Preliminary Oral Examination for Master Degree

Fault Tolerant Techniques on CGRA

April 22nd, 2013

Jihoon Kang

Dependable Computing Lab
Department of Computer Science
Yonsei University

Committee

Kyoungwoo Lee

Bernd Burgstaller

Yosub Han

Contents

- Motivation
- Related works
- Problem definition
- Thesis proposal
  - Selective validation for efficient protection
- Experiments
- Conclusions

1. Motivation

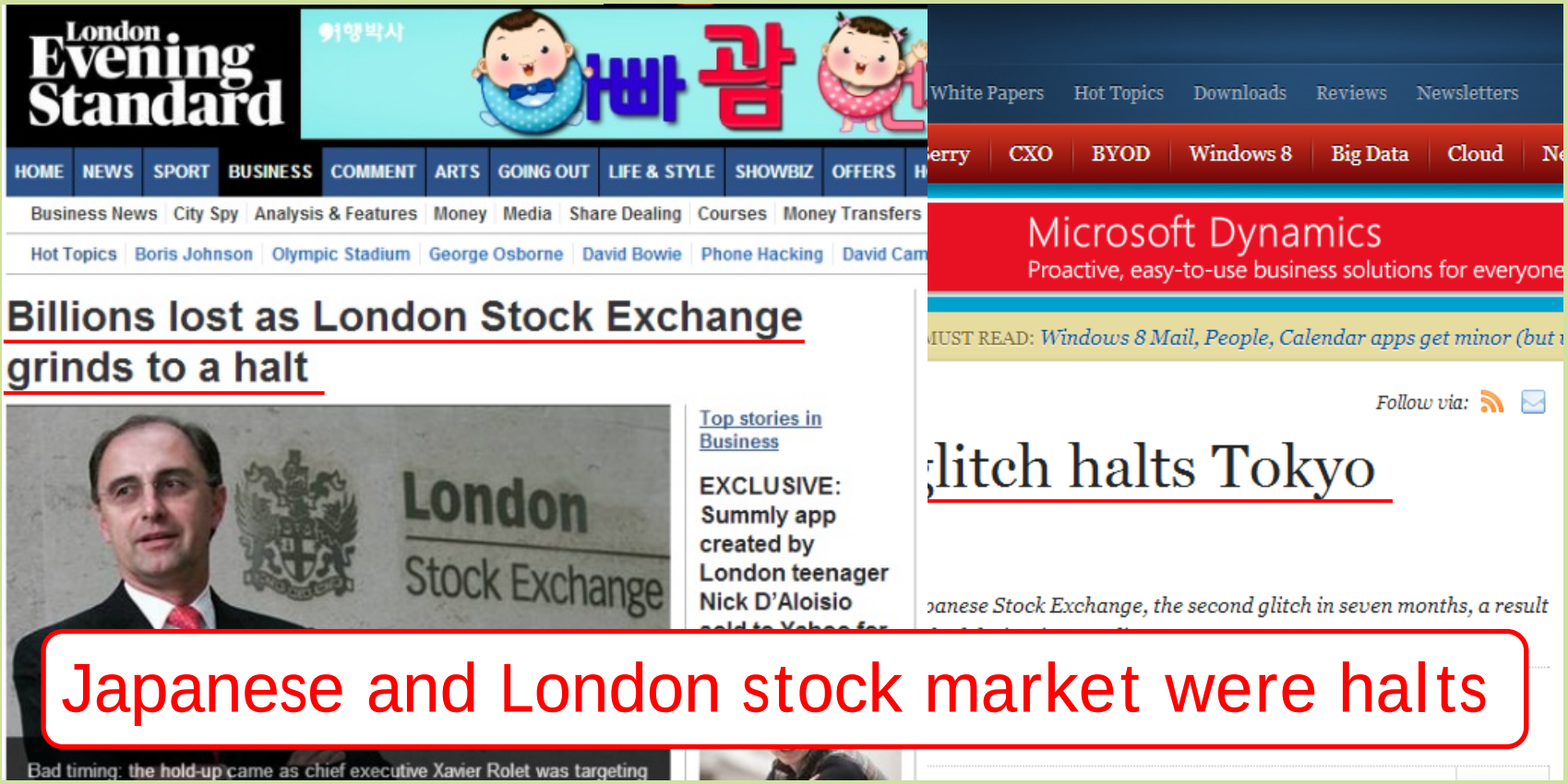
Reliability is important

- Soft errors can cause significant financial loss

Computer system doesn't work properly because of soft error

Reliability is important

- Soft errors can cause significant financial loss



Reliability is important

- Soft errors can cause significant financial loss

Reliability is important

- Reliability is related to our life

Are Cosmic Rays a Factor in Toyota Acceleration Problems?

Wednesday, March 17, 2010 by Roger D. "Skip" Slates

As Toyota continues to grapple with the unintended acceleration-related crisis, existing theories about the reasons for the acceleration are giving way to seemingly far-fetched new ideas. According to some news reports, federal regulators are studying the possibility that cosmic rays are responsible for Toyota vehicles suddenly accelerating to high speeds.

First the facts. The effect of cosmic rays on electronics has been known at least since the 1950s. Researchers have known that r
spacecraft. In fact, aircraft are d
radiation.

In the 1970s, researchers also
and interfere with cell phones a
crashes, and malfunctioning of
automobiles has never been st

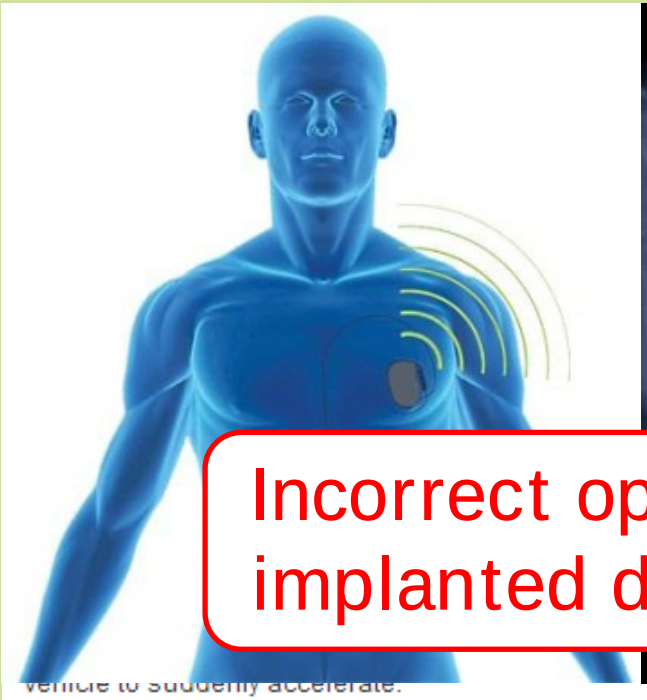
That might change as Californi
find more answers for the sudden acceleration in these v
electronic throttle control systems. In fact, Toyota vehicles
Toyota vehicles included in the recalls come with micropr
now being thrown about is that cosmic radiation could int
vehicle to suddenly accelerate.

