Fault Tolerance in Neuromorphic Computing Systems

Yu Wang[1], Mengyun Liu[2], Krishnendu Chakrabarty[2], Lixue Xia[1]

[1]Tsinghua University, China
[2]Duke University, U.S.A.

Email: yu-wang@tsinghua.edu.cn

http://nicsefc.ee.tsinghua.edu.cn

Outline

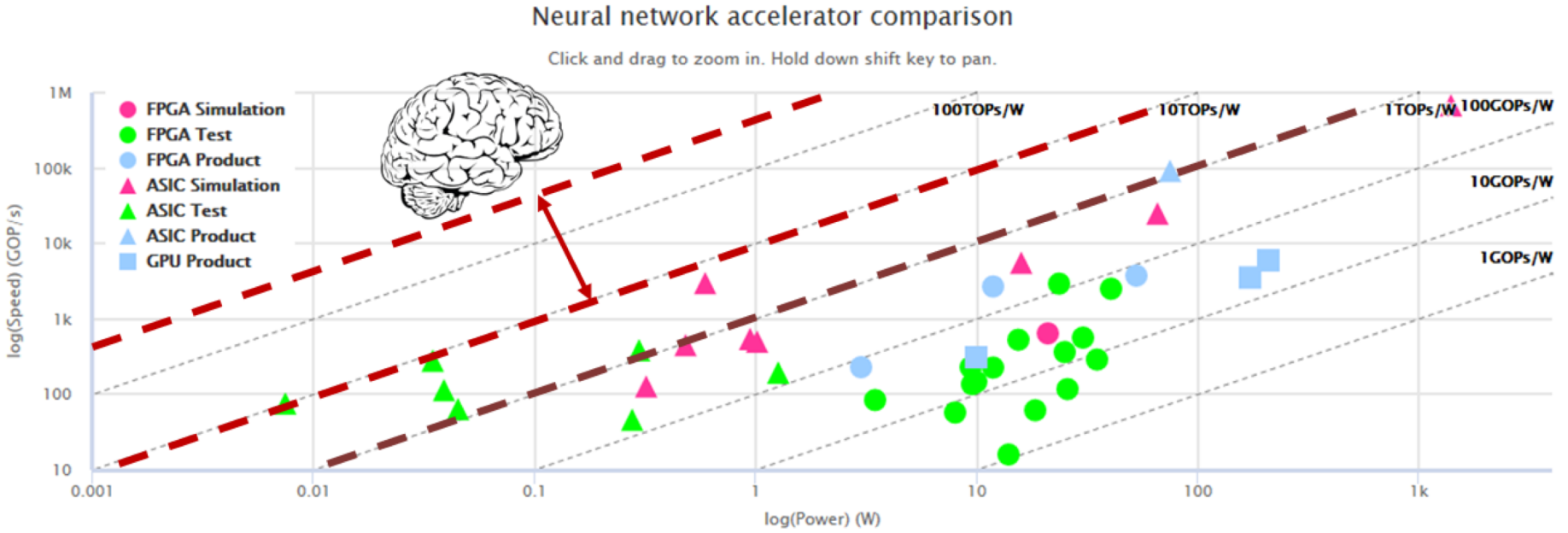
- Background and Introduction
  - Convolutional Neural Network Accelerators
  - RRAM and RRAM-based Computing System
  - RRAM Fault Models
- Fault-tolerance for Device-level Faults
  - Stuck-At-Fault (SAF)
  - Limited Endurance
  - State Drifting Problem and Resistance Variation
  - Non-linear Resistance Distribution
- Fault-tolerance for Circuit-level Faults
  - Wire Resistance and IR-drop
- Fault-tolerance for System-level Faults
  - Unbalanced Writing

Outline

- **Background and Introduction**
  - **Convolutional Neural Network Accelerators**
  - **RRAM and RRAM-based Computing System**
  - **RRAM Fault Models**
- Fault-tolerance for Device-level Faults
  - Stuck-At-Fault (SAF)
  - Limited Endurance
  - State Drifting Problem and Resistance Variation
  - Non-linear Resistance Distribution
- Fault-tolerance for Circuit-level Faults
  - Wire Resistance and IR-drop
- Fault-tolerance for System-level Faults
  - Unbalanced Writing

Neural Network Accelerators

- The energy efficiency of existing Neural Network accelerators is limited at ~10TOPs/W (i.e. 0.1pJ/OP)
  - The gap between brain (500TOPs/W) and accelerators is more than **50x**



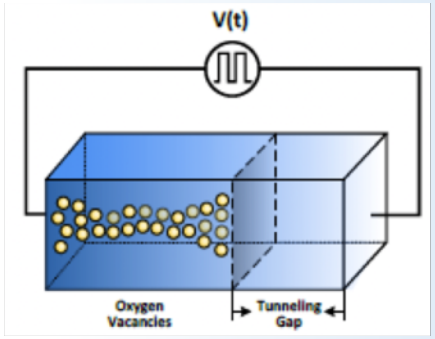
https://nicsefc.ee.tsinghua.edu.cn/projects/neural-network-accelerator/

- Large data movements cause high energy consumption in CNN
  - The data transfer in GPU consumes **2 orders of magnitude** more energy than a floating-point operation [Han S, et al. ISCA' 16]

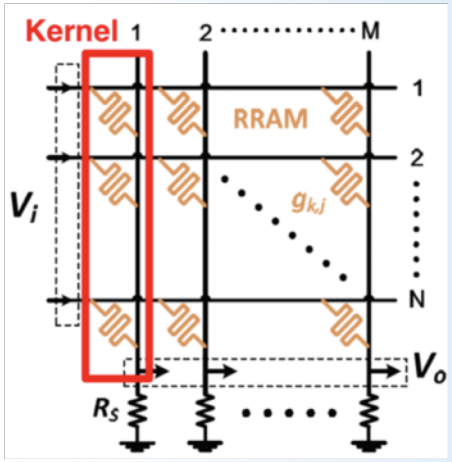
RRAM-based Computing System (RCS)

- ***RRAM and Processing-In-Memory*** provide alternative solutions to realize better implementation of CNN

Nonvolatile RRAM Device



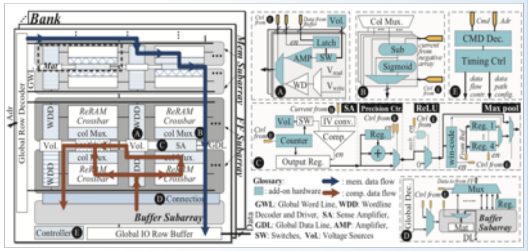
RRAM Crossbar



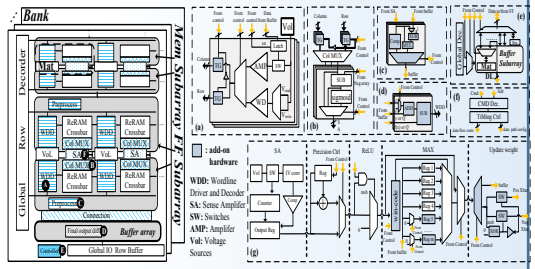
O(n²)→O(n⁰)

$$V_{oj} = r \cdot \sum_k (V_{ik} \cdot g_{kj}),\ g_{kj} = \frac{1}{M_{kj}}$$

RRAM-based Computing System



For Inference: PRIME
[P. Chi, et al. ISCA2016]



For Training: TIME
[M. Cheng, et al. DAC2017]

Representative RCS Design

	RRAM Precision	*Function*	*Performance*	*Ref.*
ISAAC	2 bits	CNN Inference	**14.8x** Throughput and **4.4x** Energy Efficiency (Compared with DaDianNao)	A. Shafiee, et al. ISCA'16
PRIME	4 bits	CNN Inference	**2360x** Speedup and **895x** Energy Efficiency (Compared with DianNao)	P. Chi, et al. ISCA'16
PipeLayer	5~6 bits	CNN Inference and Training	**42.45x** Speedup and **7.17x** Energy Efficiency (Compared with GPU)	L. Song, et al. HPCA'17
TIME	4 bits	CNN/DRL Inference and Training	CNN: **1.3x** Speedup and **19.6x** Energy Efficiency (Compared with DaDianNao) DRL: **126x** Energy Efficiency (Compared with GPU)	M. Cheng, et al. TCAD'18
NTHU Chip	1 bit	Binary DNN/CNN	CNN: 14.8ns/MAC FCN: 15.6ns/MAC (LeNet-5 @ MNSIM)	W. Chen, et al. ISSCC'18

RCS Faults

- The general ***RCS faults*** mean ***RRAM device faults*** and other ***non-ideal factors*** which will cause computation deviation

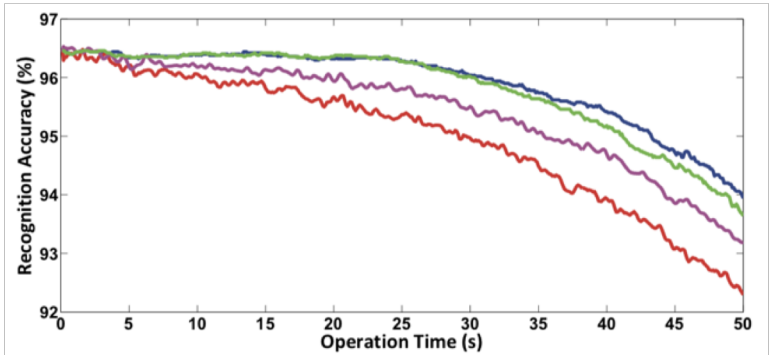
Device-Level
- Stuck-At-Fault (SAF)
- Limited Endurance
- State Drift Problem
- Resistance Variation
- Non-linear Resistance Distribution

