Using conversion algorithm to compensate errors in analog computing via nano-crossbar

Miao Hu, John Paul Strachan, Stanley R. Williams
Miao.hu@hpe.com
Hewlett Packard Enterprise Labs
Outline

– Motivation and concept
– Design challenges
– Devices
– Solution with the conversion algorithm
– Result
– Conclusion
Von-Neumann machines reach its bottleneck

The end of Dennard scaling for general-purpose CMOS

Von-Neumann Bottleneck

Compute with a system of efficient SoCs and accelerators having computing in/near memory features

Michael Byrne, "Memory Is Holding Up the Moore's Law Progression of Processing Power", 2014.
Important applications with high computing complexity but low computing accuracy

–Image classification in Deep Learning neural network

Opportunities for DPE
1) 70-90% of computation time consumed in the Convolution layers [1]
2) Recent work shows that only 10-12 bit representations required to maintain state-of-the-art classification accuracy [2]

Concept of Dot-Product Engine with memristor crossbar

– Memristor crossbar as computing memory

**Input 2:** Array of conductances \( \{G_{ij}\} \)

**Input 1:** Vector of voltages \( \{V_i\} \)

**Output:** Vector of currents \( \{I_i\} \)

Ideal: \( I_j = \sum_i G_{ij} \cdot V_i \)

- Crossbar array naturally represents a matrix
- Compute dot product through Ohm’s Law
- Highly parallel multiply & accumulate – favorable scaling with array size

However…
Many challenges to implement!
Problem

– In a real crossbar, \( I_j^0 \neq \sum_j G_{ij} \cdot V_i \)
– Because of nonlinear device resistance, input/output resistance, wire resistance, temperature and etc...

Device voltage (All 0.5V input)

Input/output resistance: 100 ohm
Wire segment resistance: 10 ohm
Device resistance: 50k ohm to 1M ohm
Challenges to implement DPE with crossbar array

–A realistic DPE needs to address following challenges:

1. A stable and programmable analog device with linear resistance

2. A Transistor-like selector

3. An analog programming scheme

4. An algorithm to mapping mathematic variables with circuit parameters

5. Target applications with fixed matrix values
TaOx memristor device with linear static resistance

- Good linearity of stats up to 10 MΩ
- Good linearity of states up to +/- 0.3V
1T1R crossbar array with linear and stable analog states

4x4 1T1R array

Use feedback programming algorithm to achieve 32 levels with 1uA tolerance

IV DC curves

Programming signal: B1530
Current read: B1500 SMUs
DPE demonstrator – parallel array tester

1T1M crossbar wafer
Array size:
4×4 to 128×64

Pin pad:
Maximum 260 pins:
128 row pins
64 column pins
64 selector pins
4 ground pins

DPE boards:
Parallel signal support:
Functions:
Dot-product operation
Single/Multi device read/write
Pulse width > 160 ns
Voltage: -10 to +10
Summary of the 1T1R devices

- What is the status of achieving linear repeatable response, low power, sufficiently long retention, fast writes, sufficiently distinguishable resistances in different states, and long write endurance in one nanoscale device?
- Is the access device issue solved? What are the remaining issues?

For Dot-product Engine 1T1R devices:
- Linear repeatable response: Good enough < 0.3V
- Long retention: days and weeks, and it’s overall stable.
- Fast writes: < 100 ns
- Distinguishable resistances: 5~6 bits
- Long write endurance: > 10^8
- Access device: transistor is the best solution so far.
  - Problem of existing selectors for analog computing:
    - Variation & stability
    - Nonlinear ON state
    - Yield
    - Require high read voltage for computing
Challenges to implement DPE with crossbar array

A realistic DPE needs to address following challenges:

1. A stable and programmable analog device with linear resistance: ✓
2. A Transistor-like selector: use transistor ✓
3. An analog programming scheme: transistor-assisted close-loop tuning ✓
4. Target applications with fixed matrix values: DFT, Deep networks ✓
5. An algorithm to mapping mathematic variables with circuit parameters ?
Conversion algorithm – Basic idea