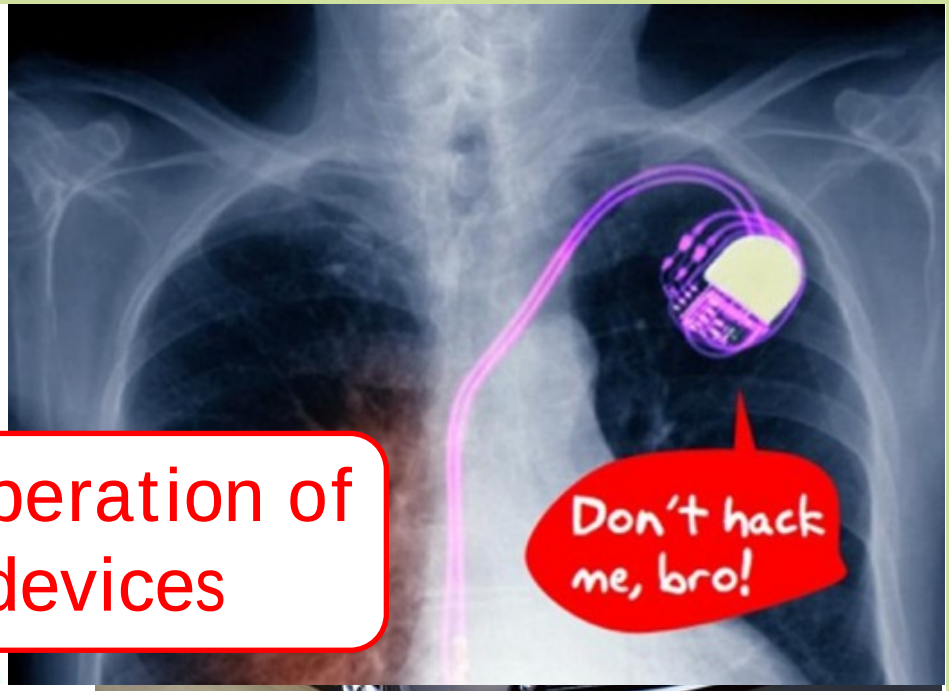
According to radiation testing experts, considering the nu
possible that these automobiles are at risk of interfere
electronics by suddenly flipping a bit from a zero to a o
electronics get smaller, voltages become lower and de

Unintended acceleration

Reliability is important

- Reliability is related to our life

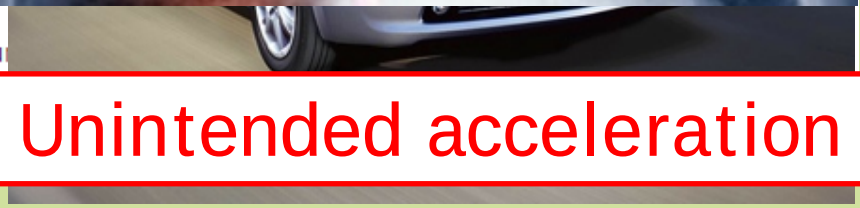




Incorrect operation of implanted devices

vehicle to suddenly accelerate.

According to radiation testing experts, considering the nu
possible that these automobiles are at risk of interfere
electronics by suddenly flipping a bit from a zero to a o
electronics get smaller, voltages become lower and de



Unintended acceleration

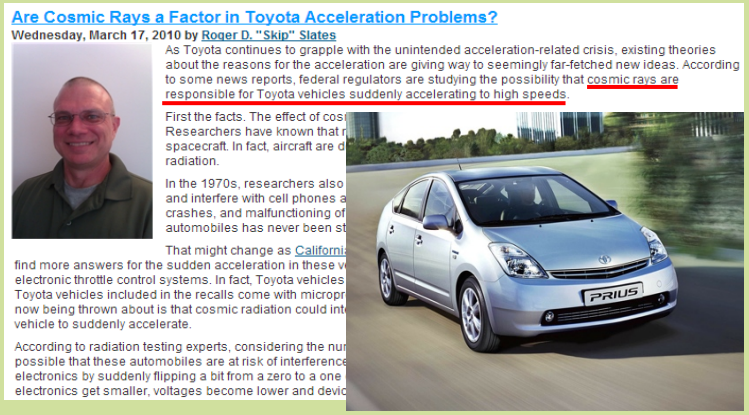
Reliability is important

- Robot malfunctions due to lots of radiation

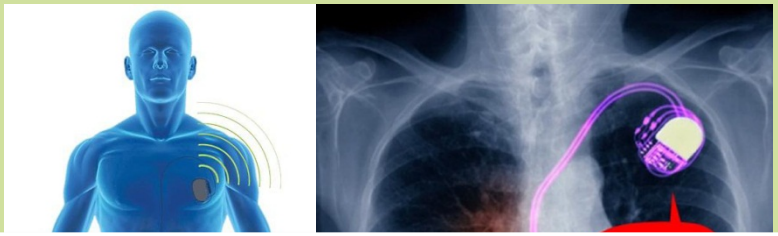
Reliability is important

- Soft error damaged financial losing

- Reliability is directly connected our life



- Reliability is directly connected our life

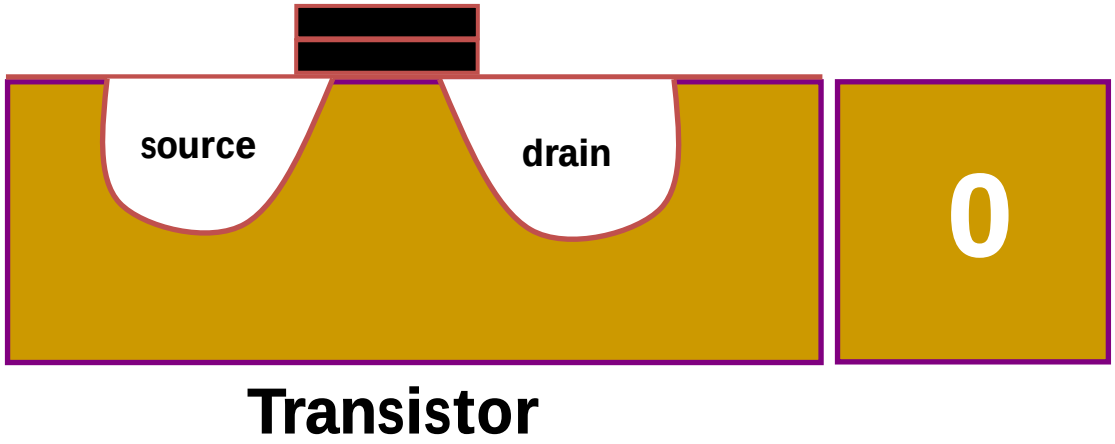


- Vulnerable reliability

Reliability is an important concern

Soft error – an increasing concern

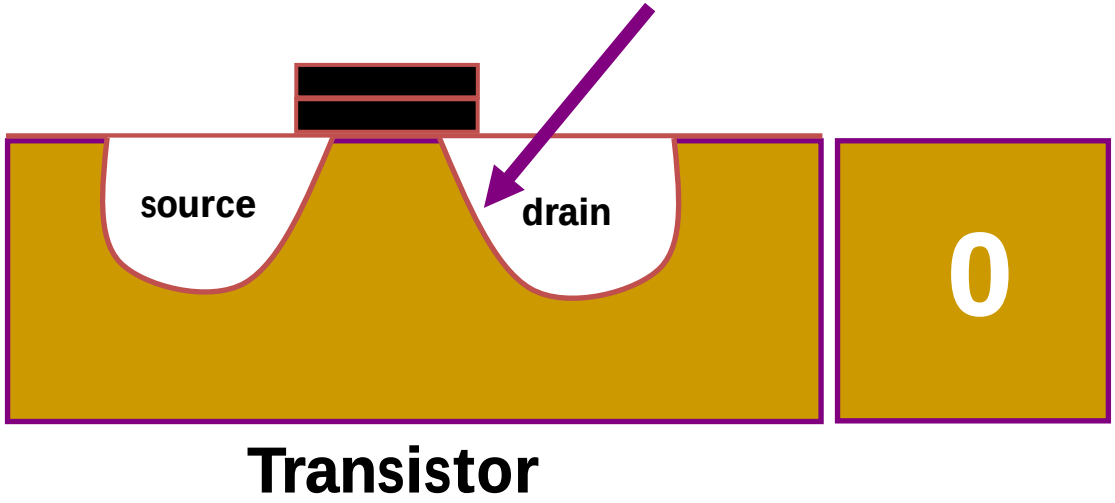
- **Temporary bit flip in a semiconductor device**