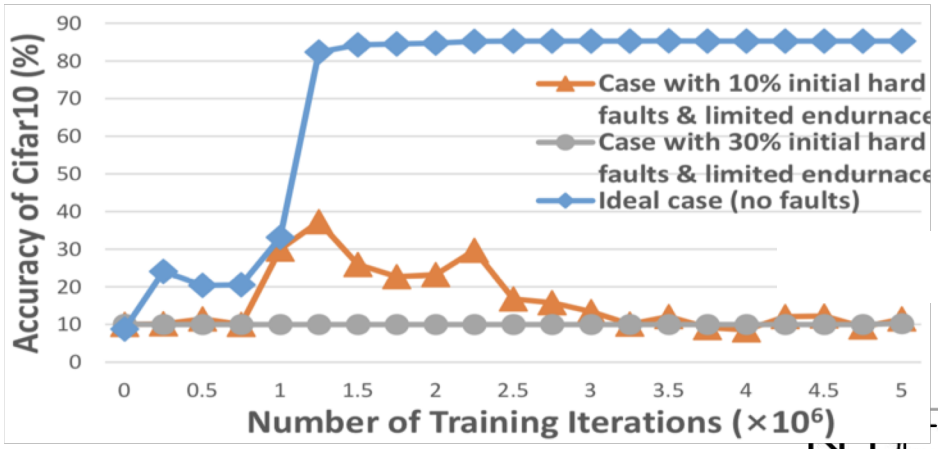
Circuit-Level
- Wire Resistance
- IR-drop
- Sneak Path

System-Level
- Unbalanced Writing

- RCS faults make the CNN computing inaccurate and the system unreliable

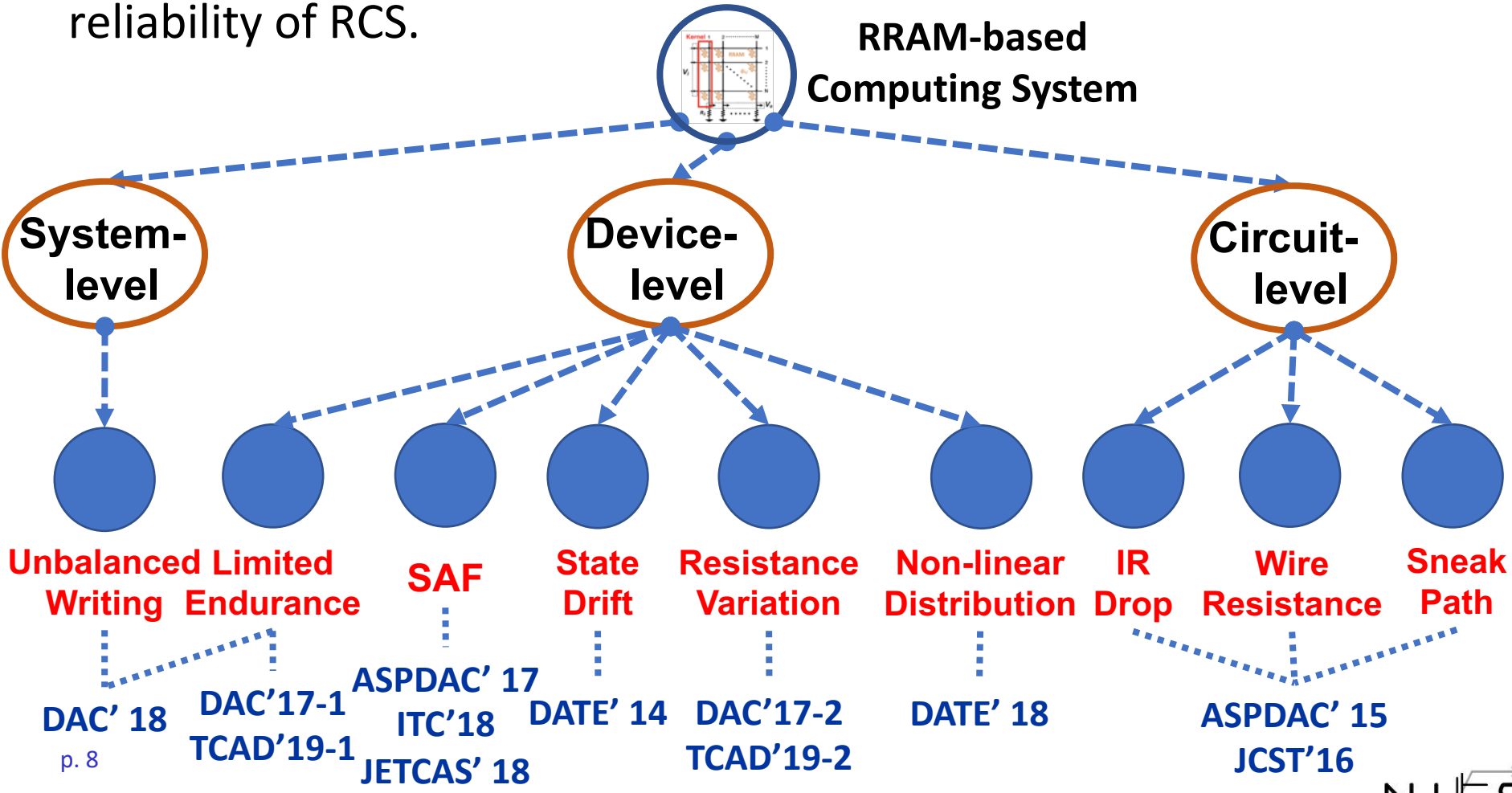
Yield	Ideal	95%	90%	80%
Accuracy	97.8%	26.7~60.4%	15.5~38.6%	10.6~28.0%
Reduction	-	>37%	>59%	>69%





Fault Tolerance in RCS

- According to the type of ***RCS faults***, we have proposed corresponding ***fault tolerance methods*** to rescue the computation accuracy and reliability of RCS.

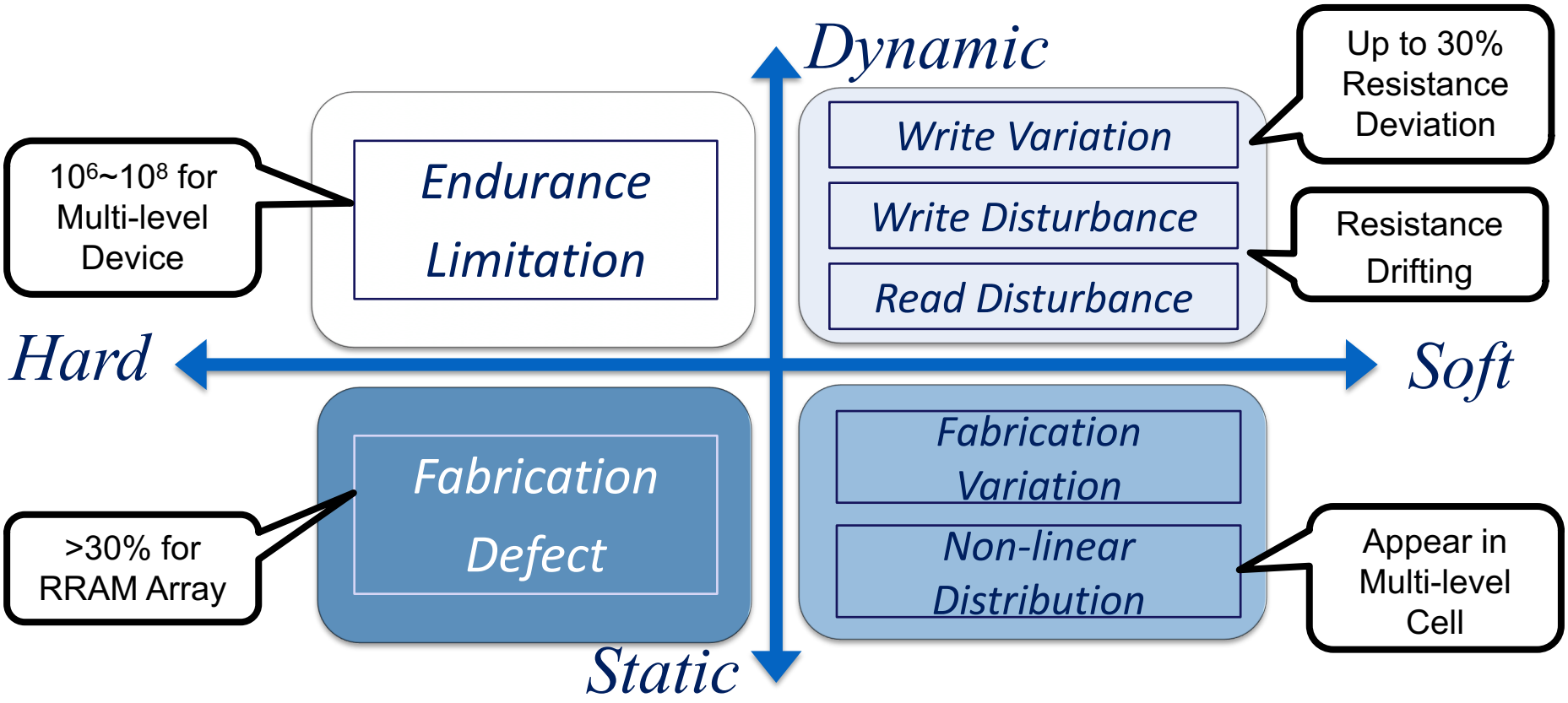


Outline

- Background and Introduction
  - Convolutional Neural Network Accelerators
  - RRAM and RRAM-based Computing System
  - RRAM Fault Models
- **Fault-tolerance for Device-level Faults**
  - **Stuck-At-Fault (SAF)**
  - **Limited Endurance**
  - **State Drifting Problem and Resistance Variation**
  - **Non-linear Resistance Distribution**
- Fault-tolerance for Circuit-level Faults
  - Wire Resistance and IR-drop
- Fault-tolerance for System-level Faults
  - Unbalanced Writing

