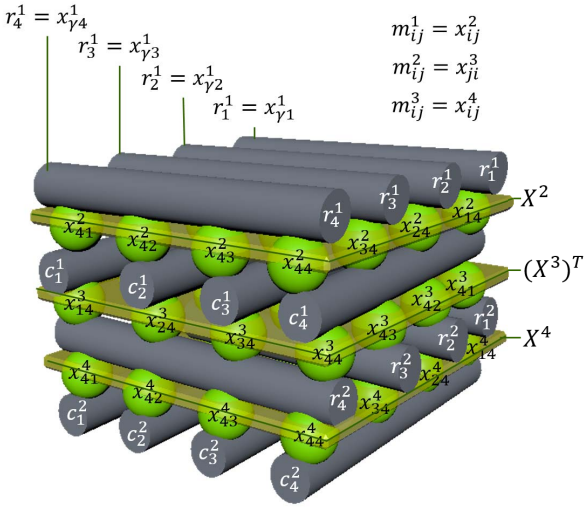

Fig. 2: $4\times4\times3$ crossbar illustrating the configuration proposed in order to compute the product of Boolean matrices.

Theorem 1. *Let $\mathbb{C} = \{M, R, C\}$ be an $n \times n \times L$ 3D crossbar and let $X^1, \dots, X^\alpha, \alpha \geq 2$, denote a set of Boolean matrices with k-chain product $S^k = (s^k_{ij}) = X^1X^2 \cdots X^{k+1}$. If $M^k = \begin{cases} X^{k+1} & , k \text{ odd} \\ (X^{k+1})^T & , k \text{ even} \end{cases}$, then $\begin{cases} c^{\alpha/2}_j = s^{\alpha-1}_{\gamma j} & , \alpha \text{ even} \\ r^{(\alpha+1)/2}_i = s^{\alpha-1}_{\gamma i} & , \alpha \text{ odd} \end{cases}$ for any row index $\gamma \in \{1, \dots, n\}$.*

$$r_i^k = \bigvee_{j_{k-1}=1}^{n} \left(\dots \left(\bigvee_{j_2=1}^{n} \left(\bigvee_{i_2=1}^{n} \left(\bigvee_{j_1=1}^{n} \left(\bigvee_{i_1=1}^{n} \left(r_{i_1}^1 \wedge m_{i_1 j_1}^1 \right) \wedge m_{i_2 j_1}^2 \right) \wedge m_{i_2 j_2}^3 \right) \wedge m_{i_3 j_2}^4 \right) \dots \right) \wedge m_{i j_{k-1}}^{2(k-1)} \right) \tag{4}$$

$$c_j^k = \bigvee_{i_k=1}^{n} \left(\dots \left(\bigvee_{j_2=1}^{n} \left(\bigvee_{i_2=1}^{n} \left(\bigvee_{j_1=1}^{n} \left(\bigvee_{i_1=1}^{n} \left(r_{i_1}^1 \wedge m_{i_1 j_1}^1 \right) \wedge m_{i_2 j_1}^2 \right) \wedge m_{i_2 j_2}^3 \right) \wedge m_{i_3 j_2}^4 \right) \dots \right) \wedge m_{i k j}^{2k-1} \right) \tag{5}$$

$$s_{\gamma j}^k = \bigvee_{i=1}^{n} \left(\dots \left(\bigvee_{j_2=1}^{n} \left(\bigvee_{i_2=1}^{n} \left(\bigvee_{j_1=1}^{n} \left(\bigvee_{i_1=1}^{n} \left(x_{\gamma i_1}^1 \wedge x_{i_1 j_1}^2 \right) \wedge x_{j_1 i_2}^3 \right) \wedge x_{i_2 j_2}^4 \right) \wedge x_{j_2 i_3}^5 \right) \dots \right) \wedge x_{ij}^{k+1} \right) \tag{6}$$

Proof. Let $r_i^1 = x_{\gamma i}^1$. The proof is by induction on α, where the notations $=_{(i)}$ and $=_{IH}$ indicate that the result follows from equation (i) and the inductive hypothesis, respectively.
Base case: When $\alpha = 2$,

$$c_j^{\alpha/2} = c_j^1 =_{(2)} \bigvee_{i=1}^{n} (r_i^1 \wedge x_{ij}^2) = \bigvee_{i=1}^{n} (x_{\gamma i}^1 \wedge x_{ij}^2) =_{(3)} s_{\gamma j}^1$$

Inductive hypothesis: Assume that $c_j^{\beta/2} = s_{\gamma j}^{\beta-1}$ for even β and $r_i^{(\beta+1)/2} = s_{\gamma i}^{\beta-1}$ for odd β.
Inductive step: For $\alpha = \beta + 1 > 2$, two cases arise.