D. Shivakumar et al. “Modeling the effect of technology trends on the soft error rate of combinational logic” 2002 DSN

Soft error – an increasing concern

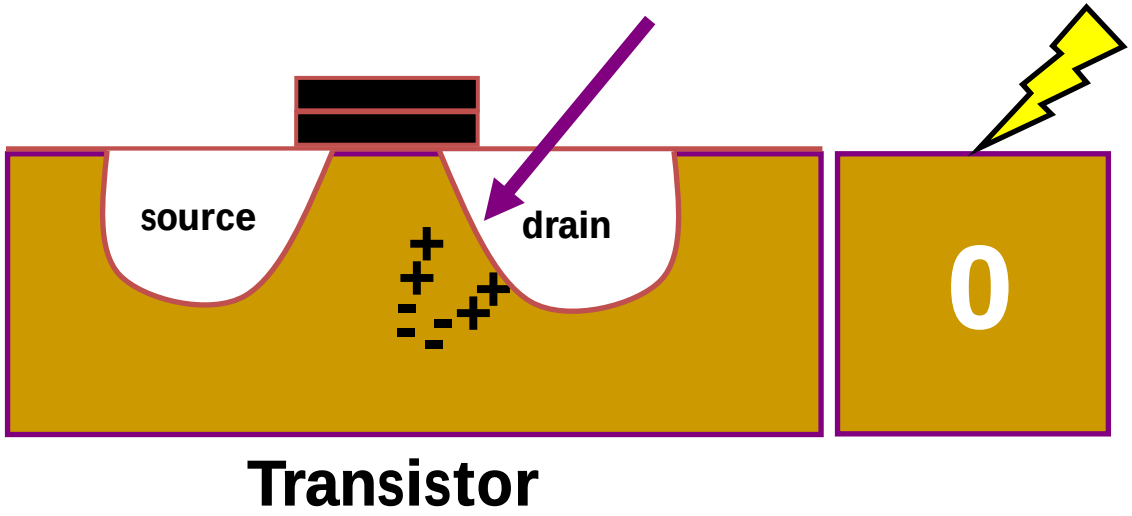
- **Temporary bit flip in a semiconductor device**



D. Shivakumar et al. “Modeling the effect of technology trends on the soft error rate of combinational logic” 2002 DSN

Soft error – an increasing concern

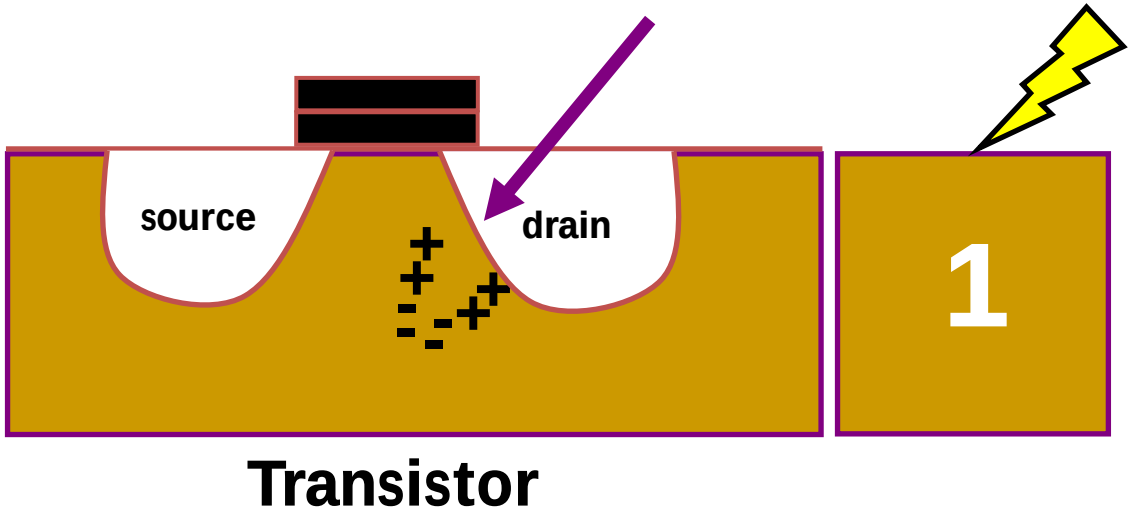
- **Temporary bit flip in a semiconductor device**



D. Shivakumar et al. “Modeling the effect of technology trends on the soft error rate of combinational logic” 2002 DSN

Soft error – an increasing concern

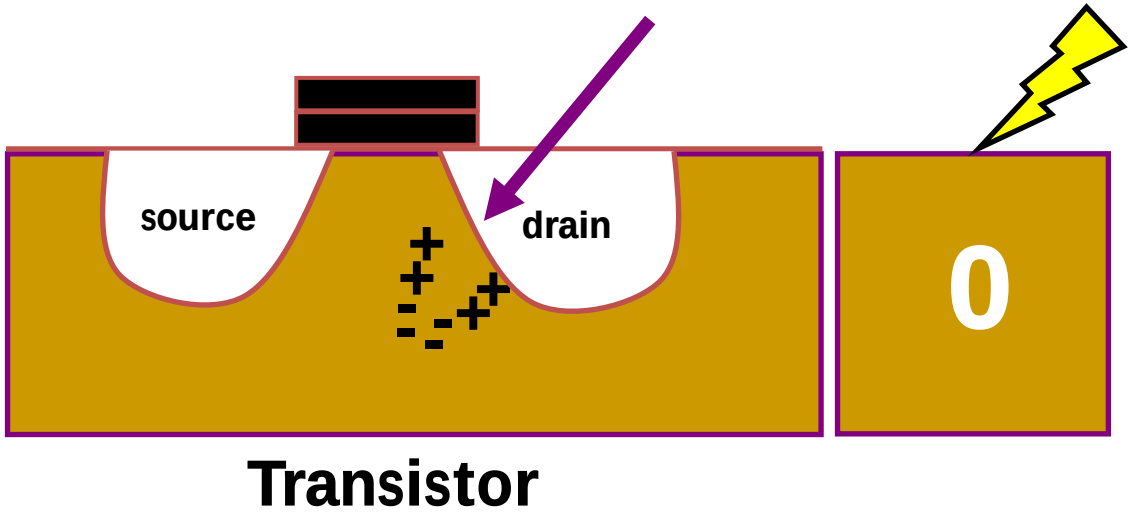
- **Temporary bit flip in a semiconductor device**