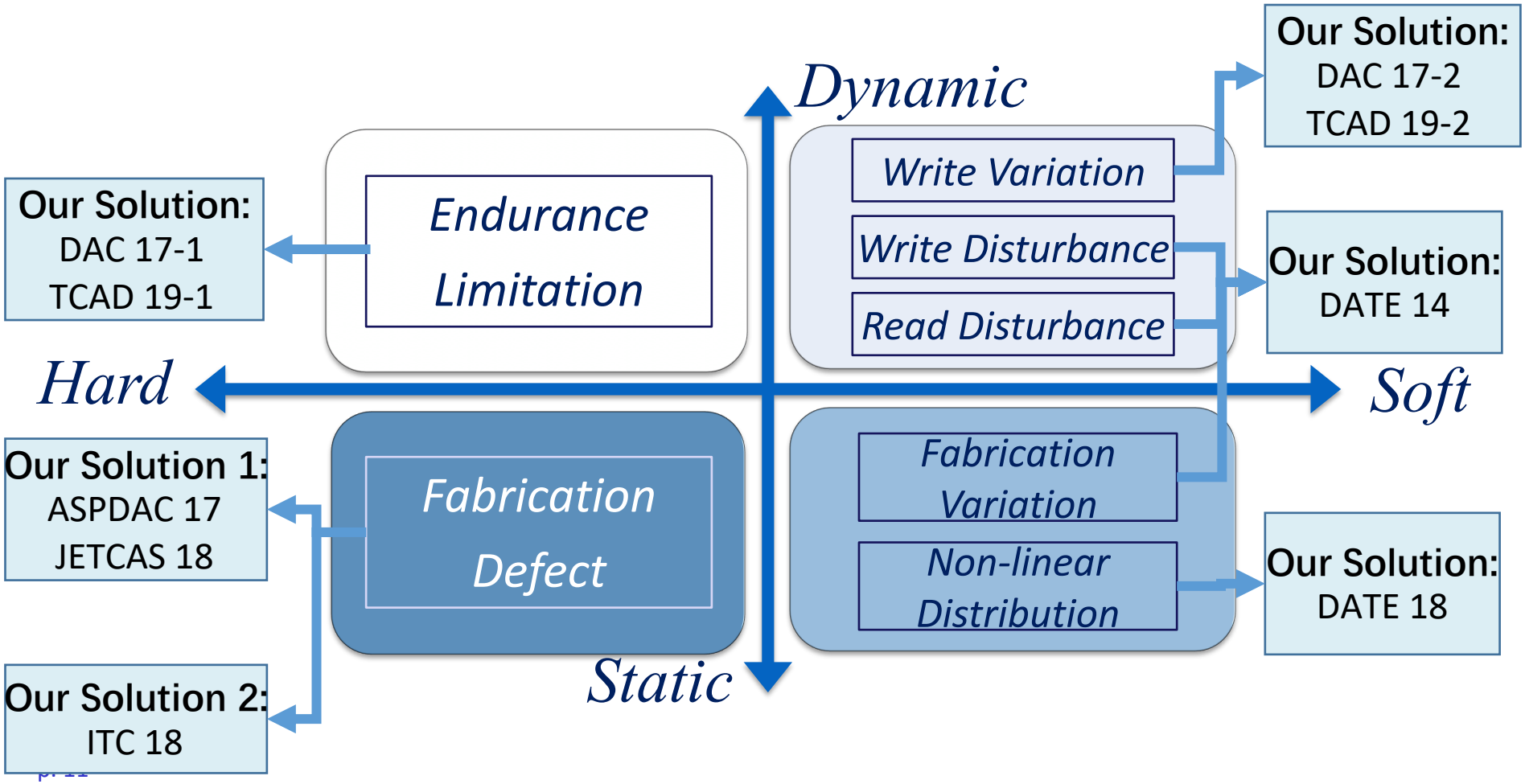
Overview of RRAM Device Faults

- Hard / Soft: Resistance Unchangeable / Changeable
- Static / Dynamic: Generated during Fabrication / Read-and-Write



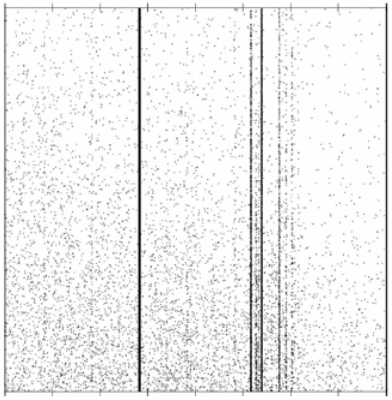
Overview of RRAM Device Faults

- ***Fault-tolerance methods*** for RRAM device faults



Device Fault 1: SAF

- Stuck-At-Faults (SAFs): the resistance states ***cannot be changed***
- SAFs cause significant ***accuracy loss*** of neuromorphic computing
  - The recognition accuracy of the MNIST drops to 17.75% @ 20% SAFs

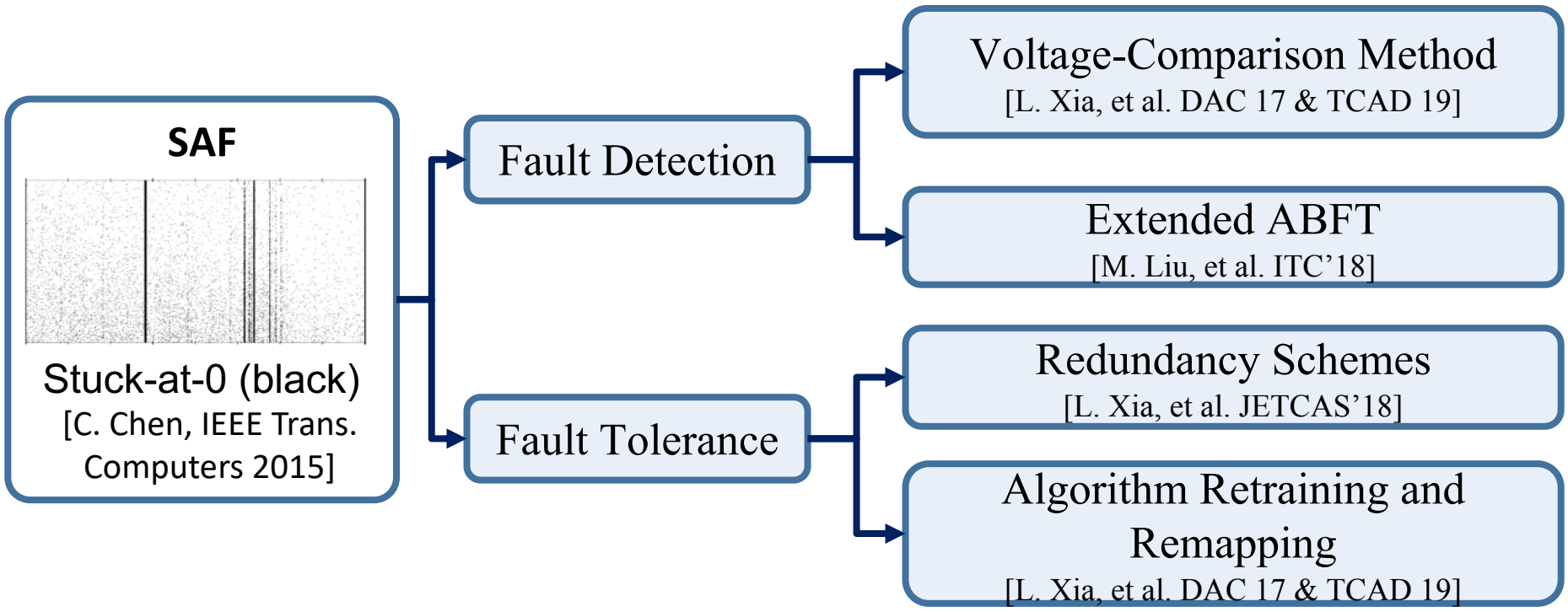


Stuck-at-0 (black)
[C. Chen, IEEE Trans. Computers 2015]

Yield	Ideal	95%	90%	80%
Accuracy	97.8%	26.7~60.4%	15.5~38.6%	10.6~28.0%
Reduction	-	>37%	>59%	>69%

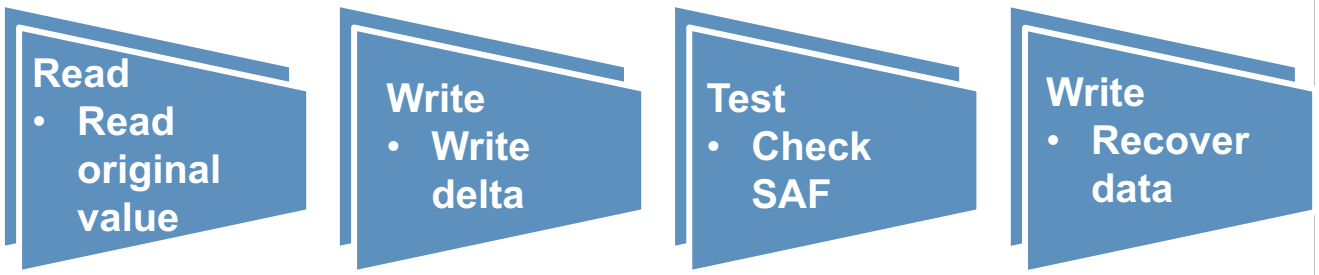
Fault Tolerance Method for SAF

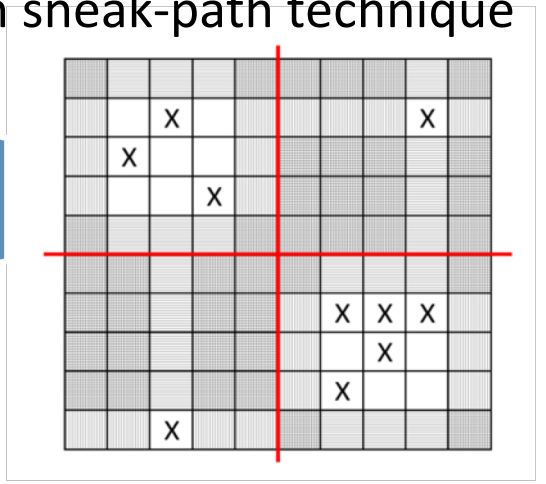
- SAF tolerant framework contains two parts:
  - ***Fault detection***: identify the SAF position
  - ***Fault tolerance***: restore the accuracy