– Find $G_{\text{new}}$ satisfy the following equation:

$$\text{For arbitrary } V_{\text{in}}, \quad V_{\text{in}} \cdot G \approx \text{crossbar}(G_{\text{new}}, V_{\text{in}}, \text{etc.})$$

– Benefits:
  – Minimize circuit cost and programming cost
  – Can tolerate most circuit issues
  – Can apply to general matrix.
Result of conversion algorithm

– Use Discrete Cosine Transform as example:

Concentration algorithm tunes device conductance to compensate wire resistance, sneak current, device nonlinearity and yield.
DPE computing accuracy with the conversion algorithm

\[
\text{Std} = 0.0045
\]

**Percentage error distribution. mean = 1std = 0.00445**

Actual value/Ideal value for DCT matrix
(Discrete Cosine Transform)

* LM + restoration assume complex restoration circuit can be afforded for the specific matrix, which is not practical.
DPE computing accuracy vs. Memristor accuracy

Plateau due to remaining device nonlinearity
Energy efficiency and speed estimation

– Circuit assumptions: $P_{diss} = 100 \, \mu W$ per channel, $f_s = 10 \, MHz$, $B = 8$-bit resolution

– FoM: $P = 2^B\cdot f_s = 2.56e9$; $F = P/P_{diss} = 2.56e13$, can be achieved since 2005.


![Graphs showing energy consumption and crossbar size](image)

>90% energy is consumed by peripheral circuits

Softmax neural network on crossbar

– Very slight performance degradation even with large device error
– Because of well-trained weight matrix.

MNIST Recognition accuracy vs. Programming error

Recognition accuracy

0.00% 10.00% 20.00% 30.00% 40.00% 50.00% 60.00% std% of memristor programming error

93.00% 92.00% 91.00% 90.00% 89.00% 88.00% 87.00%
A deep nonlinear encoder network for MNIST (Salakhutdinov and Hinton, AI-Stats 2007)

- ~1% error, 100 misclassification in 10k test samples.
Apply DPE in machine learning

– Partition the matrix with 128×128 crossbars
– Two approaches:

**full DAC+ADC support (use DPE core)**

- **Expensive but error will not propagate**

Tune memristor to very resistive state (like 100 M) to match NaN value

**Pure analog implementation**

- **Error will accumulate and propagate**

Analog signal to next layer
Result comparison

– Performance, tests are repeated 10 times for each noise setting:
  – **1.18%** error for software,
  – **1.13%** error for DPE with 4-bit DAC+ADC
  – **3.70%** error for DPE with pure analog implementation.

– RTN(Binary noise) degrades the system accuracy up to ~0% (DAC+ADC) or ~1% (Pure Analog)

– Pure analog implementation hits the accuracy by **1.48%**
Other applications

– Discrete Fourier Transform, Convolution, IoT

Matrix T for convolution kernel

Miao, ICRC 2016
Conclusion

– We analyzed the challenges for a practical Dot-Product Engine implementation on nano-crossbars

– We present a conversion algorithm with near-zero overhead:
  – Scalable up to 512x512 crossbar model or even more
  – Up to 8-Bit output accuracy
  – <2 second on a normal desktop workstation for 128x128 crossbar

– DPE is excellent as accelerators for off-line machine learning algorithms:
  – More than 3 orders of magnitude improvement comparing to the best possible ASIC
    – 1,000 to 10,000 better speed-energy efficiency product
  – Enough and flexible computing accuracy for trained NNs (no training yet)
Thank you!

For more detail please refer:

Simulated accuracy for softmax neural network on crossbar

- Very slight performance degradation even with large device programming error
- Because of well-trained weight matrix.

![Graph showing MNIST Recognition accuracy vs. Programming error. The graph shows a slight decrease in recognition accuracy with increasing programming error, indicating robust performance.](image-url)
Conversion algorithm optimization: Temperature

- Best to calibrate the temperature to the working temperature, but it has a large tolerance margin.
- Test case: 32x32 crossbar, 10 ohm wire segment, calibrated at 0.25V, Tested at 0~0.5V.