D. Shivakumar et al. “Modeling the effect of technology trends on the soft error rate of combinational logic” 2002 DSN

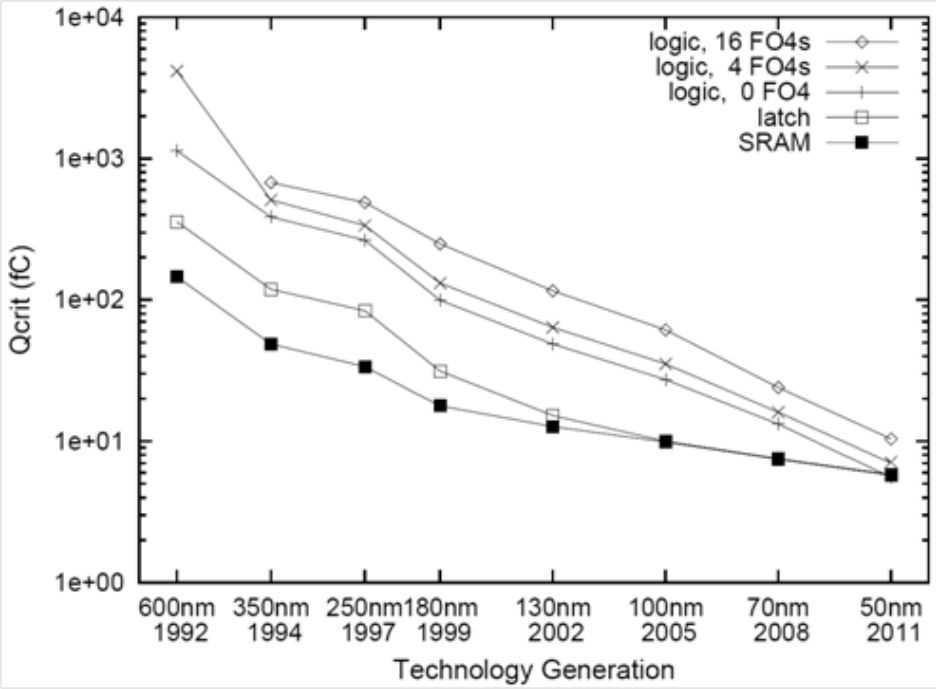
Soft error – an increasing concern

- **Temporary bit flip in a semiconductor device**



D. Shivakumar et al. “Modeling the effect of technology trends on the soft error rate of combinational logic” 2002 DSN

Soft error – an increasing concern



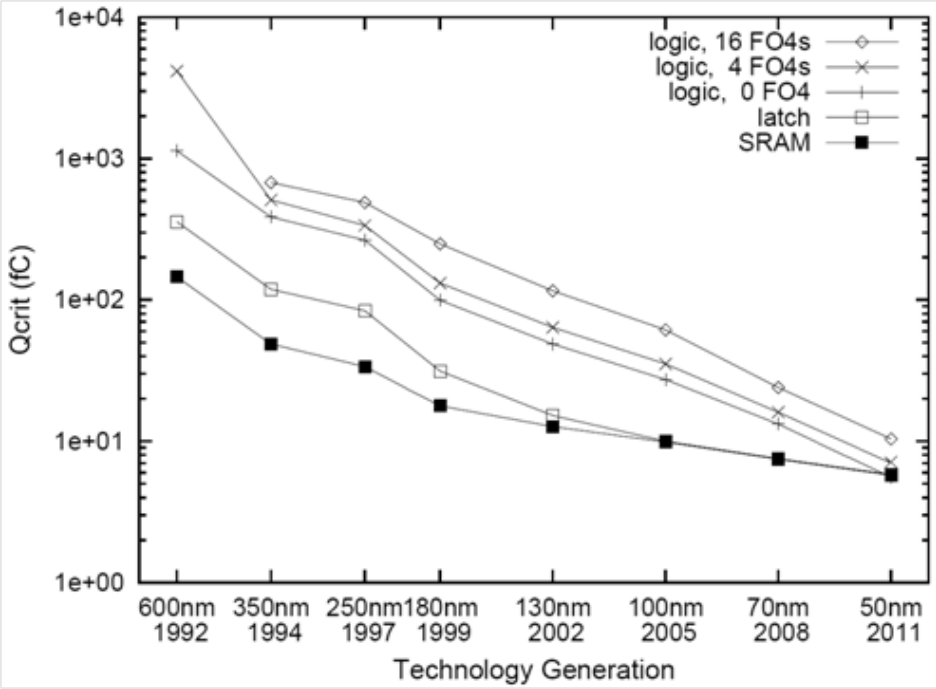
$$SER \propto N_{flux} \times CS \times exp\{-\frac{Q_{critical}}{Q_s}\}$$

where $Q_{critical}$ = Capacitance x Voltage

- Soft error rate
  - Is now 1 per year
  - Exponentially increases with technology scaling

D. Shivakumar et al. "Modeling the effect of technology trends on the soft error rate of combinational logic" 2002 DSN

Soft error – an increasing concern



$$SER \propto N_{flux} \times CS \times exp\{-\frac{Q_{critical}}{Q_s}\}$$

where $Q_{critical}$ = Capacitance x Voltage

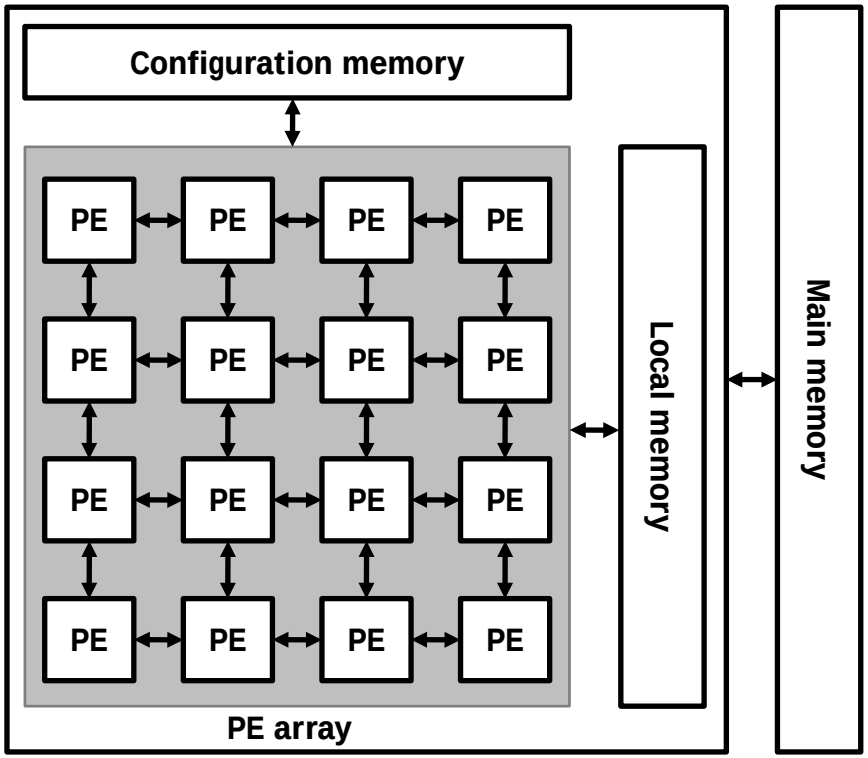
- Soft error rate
  - Is now 1 per year
  - Exponentially increases with technology scaling