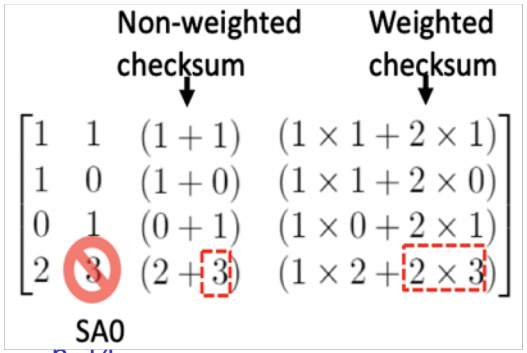
SAF Detection: Identify SAF Position

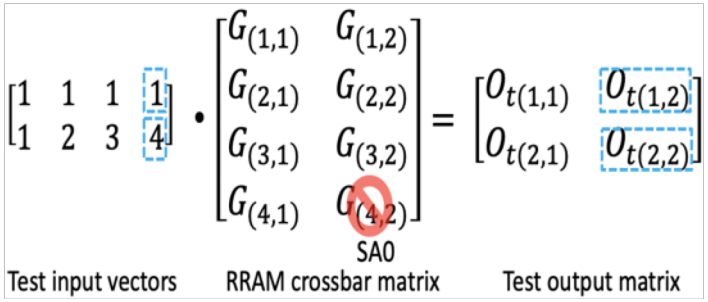
- Voltage comparison method [DAC 2017 & TCAD 2019]
  - Speed up detection by more than **14X** compared with sneak-path technique

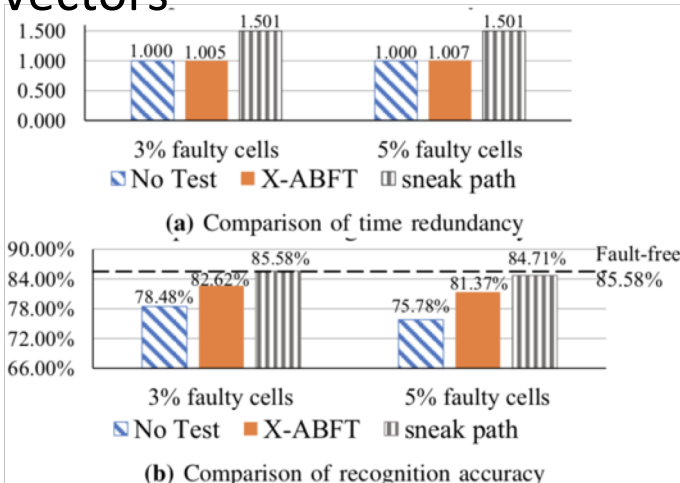




- X-ABFT [ITC 2018]
  - Identify the faulty column: add two checksum columns
  - Identify the faulty row: apply multiple test input vectors

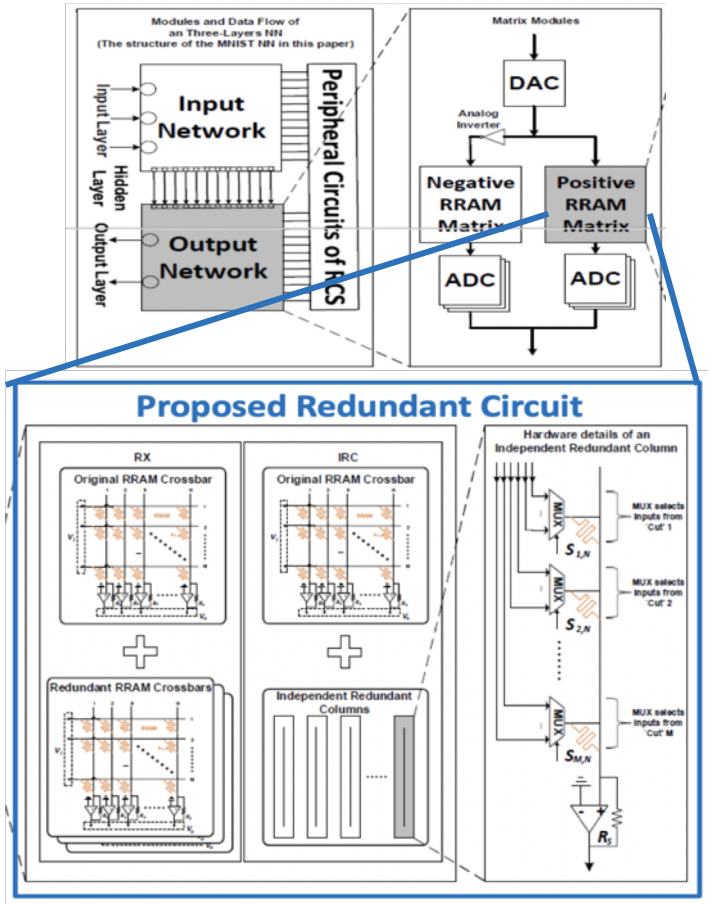


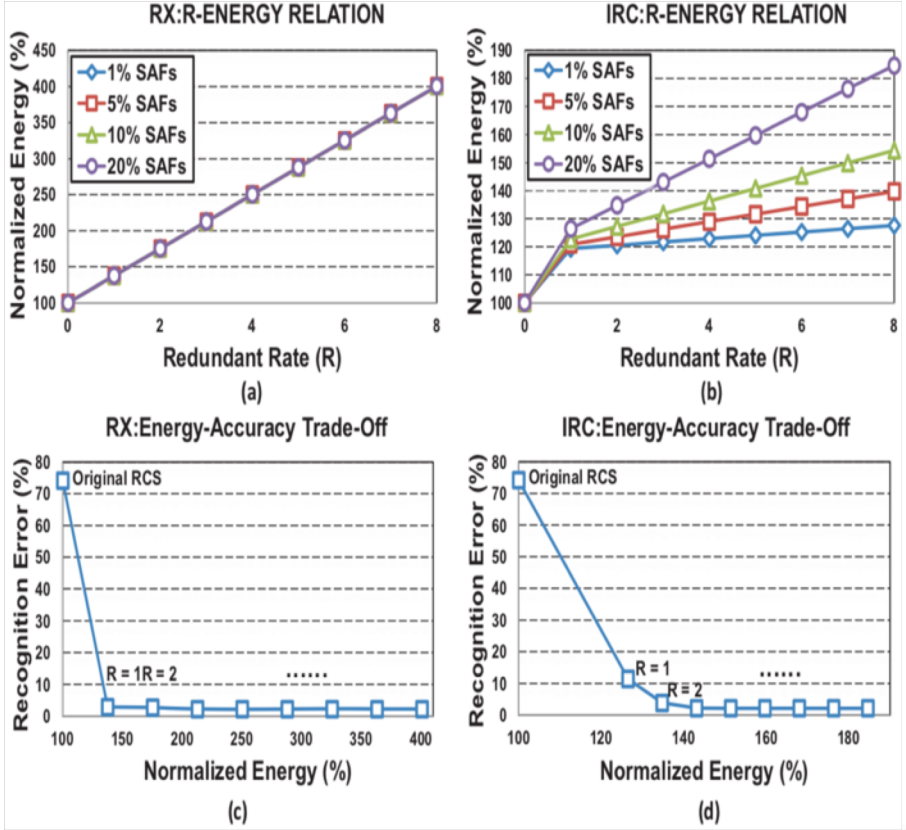




SAF Tolerance: Restore the Accuracy

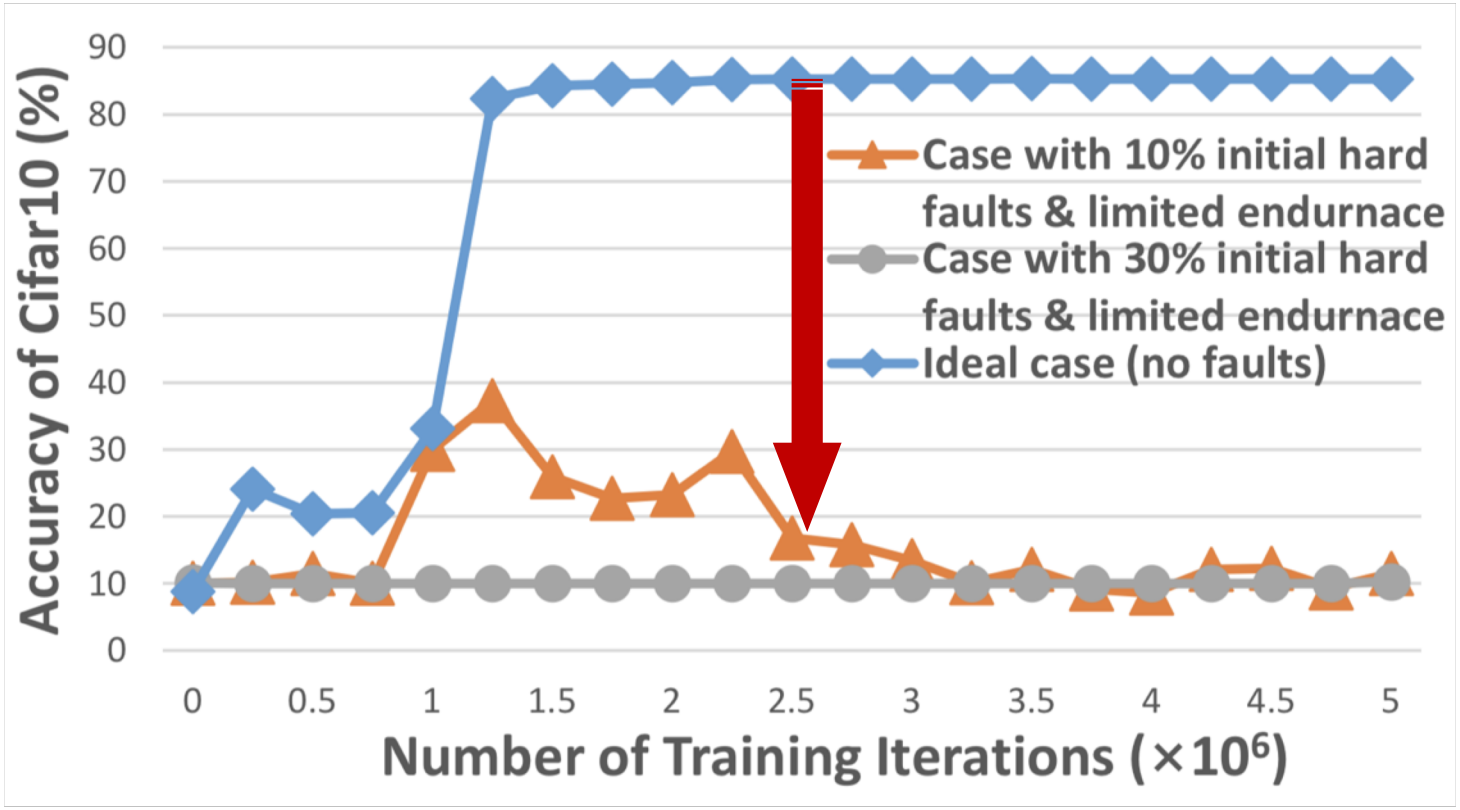
- Computation-oriented ***redundancy scheme*** [ASPDAC 2017 & JETCAS 2018]
  - Mapping algorithm with inner fault-tolerant ability when using redundant crossbars and independent redundant columns
- Improve the accuracy from **25% to 96%** w/ 10% SAF @ MNIST