- $\beta + 1$ even:

$$\begin{aligned}
c_j^{\alpha/2} = c_j^{(\beta+1)/2} &=_{(2)} \bigvee_{i=1}^{n} \left(r_i^{(\beta+1)/2} \wedge m_{ij}^{\beta} \right) \\
&=_{(1)} \bigvee_{i=1}^{n} \left(\bigvee_{k=1}^{n} \left(c_k^{(\beta-1)/2} \wedge m_{ik}^{\beta-1} \right) \wedge m_{ij}^{\beta} \right) \\
&=_{(7)} \bigvee_{i=1}^{n} \left(\bigvee_{k=1}^{n} \left(c_k^{(\beta-1)/2} \wedge x_{ki}^{\beta} \right) \wedge x_{ij}^{\beta+1} \right) \\
&=_{IH} \bigvee_{i=1}^{n} \left(\bigvee_{k=1}^{n} \left(s_{\gamma k}^{\beta-2} \wedge x_{ki}^{\beta} \right) \wedge x_{ij}^{\beta+1} \right) \\
&=_{(3)} \bigvee_{i=1}^{n} \left(s_{\gamma i}^{\beta-1} \wedge x_{ij}^{\beta+1} \right) =_{(3)} s_{\gamma j}^{\beta}
\end{aligned}$$

- $\beta + 1$ odd:

$$\begin{aligned}
r_i^{(\alpha+1)/2} &= r_i^{(\beta+2)/2} \\
&=_{(1)} \bigvee_{j=1}^{n} \left(c_j^{\beta/2} \wedge m_{ij}^{\beta} \right) \\
&=_{(2)} \bigvee_{j=1}^{n} \left(\bigvee_{k=1}^{n} \left(r_k^{\beta/2} \wedge m_{kj}^{\beta-1} \right) \wedge m_{ij}^{\beta} \right) \\
&=_{(7)} \bigvee_{j=1}^{n} \left(\bigvee_{k=1}^{n} \left(r_k^{\beta/2} \wedge x_{kj}^{\beta} \right) \wedge x_{ji}^{\beta+1} \right) \\
&=_{IH} \bigvee_{j=1}^{n} \left(\bigvee_{k=1}^{n} \left(s_{\gamma k}^{\beta-2} \wedge x_{kj}^{\beta} \right) \wedge x_{ji}^{\beta+1} \right) \\
&=_{(3)} \bigvee_{j=1}^{n} \left(s_{\gamma j}^{\beta-1} \wedge x_{ji}^{\beta+1} \right) =_{(3)} s_{\gamma i}^{\beta}
\end{aligned}$$

□

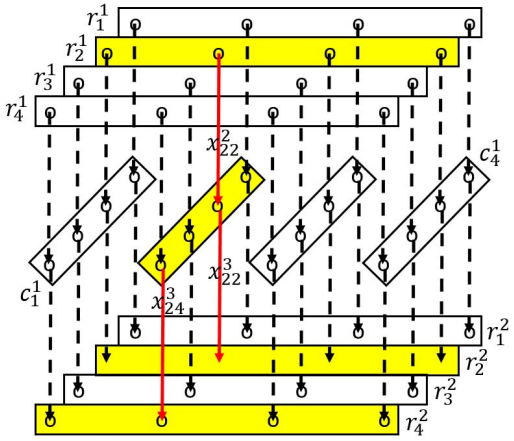

Fig. 3: Dashed lines represent interconnections between row and column wires. Yellow bars denote a wire with a truth value of 1 and solid red lines are due to *Axiom 1*; they correspond to interconnects redirecting flow from one wire to another.