### Mean vs. Temperature

<table>
<thead>
<tr>
<th>Temperature</th>
<th>Mean Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>T = 300K</td>
<td>1.05</td>
</tr>
<tr>
<td>T = 400K</td>
<td>1.11</td>
</tr>
</tbody>
</table>

### Std vs. Temperature

<table>
<thead>
<tr>
<th>Temperature</th>
<th>Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>T = 300K</td>
<td>0.02</td>
</tr>
<tr>
<td>T = 400K</td>
<td>0.03</td>
</tr>
</tbody>
</table>
Theoretical analysis of conversion algorithm for W to G’

–Problem definition:
  –Assume $W$ is positive, can we use a crossbar with wire block resistance $G_w$ and linear devices with tuned conductance map $G’$, to realize ideal calculation $I_{out} = V_{in} \cdot G = a \cdot X \cdot W + b$ for any input vector $X$ with zero error? $a$ and $b$ are coefficients to linearly map $X$ to $V_{in}$ and $W$ to $G$.

–Answer:
  –Yes, there is $G’$ for ideal matrix vector multiplication with arbitrary inputs.
  –However, this $G’$ will be extremely difficult to be analytically calculated.
Start with the simplest example (2x2 crossbar)

**Top node KCL equations:**
(Vt11-Vin1)*Gw + (Vt11-Vt12)*Gw + (Vt11-Vb11)*G11 = 0;
(Vt21-Vin2)*Gw + (Vt21-Vt22)*Gw + (Vt21-Vb21)*G21 = 0;
(Vt12-Vt11)*Gw + (Vt12-Vb12)*G12 = 0;
(Vt22-Vt21)*Gw + (Vt22-Vb22)*G22 = 0;

**Bot node KCL equations:**
(Vb11-Vt11)*G11 + (Vb11-Vb21)*Gw = 0;
(Vb21-Vt21)*G21 + (Vb21-Vb11)*Gw + Vb21*Gw = 0;
(Vb12-Vt12)*G12 + (Vb12-Vb22)*Gw = 0;
(Vb22-Vt22)*G22 + (Vb22-Vb12)*Gw + Vb22*Gw = 0;

**Variable definition:**
Gw: Wire block resistance;
Vtij: top voltage of the cross-point at ith row and jth column
Vbij: bottom voltage of the cross-point at ith row and jth column
Gij: conductance of the cross-point device at ith row and jth column
Calculate voltage across devices $V_{device}$

\[
\begin{bmatrix}
2Gw + G11 & -Gw & -G11 \\
-Gw & 2Gw + G21 & -Gw \\
-Gw & -G11 & Gw + G12 \\
-G11 & -G21 & Gw + G22 \\
-G22 & -Gw & Gw + G11 \\
-G12 & -Gw & 2Gw + G21 \\
-G22 & -Gw & 2Gw + G22
\end{bmatrix}
\begin{bmatrix}
V_{t11} \\
V_{t21} \\
V_{t12} \\
V_{t22} \\
V_{b11} \\
V_{b21} \\
V_{b22}
\end{bmatrix} +
\begin{bmatrix}
-Vin1 \\
-Vin2
\end{bmatrix} = 0
\]

\[
A \begin{bmatrix} V_t \\ V_b \end{bmatrix} + B V_{in}^T = 0 \rightarrow \begin{bmatrix} V_t \\ V_b \end{bmatrix} = -A^{-1} B V_{in}^T
\]

\[
V_{device} = [I \ -I] \begin{bmatrix} V_t \\ V_b \end{bmatrix} = -[I \ -I] A^{-1} B V_{in}^T = C V_{in}^T
\]
To realize ideal matrix vector multiplication

- Sufficient condition: Multiplication at each entry is accurate, $V_{\text{device}} \cdot G' = V_{\text{device\_ideal}} \cdot G$.