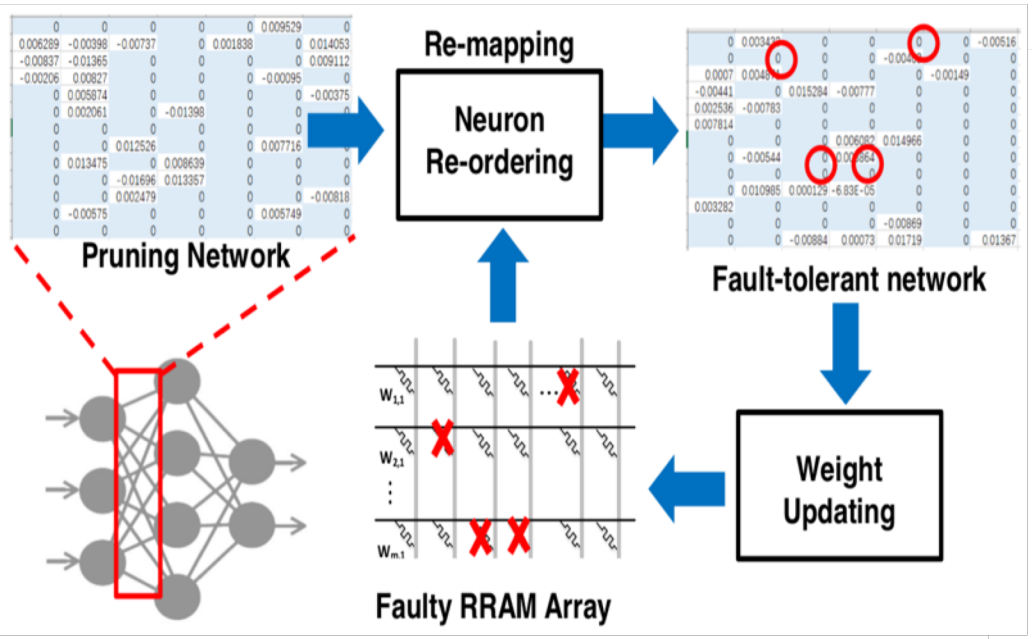
Device Fault 2: Limited Endurance

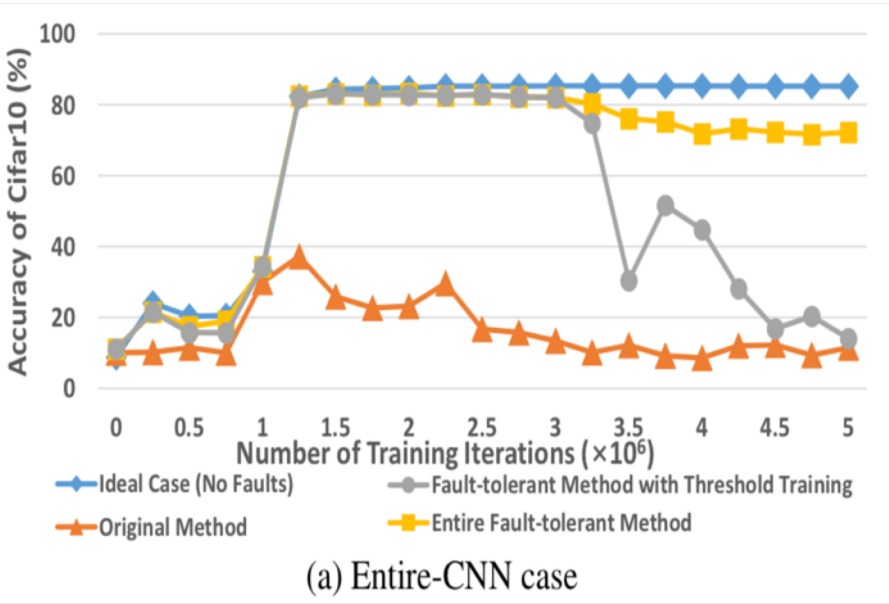
- The write endurance of RRAM is ***limited***
  - The typical endurance of a multi-level RRAM cell ranges from 10^6 to 10^8
  - Writing RRAM over the endurance may lead to SAF
- SAF caused by limited endurance will ***hurt training performance***



Fault Tolerance Method for Limited Endurance

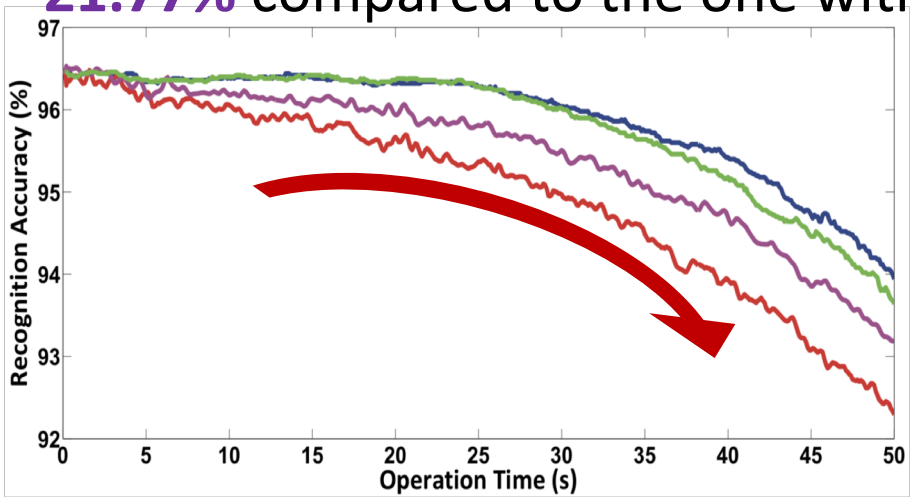
- ***Fault-tolerant training and remapping*** [DAC 2017-1 & TCAD 2019-1]
  - Use a threshold-training method to reduce the write times
  - Use a re-mapping scheme: map pruned network value to Stuck-At-0 cells
- Improve the accuracy from **37% to 83%** @ Cifar-10

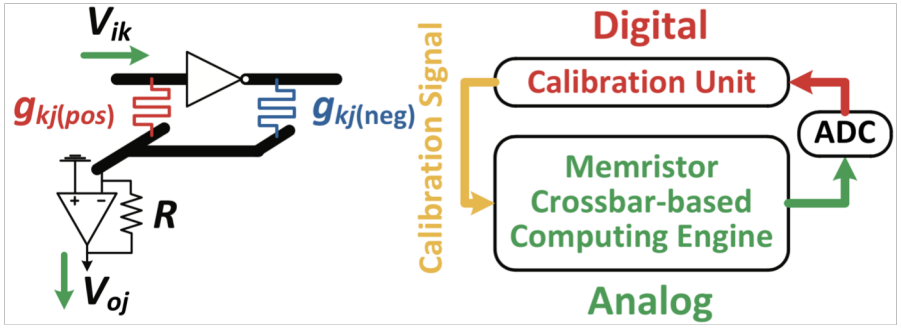




Device Fault 3: State Drifting Problem

- ***State drifting:*** Read operations also change the RRAM resistance slowly
- State drifting causes ***a decline of RCS's performance***
- ICE: ***Inline Calibration*** for Memristor Crossbar-based Computing Engine [DATE 2014]
  - Periodically interrupt-and-benchmark (I&B) RCS
  - Minimize the negative impact of the I&B operation on system performance
- Achieves a calibration efficiency of **91.18%** on average, improving **21.77%** compared to the one with a constant calibration period





Device Fault 4: Resistance Variation

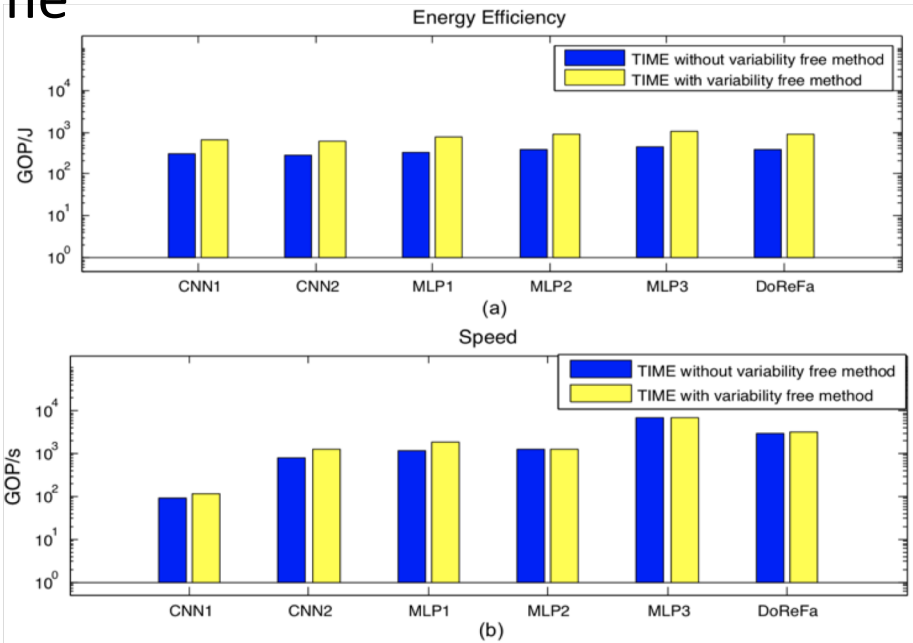
- ***Resistance variation:*** the actual change of RRAM resistance is different from the target $\Delta R_{real} \sim N(\Delta R_{accurate}, 0.09R)$
- Resistance variation makes write operation ***inaccurate***
- ***Variability-free Tuning*** Scheme [DAC 2017-2 & TCAD 2019-2]
  - Use ideal value and variance of RRAM model for tuning w/ 3σ principle
- The energy efficiency is improved **2.28x** on average, **2.29x** at most compared with existing tuning scheme