By applying *Theorem 1* on $X^1, \dots, X^\alpha \in \{0,1\}^{n \times n}$ for some row index γ, we can compute the entries in the γ^{th} row vector of $S^{\alpha-1}$. That is, the values of $s_{\gamma 1}^{\alpha-1}, s_{\gamma 2}^{\alpha-1}, \dots, s_{\gamma n}^{\alpha-1}$ will be contained in $r_1^{(\alpha+1)/2}, r_2^{(\alpha+1)/2}, \dots, r_n^{(\alpha+1)/2}$ when α is odd and in $c_1^{\alpha/2}, c_2^{\alpha/2}, \dots, c_n^{\alpha/2}$ when α is even. Therefore, we need only repeat this procedure n times, one for each $\gamma \in \{1, \dots, n\}$, in order to compute $S^{\alpha-1}$. We elucidate this approach with a simple example. Let $X^1, X^2, X^3 \in \{0,1\}^{4\times 4}$ be defined below.

$$X^1 = X^2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad X^3 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \end{pmatrix}$$

Clearly, $S^2 = X^1 X^2 X^3 = X^3$ since X^1 and X^2 are identity matrices. This problem instance can be mapped to a $4 \times 4 \times 2$ crossbar $\mathbb{C} = (\{M^1, M^2\}, \{R^1, R^2, R^3, R^4\}, \{C^1, C^2, C^3, C^4\})$ by setting $M^1 = X^2$ and $M^2 = (X^3)^T$. In order to compute $s_{11}^2, s_{12}^2, s_{13}^2, s_{14}^2$, let $r_1^1 = 1, r_2^1 = r_3^1 = r_4^1 = 0$. From *Theorem 1*, it follows that $r_1^2, r_2^2, r_3^2, r_4^2$ will hold the values of $s_{11}^2, s_{12}^2, s_{13}^2, s_{14}^2$. Similarly, the values for the second row of S^2 $(s_{21}^2, s_{22}^2, s_{23}^2, s_{24}^2)$ are computed by setting $r_2^1 = 1, r_1^1 = r_3^1 = r_4^1 = 0$ as can be seen in Figure 3.

IV. Experimental Results

These results have been verified through HSPICE simulations using Schottky diodes and resistors in 1D1R structures. Interconnects corresponding to variables with value 1 have ON resistances of 10Ω and variables with value 0 have OFF resistances of $100\text{k}\Omega$. A $2V$ voltage pulse was applied on the topmost row wires in accordance with *Theorem 1* and a resistor-to-ground with resistance $1M\Omega$ was placed on each of the bottommost wires in order to read the outputs, which are shown in the matrix R_V below. Note that the entries of R_V coincide with the entries of $S^2 = X^1X^2X^3 = X^3$ as intended. It can be seen that whenever $s^2_{ij} \in S^2$ evaluates to 1, a voltage value (with respect to ground) of $6-8mV$ is read. Conversely, whenever $s^2_{ij} \in S^2$ evaluates to 0, a voltage value of approximately $6\mu V$ is measured at the outputs. We have also verified our approach on randomized Boolean matrices using a varying number of layers. While there is a significant read margin between 0 and 1 values when up to 4 layers are used, a substantial signal degradation is observed for 3D crossbars with 8 or more layers.

$$R_V = \begin{pmatrix} 6.4374mV & 6.1352\mu V & 6.2094\mu V & 8.4733mV \\ 6.5884\mu V & 8.4738mV & 6.7009\mu V & 8.4738mV \\ 6.4275mV & 6.3153\mu V & 8.4734mV & 6.0886\mu V \\ 6.4288mV & 8.4746mV & 8.4747mV & 7.8898\mu V \end{pmatrix}$$

In order to compare the performance of 3D ReRAM against other popular memory architectures, we have simulated a 2 MB memory using the DESTINY memory modeling tool [14]. The results can be seen in Table I. Note that each memory architecture prevails in some aspect of performance. eDRAM has the lowest read and write latency, SRAM requires a low write energy at the cost of area and substantial leakage power, 2D ReRAM has low leakage and read energy, and 3D ReRAM benefits from exceptionally small area. In fact, DESTINY demonstrates that a 32 GB memory would only occupy an area of 33.763 mm^2 when using 22 nm technology in a 16-layer 3D ReRAM.

	Read Latency (ns)	Write Latency (ns)	Read Energy (pJ)	Write Energy (pJ)	Leakage (mW)	Area (mm^2)
3D ReRAM	124.09	139.51	122.33	129.11	9.852	0.0027
2D ReRAM	14.325	22.9	5.12	3.842	2.268	0.0365
SRAM	41.541	41.541	386.6	2.31	1924	1.259
eDRAM	9.179	9.179	41.558	1009	2.637	0.2764

TABLE I: DESTINY [14] simulations of different memory architectures using 22 nm technology, including the 3D ReRAM used in this paper. In order to avoid bias, we utilize the default parameters included in the simulator.