Soft errors are becoming a critical design concern

D. Shivakumar et al. "Modeling the effect of technology trends on the soft error rate of combinational logic" 2002 DSN

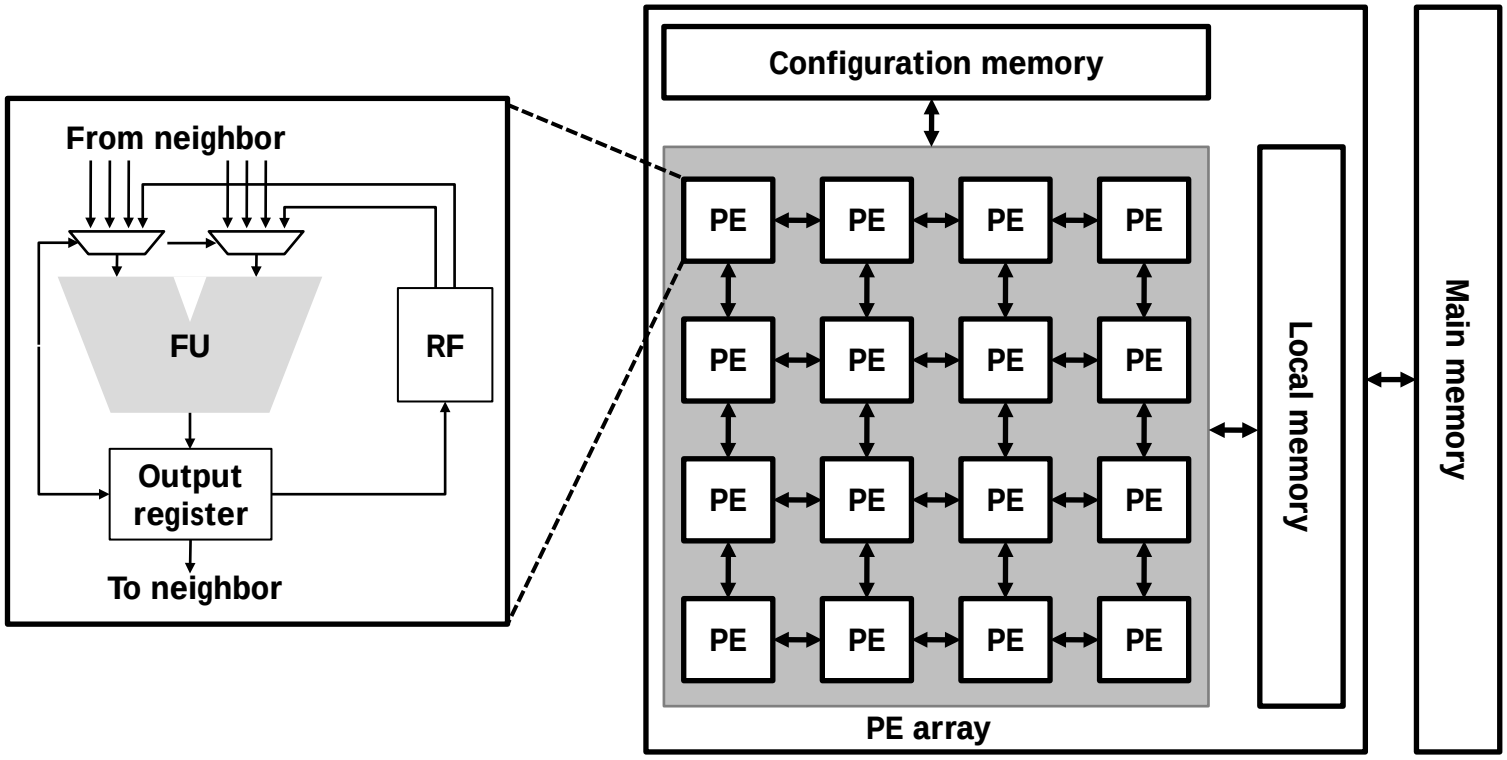
CGRA Architecture

- Emerging architecture
  - High performance, flexibility, low power

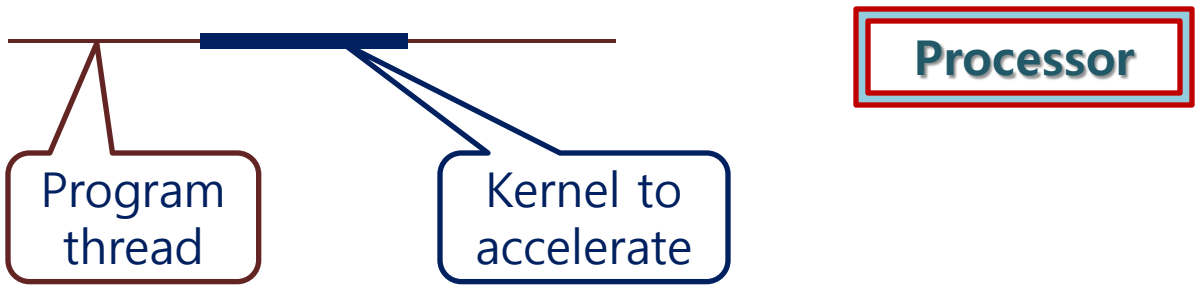


CGRA Architecture

- Emerging architecture
  - High performance, flexibility, low power



Advantages of CGRA



- Specific kernels in a thread can be power/performance critical

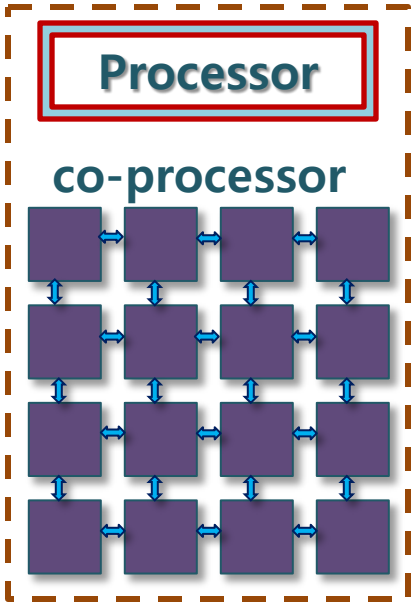
A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