Algorithm 2: Variability-free tuning scheme

Input: Current resistance of RRAM $R_{current}$, target resistance range R_{target}, target resistance range ϵ, change of resistance δR

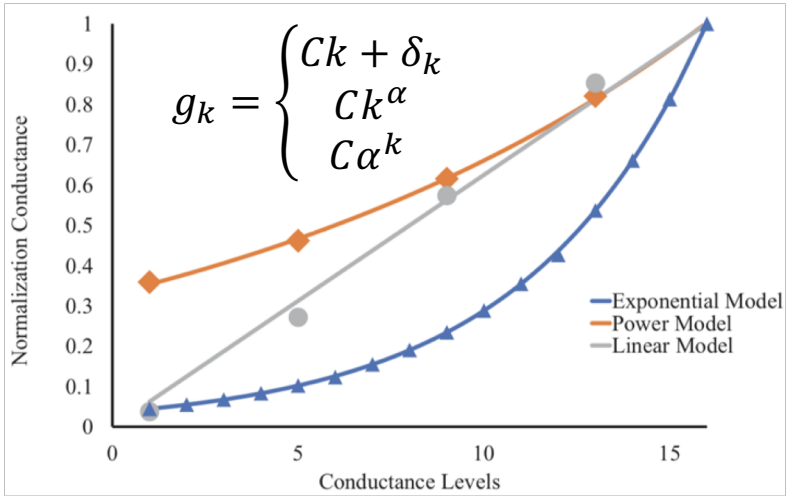
Output: $R_{current}$

```
while ε > abs(R_current − R_target) do
    ΔR_real ~ N(ΔR_accurate, 0.09R)
    Produce the ΔR_real according to N(ΔR_accurate − 3σ, 0.09R)
    Obtain V_pulse by lookup table based on RRAM
    Use V_pulse to change the resistance of RRAM
    obtain the R_current
    ΔR_accurate = R_accurate − R_target
end
return R_current
```

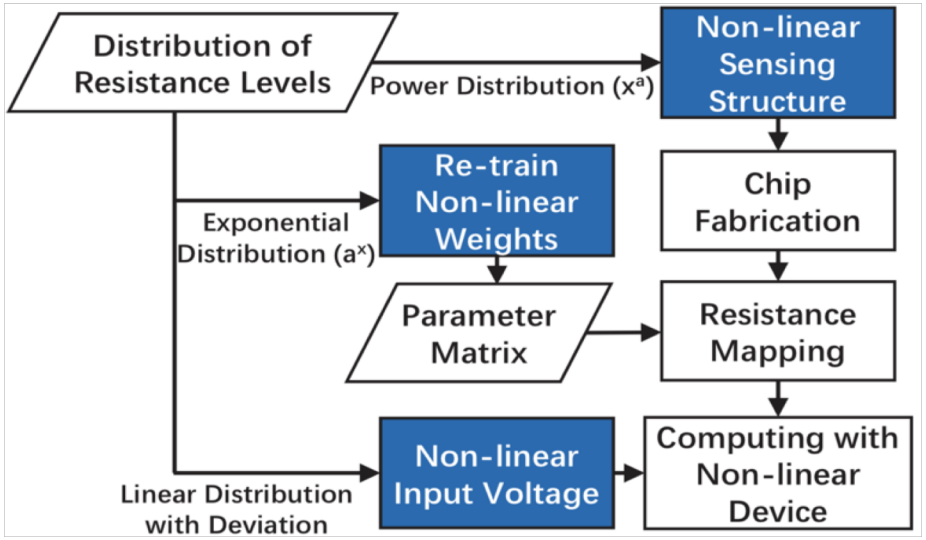


Device Fault 5: Nonlinear Resistance Distribution

- The actual fabricated multi-level devices show the distribution of resistance level is ***not linear***
  - Nonlinear resistance distribution causes ***high computing RMSE***
- Computation ***accuracy recovery*** framework [DATE 2018]
  - Applying non-linear voltage, retraining, and designing new sensing structure
- Devices with non-linear conductance levels can achieve **the same accuracy** as the ideal linear devices, reduce **99%** RMSE



Non-linear resistance distribution



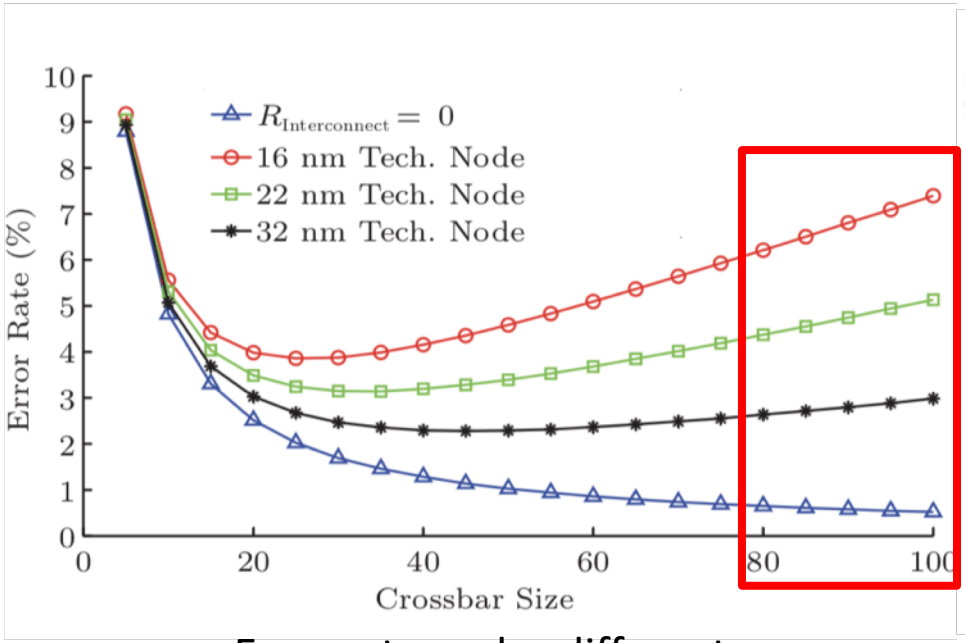
The overall accuracy recovery framework

Outline

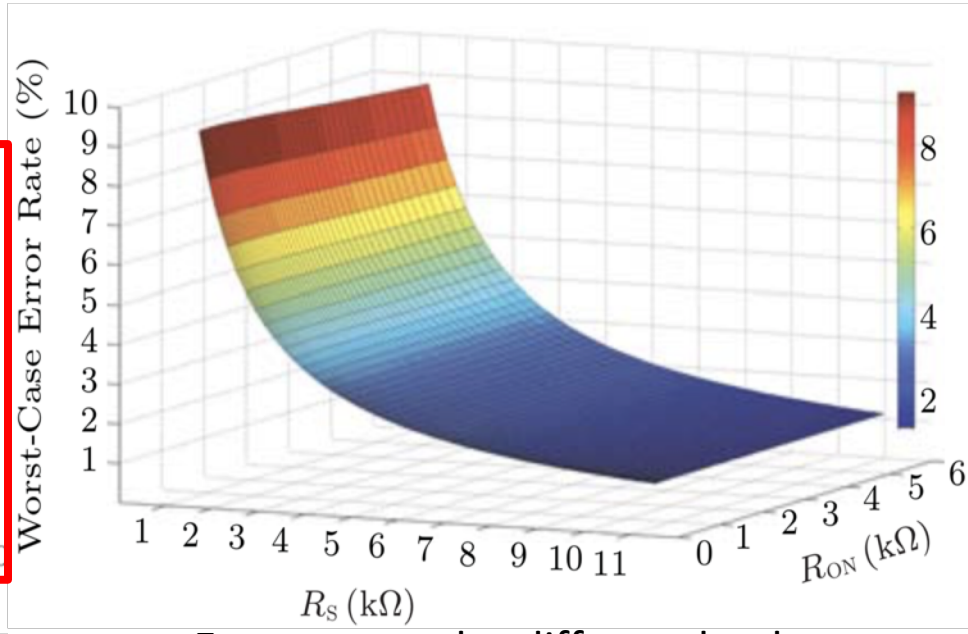
- Background and Introduction
  - Convolutional Neural Network Accelerators
  - RRAM and RRAM-based Computing System
  - RRAM Fault Models
- Fault-tolerance for Device-level Faults
  - Stuck-At-Fault (SAF)
  - Limited Endurance
  - State Drifting Problem and Resistance Variation
  - Non-linear Resistance Distribution
- **Fault-tolerance for Circuit-level Faults**
  - **Wire Resistance and IR-drop**
- Fault-tolerance for System-level Faults
  - Unbalanced Writing

Circuit-level Faults and Non-ideal Factors

- ***Wire resistances*** cause ***IR drop problem***, resulting in accuracy loss in large crossbars
- Choices of ***load resistor*** and ***RRAM resistance range*** also influence the computation accuracy



Error rate under different crossbar sizes



Error rate under different load resistances and RRAM range

Fault Tolerance Method for Circuits Fault

- ***Technological exploration*** of RRAM crossbar to overcome the circuit-level faults and non-ideal factors [ASPDAC 2015 & JCST 2016]
- ***Results of Technological exploration:***
  - Achieve **10.98%** improvement of recognition accuracy on the MNIST dataset and **26.4%** energy savings compared with previous work
  - More than **84.4%** power saving can be achieved at the cost of little accuracy reduction