It is worth noting that the approach proposed in this paper is applicable to matrices whose dimensions are greater than those of the 3D crossbar. This result follows from seminal work found in Cannon's thesis, where it is shown that matrix multiplication can be carried out by smaller sub-matrix block products [15].

V. Conclusion

We have shown how to compute the product of a set of Boolean matrices by mapping it to a 3-dimensional crossbar memory. The correctness of the proposed approach was proven mathematically and a simple example was given to elucidate its effectiveness, with a read margin of three orders of magnitude between the output voltages of 0 and 1 values. This result attempts to ameliorate the divide between the traditional computation model of von Neumann architectures and the memory-processor integration paradigm in 3D ReRAM.

References

[1] Alvaro Velasquez and Sumit Kumar Jha. Parallel computing using memristive crossbar networks: Nullifying the processor-memory bottleneck. In *Design & Test Symposium (IDT), 2014 9th International*, pages 147–152. IEEE, 2014.

[2] H-S Philip Wong, Heng-Yuan Lee, Shimeng Yu, Yu-Sheng Chen, Yi Wu, Pang-Shiu Chen, Byoungil Lee, Frederick T Chen, and Ming-Jinn Tsai. Metal–oxide rram. *Proceedings of the IEEE*, 100(6):1951–1970, 2012.

[3] Dimin Niu, Cong Xu, Naveen Muralimanohar, Norman P Jouppi, and Yuan Xie. Design of cross-point metal-oxide reram emphasizing reliability and cost. In *2013 IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 17–23. IEEE, 2013.

[4] Kuk-Hwan Kim, Sung Hyun Jo, Siddharth Gaba, and Wei Lu. Nanoscale resistive memory with intrinsic diode characteristics and long endurance. *Applied Physics Letters*, 96(5):053106, 2010.

[5] A. Velasquez and S. K. Jha. Parallel boolean matrix multiplication in linear time using rectifying memristors. In *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1874–1877, May 2016.

[6] Alvaro Velasquez and Sumit Kumar Jha. Automated synthesis of crossbars for nanoscale computing using formal methods. In *Nanoscale Architectures (NANOARCH), 2015 IEEE/ACM International Symposium on*, pages 130–136. IEEE, 2015.

[7] Alvaro Velasquez and Sumit Kumar Jha. Fault-tolerant in-memory crossbar computing using quantified constraint solving. In *Computer Design (ICCD), 2015 33rd IEEE International Conference on*, pages 101–108. IEEE, 2015.

[8] Z. Alamgir, K. Beckmann, N. Cady, A. Velasquez, and S. K. Jha. Flow-based computing on nanoscale crossbars: Design and implementation of full adders. In *2016 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1870–1873, May 2016.

[9] Eero Lehtonen, Jussi H Poikonen, and Mika Laiho. Applications and limitations of memristive implication logic. In *Cellular Nanoscale Networks and Their Applications (CNNA), 2012 13th International Workshop on*, pages 1–6. IEEE, 2012.

[10] Mika Laiho and Eero Lehtonen. Arithmetic operations within memristor-based analog memory. In *Cellular Nanoscale Networks and Their Applications (CNNA), 2010 12th International Workshop on*, pages 1–4. IEEE, 2010.

[11] Tezaswi Raja and Samiha Mourad. Digital logic implementation in memristor-based crossbars-a tutorial. In *Electronic Design, Test and Application, 2010. DELTA'10. Fifth IEEE International Symposium on*, pages 303–309. IEEE, 2010.

[12] Shahar Kvatinsky, Guy Satat, Nimrod Wald, Eby G Friedman, Avinoam Kolodny, and Uri C Weiser. Memristor-based material implication (imply) logic: Design principles and methodologies. 2013.

[13] Sumit Kumar Jha, Dilia E Rodriguez, Joseph E Van Nostrand, and Alvaro Velasquez. Computation of boolean formulas using sneak paths in crossbar computing, April 19 2016. US Patent 9,319,047.

[14] Matt Poremba, Sparsh Mittal, Dong Li, Jeffrey S Vetter, and Yuan Xie. Destiny: A tool for modeling emerging 3d nvm and edram caches. In *Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition*, pages 1543–1546. EDA Consortium, 2015.

[15] Lynn E Cannon. A cellular computer to implement the kalman filter algorithm. Technical report, DTIC Document, 1969.