Advantages of CGRA

- Specific kernels in a thread can be power/performance critical

A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

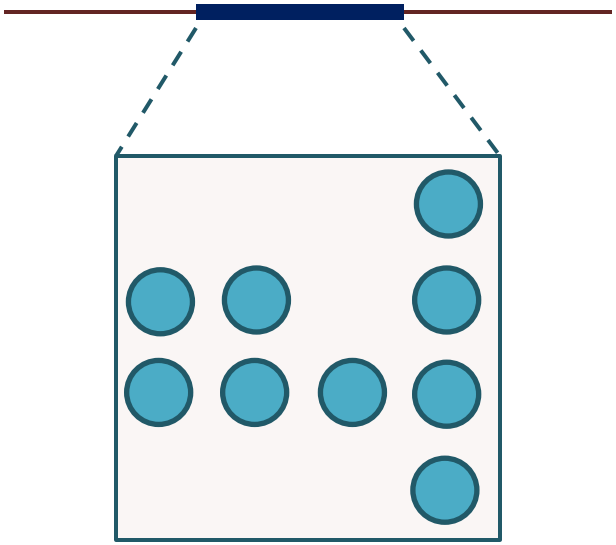
Advantages of CGRA

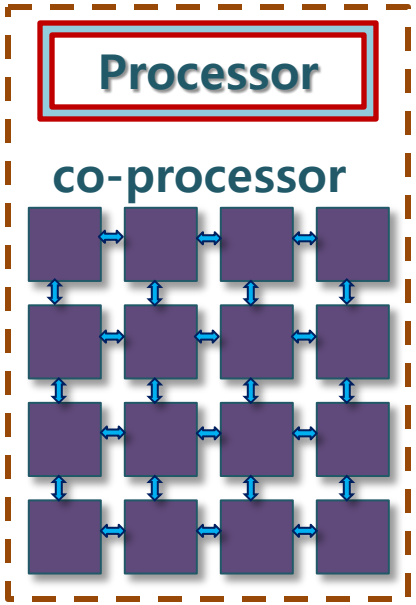


- Specific kernels in a thread can be power/performance critical

A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

Advantages of CGRA

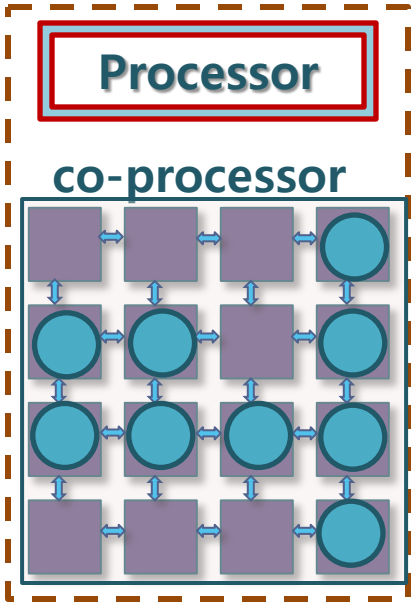




- Specific kernels in a thread can be power/performance critical
- The kernel can be mapped and scheduled for execution on the CGRA
- Using the CGRA as a co-processor (accelerator)

A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

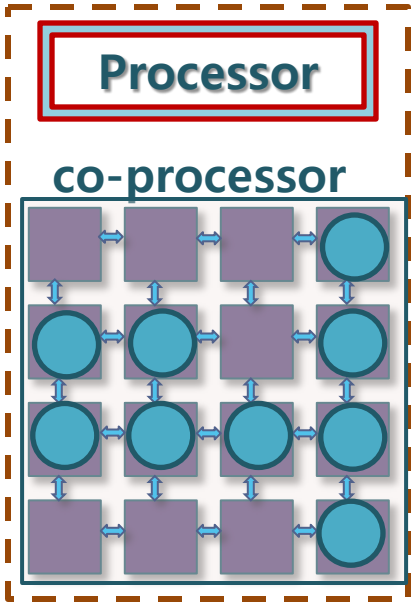
Advantages of CGRA



- Specific kernels in a thread can be power/performance critical
- The kernel can be mapped and scheduled for execution on the CGRA
- Using the CGRA as a co-processor (accelerator)

A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

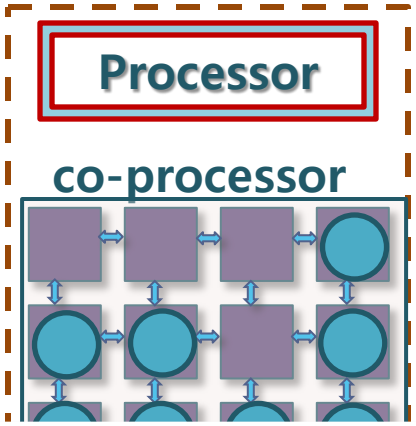
Advantages of CGRA



- Specific kernels in a thread can be power/performance critical
- The kernel can be mapped and scheduled for execution on the CGRA
- Using the CGRA as a co-processor (accelerator)
  - Power consuming processor execution is saved
  - Better performance of thread is realized
  - Overall throughput is increased

A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

Advantages of CGRA



CGRA becomes more and more popular thanks to high performance, flexibility and low power

- Specific kernels in a thread can be power/performance critical
- The kernel can be mapped and scheduled for execution on the CGRA
- Using the CGRA as a co-processor (accelerator)
  - Power consuming processor execution is saved
  - Better performance of thread is realized
  - Overall throughput is increased

A. Shrivastava et al. “Enabling Multithreading on CGRAs” 2011 ICPP

Popular CGRA usage for reliability concern

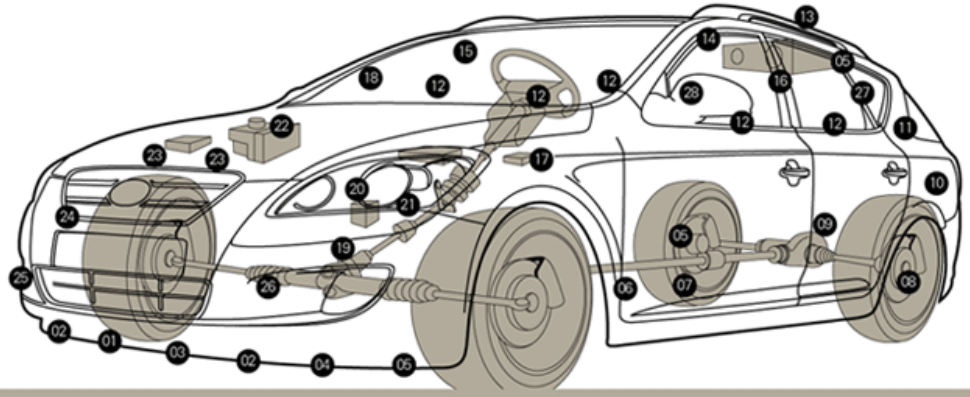
- **CGRA can be used widely close to humans**
  - **Reliability in CGRA is becoming an important issue**

https://mocana.com/blog/tag/medical-device/; www.sustainabletechnolog.com; www.kia.co.nz; www.woodward.com;

Popular CGRA usage for reliability concern

- **CGRA can be used widely close to humans**
  - **Reliability in CGRA is becoming an important issue**

Advanced Safety Vehicle



1. Road condition sensor
2. M
3. Ve
4. Fo
5. Blind spot monitoring camera
6. Drive recorder
7. Side obstacle sensor
8. Air pressure sensor
9. Inside door lock/unlock
10. Rear obstacle sensor
11. GPS sensor
12. Airbag
15. Water repelling wind shield
16. Seatbelt pretensioner
17. Driver monitoring sensor
18. Headup display
19. Steering angle sensor
20. Electronic control throttle
21. Electronic control brake
22. Fire detection sensor
25. Pedestrian collision injury reduction structure
26. Electronic control steering
27. Message display system
28. Hands-free system

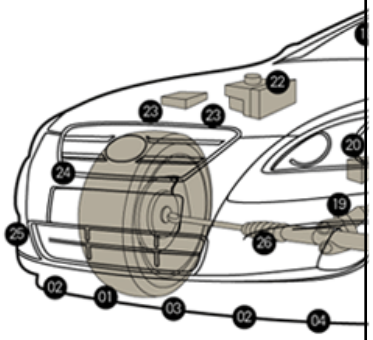
Vehicle engine control unit

https://mocana.com/blog/tag/medical-device/; www.sustainabletechnolog.com; www.kia.co.nz; www.woodward.com;