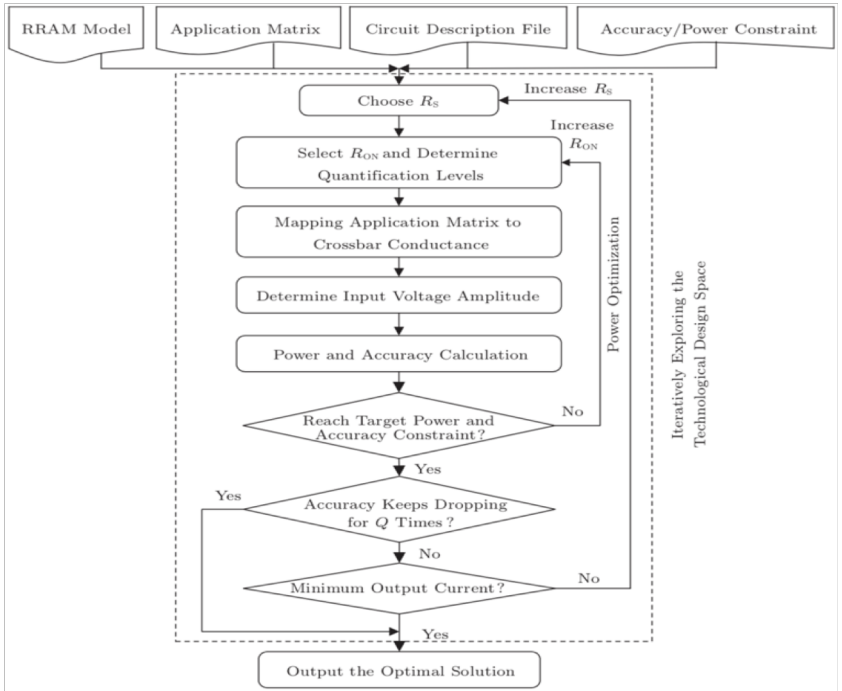


Table 2. Power Saving with a Restricted Accuracy Threshold (Initial $R_{ON} = 500\ \Omega$)

Technology Node (nm)	Accuracy Threshold (%)	Optimal R_{ON} (kΩ)	Initial Power (mW)	Optimal Power (mW)	Power Savings (%)
16	80	17.0	2.10	0.340	83.7
22	80	16.3	2.16	0.347	83.9
28	80	16.0	2.19	0.351	84.0
36	80	16.0	2.26	0.353	84.4
22	85	5.0	2.16	0.611	71.7

Outline

- Background and Introduction
  - Convolutional Neural Network Accelerators
  - RRAM and RRAM-based Computing System
  - RRAM Fault Models
- Fault-tolerance for Device-level Faults
  - Stuck-At-Fault (SAF)
  - Limited Endurance
  - State Drifting Problem and Resistance Variation
  - Non-linear Resistance Distribution
- Fault-tolerance for Circuit-level Faults
  - Wire Resistance and IR-drop
- **Fault-tolerance for System-level Faults**
  - **Unbalanced Writing**

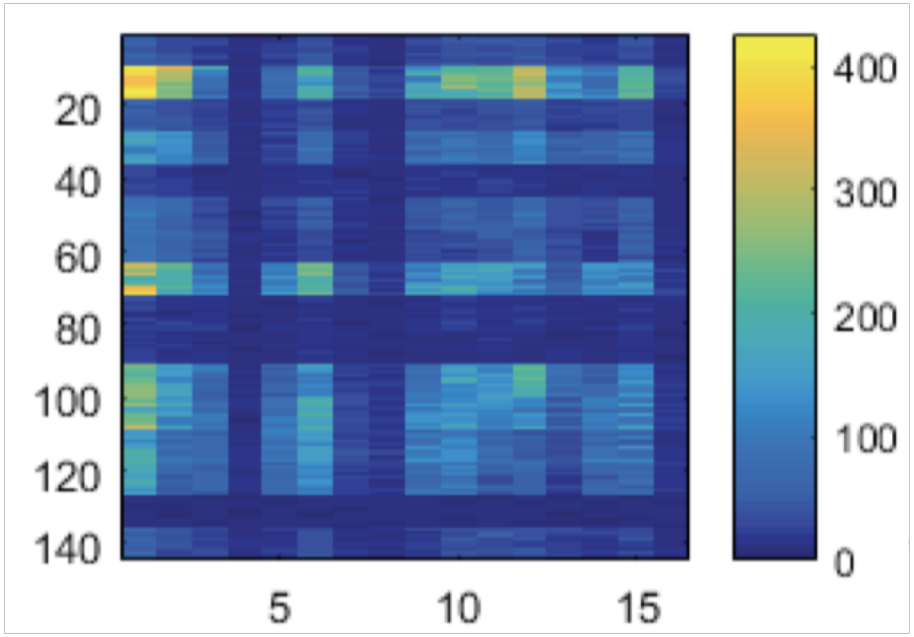
RRAM Endurance Problem in Training

- The lifetime of existing training RCS is ***short***
  - Reason I: RRAM endurance is ***limited*** (e.g., 10^6~10^8), but the number of iteration and RRAM writing in training is ***large***
  - Reason II: The weight update is ***unbalanced***

model	iteration	dataset
LeNet-5	10,000	MNIST
ResNet-20	64,000	CIFAR-10
VGG-11	78,200	CIFAR-10
ResNet-50	500,000	ImageNet

Expected Lifetime (if endurance ~ 5×10^6)

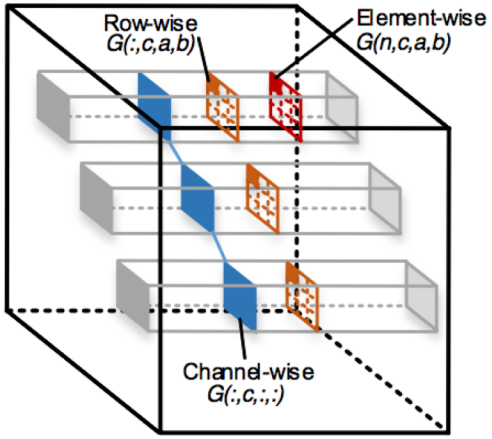
$(5\times10^6)/(5\times10^5) \approx$ 10 times



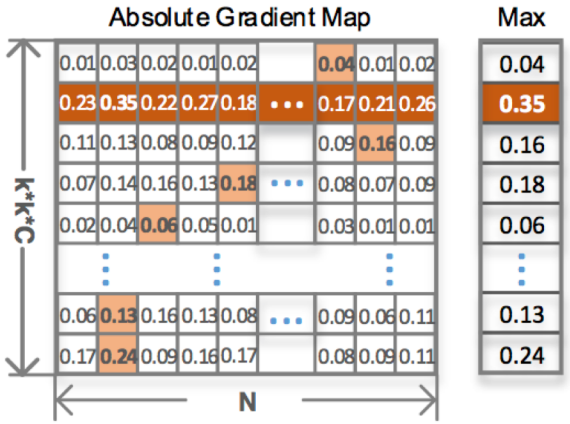
The overall write distribution of RRAM crossbars

Fault Tolerance Method for Training RCS

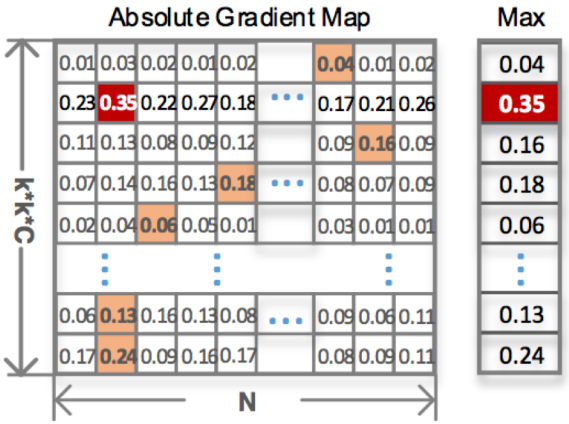
- Long Live TIME: ***improve the training lifetime*** [DAC 2018]
  - SGS: Structured Gradient Sparsification



(a) Structured Gradient Sparsity (SGS)



(b) Row-wise



(c) Element-wise

Two benefits: Structured Weight Updating
Less Sorting Operations

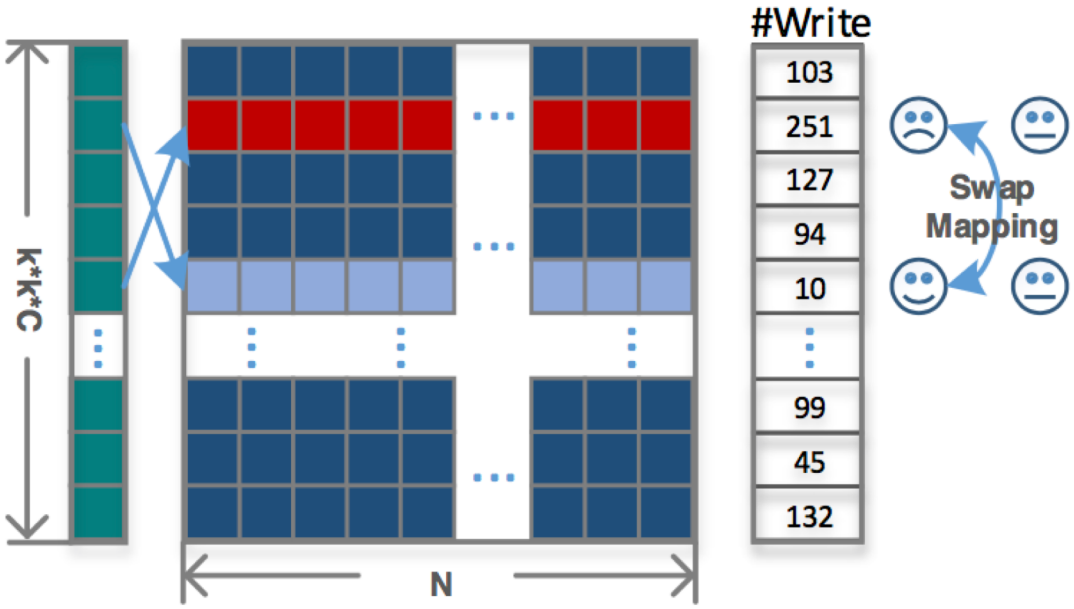
Convolution kernel: $k^2C * N$

Complexity: $k^2C * N + \mathrm{k^2C}$

O(nlogk)->O(n)

Fault Tolerance Method for Training RCS

- Long Live TIME: ***improve the training lifetime*** [DAC 2018]
  - SGS: Structured Gradient Sparsification
  - ARS: Aging-aware Row Swapping



Counter Memory Requirement:

$k^2C * B, B = \log_2 MaxIter$

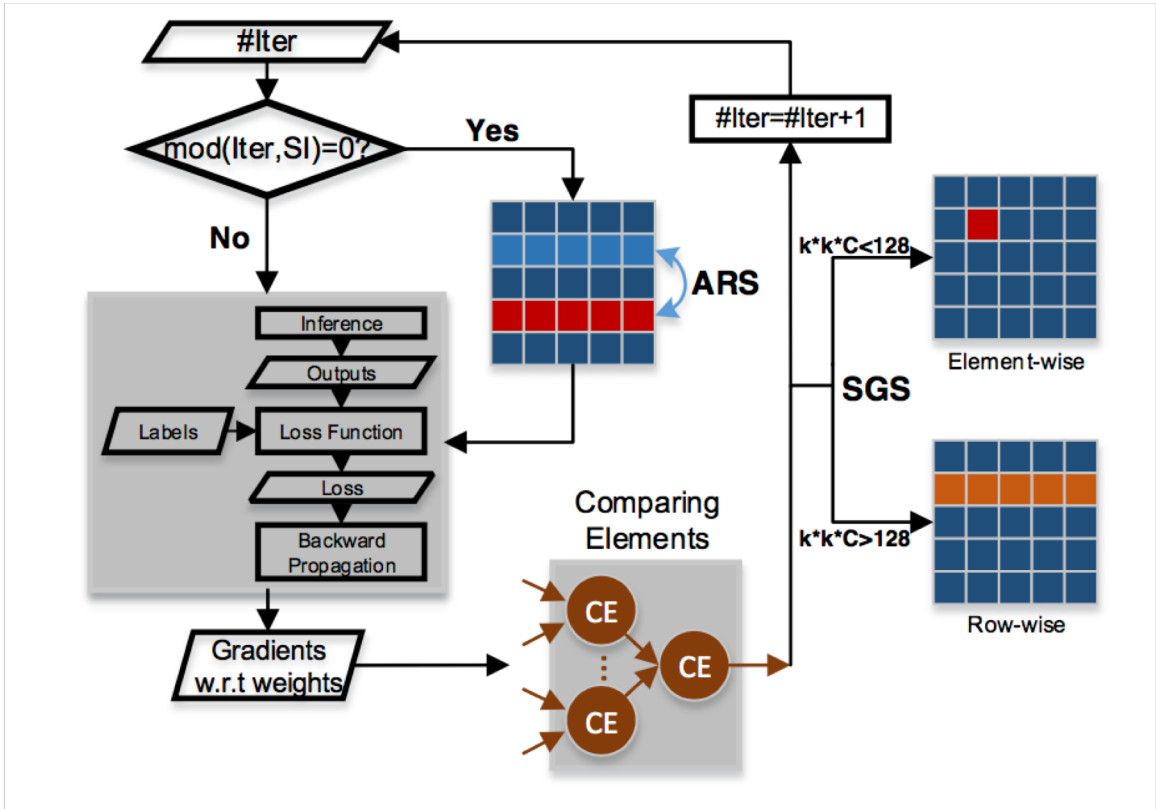
Two hyper-parameters:

ARS Interval: SI

Number of swapped rows: R

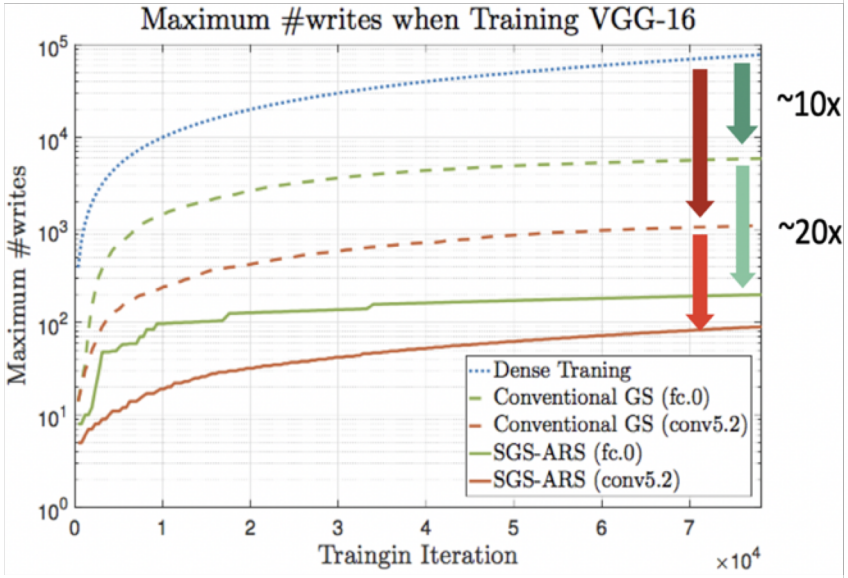
Fault Tolerance Method for Training RCS

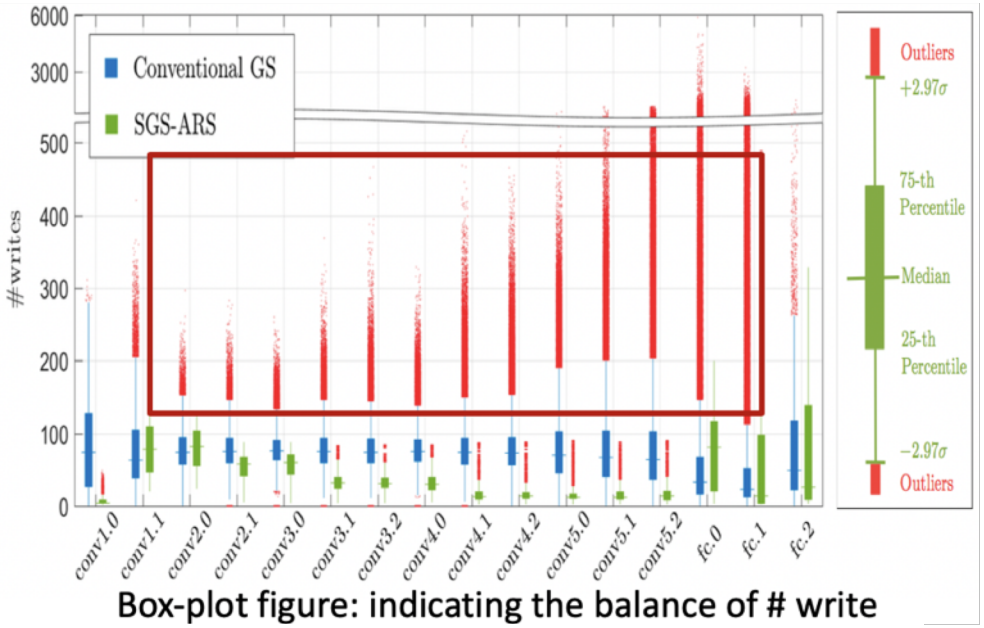
- Long Live TIME: ***improve the training lifetime*** [DAC 2018]
  - SGS: Structured Gradient Sparsification
  - ARS: Aging-aware Row Swapping
  - ***SGS-ARS Training Framework***



Fault Tolerance Method for Training RCS

- ***Results of Long Live TIME*** [DAC 2018]
  - Achieve **356x** longer lifetime in the training task of ResNet-50 on ImageNet





Box-plot figure: indicating the balance of # write

Model	Dataset	Classification Accuracy				Sparsity of SGS	#Writes		Lifetime Extension	
		SGS		Baseline						
		Top-1	Top-5	Top-1	Top-5		SGS-ARS	Baseline	SGS-ARS	FT-Train [17]
VGG-16	Cifar10	91.0% (-1.5%)	-	92.5%	-	99.7%	489	78200	160×	15×
ResNet-20	Cifar10	91.7% (+0.0%)	-	91.7%	-	99.9%	316	64124	177×	-
ResNet-50	ImageNet	75.1% (-1.0%)	92.4% (-0.5%)	76.1%	92.9%	99.8%	1264	450450	356×	-

Conclusions

- An RRAM-based computing system (RCS) provides a promising solution for neuromorphic computing
- RCS is vulnerable to faults
  - RCS contains 3-level faults: device, circuit, and system
  - Testing and fault-tolerant designs are important for RCS
  - Promising solutions have recently developed for testing and fault tolerance in RCS
- Next steps: Better understanding of the physics of defects and the impact of faults on circuit operation

Our Related Work

[1] B. Li, et al, ICE: inline calibration for memristor crossbar-based computing engine, in Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014, pp.1-4.
[2] P. Gu, et al, Technological Exploration of RRAM Crossbar Array For Matrix-Vector Multiplication, in Proceedings of the 20th Asia and South Pacific Design Automation Conference (ASP-DAC), 2015, pp.106-111.
[3] L. Xia, et al, Technological Exploration of RRAM Crossbar Array for Matrix-Vector Multiplication, in Journal of Computer Science and Technology (JCST), vol.31, No.1, 2016, pp.3-19.
[4] W. Huangfu, et al, Computation-Oriented Fault-Tolerance Schemes for RRAM Computing Systems , in Proceedings of the 22nd Asia and South Pacific Design Automation Conference (ASP-DAC), 2017.
[5] L. Xia, et al, Fault-Tolerant Training with On-Line Fault Detection for RRAM-Based Neural Computing Systems, in Design Automation Conference (DAC), 2017.
[6] M. Cheng, et al, TIME:A Training-in-memory Architecture for Memristor-based Deep Neural Network , in Design Automation Conference (DAC), 2017, pp.26:1-26:6.

Our Related Work

[7] M. Liu, et al, Fault Tolerance for RRAM-Based Matrix Operations , to appear in International Test Conference (ITC), 2018.
[8] J. Lin, et al, Rescuing Memristor-based Computing with Non-linear Resistance Levels, in Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE), 2018.
[9] Y. Cai, et al, Long Live TIME: Improving Lifetime for Training-In-Memory Engines by Structured Gradient Sparsification, in Design Automation Conference (DAC), 2018.
[10] L. Xia, et al, Stuck-at Fault Tolerance in RRAM Computing Systems , in IEEE Journal on Emerging and Selected Topics in Circuits and Systems (JETCAS), vol.8, No.1, 2018, pp.102-115.
[11] M. Cheng, et al, TIME: A Training-in-memory Architecture for RRAM-based Deep Neural Networks , to appear in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2019.
[12] L. Xia, et al, Fault-Tolerant Training Enabled by On-Line Fault Detection for RRAM-Based Neural Computing System , to appear in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2019.

Thanks for your attention