Current through the first device in actual condition/ideal condition ($G_w = +\infty$)

$$\begin{bmatrix} C_{11} \cdot G'_{11} & C_{12} \cdot G'_{11} \\ C_{21} \cdot G'_{21} & C_{22} \cdot G'_{21} \\ C_{31} \cdot G'_{12} & C_{32} \cdot G'_{12} \\ C_{41} \cdot G'_{22} & C_{42} \cdot G'_{22} \end{bmatrix} \begin{bmatrix} V_{\text{in1}} \\ V_{\text{in2}} \end{bmatrix} = \begin{bmatrix} G_{11} & 0 \\ 0 & G_{21} \\ G_{12} & 0 \\ 0 & G_{22} \end{bmatrix} \begin{bmatrix} V_{\text{in1}} \\ V_{\text{in2}} \end{bmatrix}$$

$N^3$ nonlinear equations with $N^2$ variables.

- Sufficient and necessary condition: Only matrix vector multiplication result is accurate, $I_{\text{actualoutput}} = I_{\text{idealoutput}}$.

Current through the first column in actual condition/ideal condition ($G_w=+\infty$)

$$\begin{bmatrix} C_{11} \cdot G'_{11} + C_{21} \cdot G'_{21} & C_{22} \cdot G'_{21} + C_{12} \cdot G'_{11} \\ C_{31} \cdot G'_{12} + C_{41} \cdot G'_{22} & C_{32} \cdot G'_{12} + C_{42} \cdot G'_{22} \end{bmatrix} \begin{bmatrix} V_{\text{in1}} \\ V_{\text{in2}} \end{bmatrix} = \begin{bmatrix} G_{11} & G_{21} \\ G_{12} & G_{22} \end{bmatrix} \begin{bmatrix} V_{\text{in1}} \\ V_{\text{in2}} \end{bmatrix}$$

$N^2$ nonlinear equations with $N^2$ variables.
Numerical method to approximate G’

- The main issue of ideal equations is lack of direct physical representation

  This term stands for the contribution of conductance by Vin1 to the first column, but there is no physical term stands for that in the crossbar simulation!

\[
\begin{bmatrix}
C_{11} \cdot G’_{11} + C_{21} \cdot G’_{21} & C_{22} \cdot G’_{21} + C_{12} \cdot G’_{11} \\
C_{31} \cdot G’_{12} + C_{41} \cdot G’_{22} & C_{32} \cdot G’_{12} + C_{42} \cdot G’_{22}
\end{bmatrix}
\begin{bmatrix}
Vin_1 \\
Vin_2
\end{bmatrix}
= 
\begin{bmatrix}
G_{11} & G_{21} \\
G_{12} & G_{22}
\end{bmatrix}
\begin{bmatrix}
Vin_1 \\
Vin_2
\end{bmatrix}
\]

- With device models, it still the best to tune current at every cross-point to the ideal value because:
  1. Current through each device is well-defined in the simulation, it makes calculation much easier.
  2. Since device has voltage dependence, even ideal equations will not guarantee zero error for arbitrary inputs.

\[
\begin{bmatrix}
C_{11} \cdot G’_{11} \cdot Vin_1 + C_{12} \cdot G’_{11} \cdot Vin_2 \\
C_{21} \cdot G’_{21} \cdot Vin_1 + C_{22} \cdot G’_{21} \cdot Vin_2 \\
C_{31} \cdot G’_{12} \cdot Vin_1 + C_{32} \cdot G’_{12} \cdot Vin_2 \\
C_{41} \cdot G’_{22} \cdot Vin_1 + C_{42} \cdot G’_{22} \cdot Vin_2
\end{bmatrix}
= 
\begin{bmatrix}
G_{11} \cdot Vin_1 \\
G_{21} \cdot Vin_2 \\
G_{12} \cdot Vin_1 \\
G_{22} \cdot Vin_2
\end{bmatrix}
\]