Popular CGRA usage for reliability concern

- **CGRA can be used widely close to humans**
  - **Reliability in CGRA is becoming an important issue**

Advanced Safety Vehicle



1. Road condition sensor
2. M
3. Ve
4. Fo
5. Blind spot monitoring camera
6. Drive recorder
7. Side obstacle sensor
8. Air pressure sensor
9. Inside door lock/unlock
10. Rear obstacle sensor
11. GPS sensor

12. Air
15. Wa
16. Sea
17. Dri
18. He
19. Ste
20. Ele
21. Ele

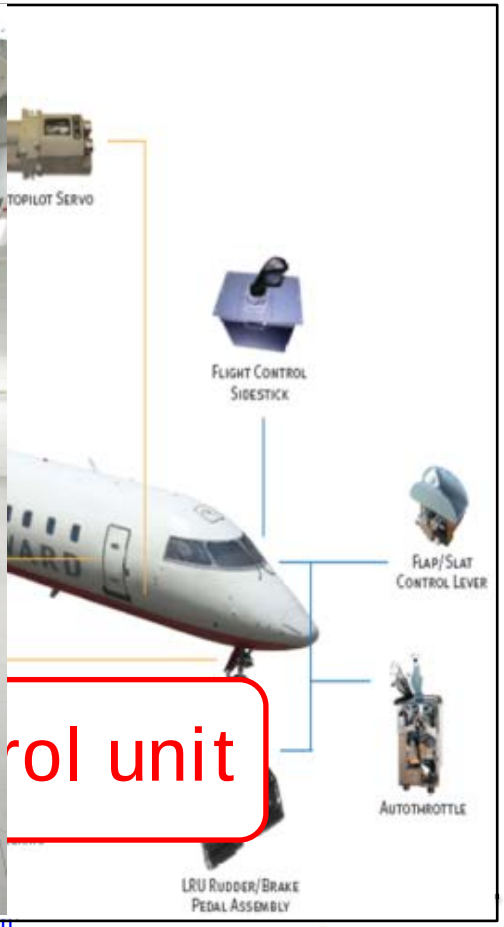
Vehicle en

Airplane control unit

https://mocana.com/blog/tag/medical-device/; www.sustainabletechnolog.com, www.kia.co.nz, www.woodward.com,

Popular CGRA usage for reliability concern

- **CGRA can be used widely close to humans**
  - **Reliability in CGRA is becoming an important issue**



rol unit

https://mocana.com/blog/tag/medical-device/; www.sustainabletechnolog.com, www.kia.co.nz, www.woodward.com,

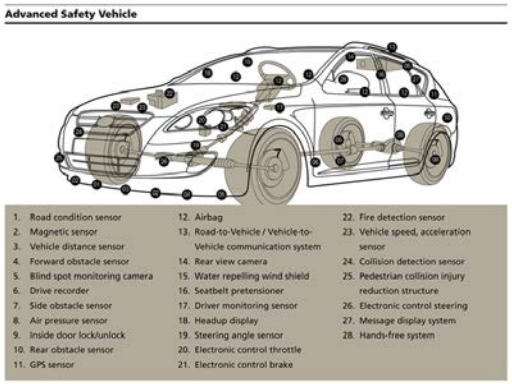
Popular CGRA usage for reliability concern

- **CGRA can be used widely close to humans**
  - **Reliability in CGRA is becoming an important issue**

https://mocana.com/blog/tag/medical-device/; www.sustainabletechnolog.com, www.kia.co.nz, www.woodward.com,

Popular CGRA usage for reliability concern

Vehicle engine control unit



Airplane control unit

Human life

Data server system

Health system

Soft error occurrence in various embedded systems can cause catastrophic results

https://mocana.com/blog/tag/medical-device/; www.sustainabletechnolog.com; www.kia.co.nz; www.woodward.com;

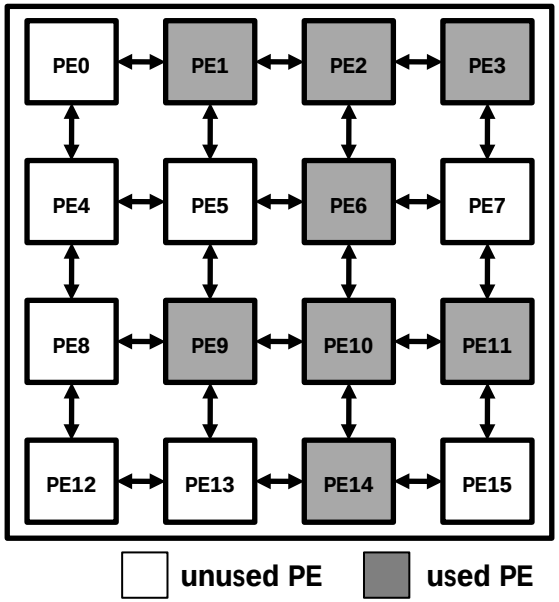
2. Related works

Soft Error : hardware protection
software protection

Previously proposed techniques for CGRAs

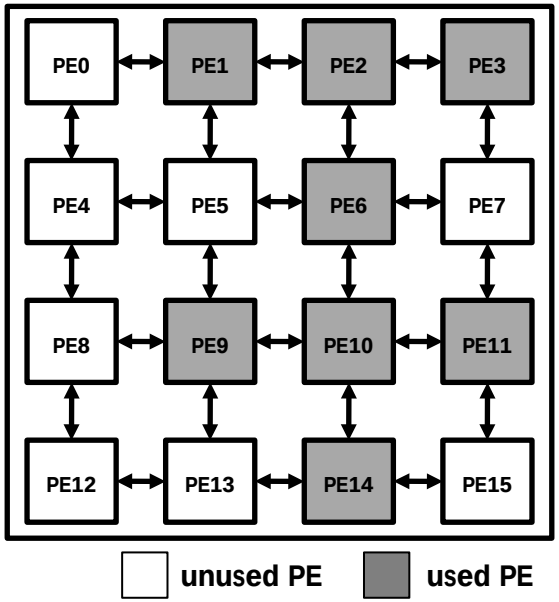
Paper	Key Idea	Experiment		drawback	Comment
[Alnajiar, 2009]	dynamic operation modes in CGRA architecture to provide the various levels of reliability under the performance constraint.	set up	- compare to the number of gate (using tool : RTL)	- flexible protection mechanism	- **area-overhead (implement voter)** - performance degradation (cant use resource fully)
		result	- area : 26.6% increase		
[jafri, 2010]	self-checking residue mode for multiplication and addition operations on DART architecture	set up	- compare to original FU, DMR FU and self-checking FU (using STMicroelectronics 130 nm)	- less area-overhead compare to DMR	- **area-overhead (implement self-checking)** - performance degradation - only detection - not cover specific fault
		result	- area : 18% decrease compare to DMR - performance : 400% decrease compare to DMR		
[Schweizer, 2011]	exploiting unused FUs for replications to increase the reliability with the minimal hardware overhead. FEHM (Flexible Error Handling Module) - supports DMR and TMR schemes on specific target architectures.	set up	- compare to TMR and Clustering PE include FEHM	- supports DMR and TMR schemes with modified FEHM - considering power	- **area-overhead (implement FEHM)** - cant apply to data intensive application
		result	- area : 12.8% decrease compare to TMR - power : 1.6~18.6% decrease compare to TMR		
[Schweizer, 2012]	to resolve previous ([Schweizer, 2011]) limitation, multiple contexts to be mapped on CGRA by using the concept of temporal Redundancy	set up	- mapping to CGRA used FFT application - estimate required area as context memory is increased - Estimate time between read and store	- enable to apply permanent, transient, timing fault - enable to apply any application	- performance degradation compare to TMR (increase context) - **still exits area-overhead**
		result	- area : 31% decrease compare to TMR - performance : NR/TMR 26%/12% decrease		
[K. Singh, 2006]	Selective apply combined scheme to code that can cause a soft error.	set up	- reliability is the percentage of time that FT matrix multiplication can run on raw architecture without system reset	- no area-overhead (software based technique)	- limited RAW architecture - **performance overhead** - not consider TMR voting
		result	- reliability : 89.2%(108 out of 1000 reset)		
[Lee, 2010]	replication & voter is implemented on PE to reduce area overhead; to reduce the critical path, add conditional execution & column wide bus; thermal impact optimal mapping on PE is proposed.	set up	- compare base arch with proposed arch based on RTL - ACS(time consuming operations) module in viterbi decoder map CGRA	- software technique - area-efficiency (minimal hardware overhead)	- **performance overhead** (replication & voter map on PE)
		result	- area : 12% increase compare to base - performance : decrease TMR(700%)/DMR (167%)		

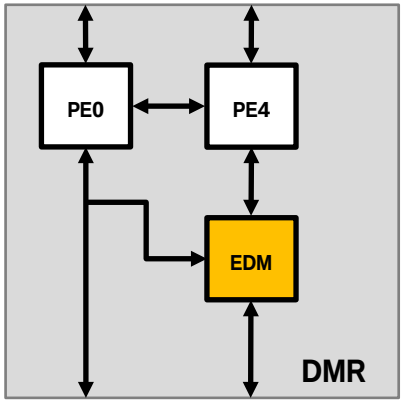
HW based techniques - area overhead

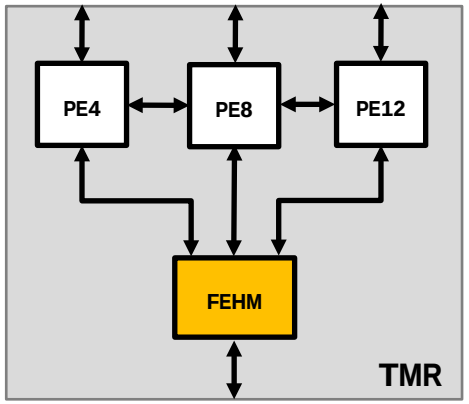


T. Schweizer, A. Kuster, S. Eisenhardt, T. Kuhn, and W. Rosenstiel, “Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures,” in IPDPSW, 2012.

HW based techniques - area overhead

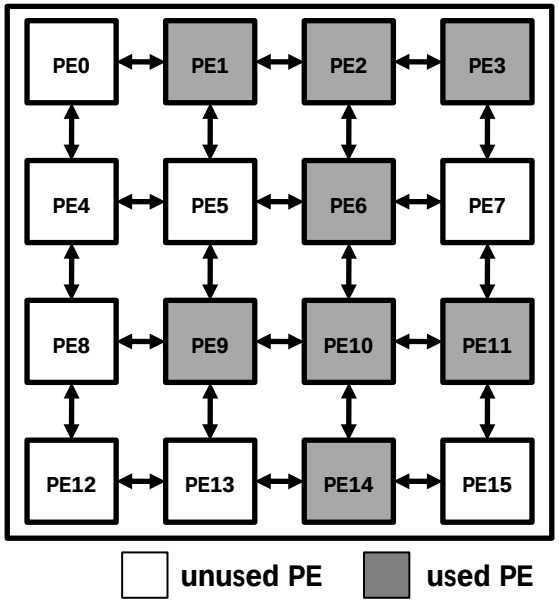


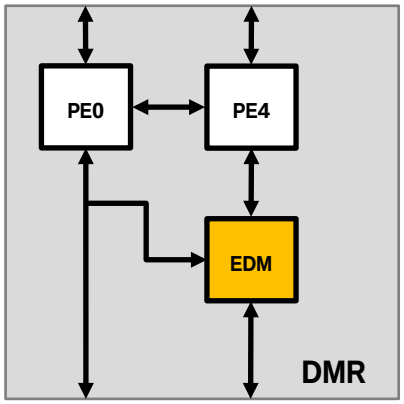


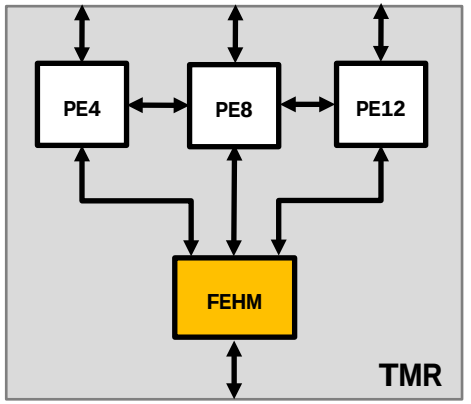


T. Schweizer, A. Kuster, S. Eisenhardt, T. Kuhn, and W. Rosenstiel, “Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures,” in IPDPSW, 2012.

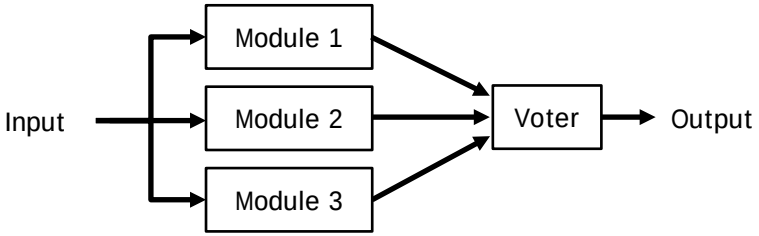
HW based techniques - area overhead





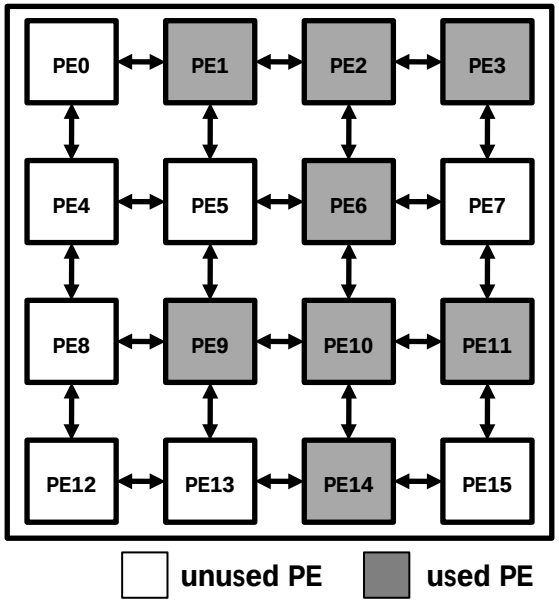


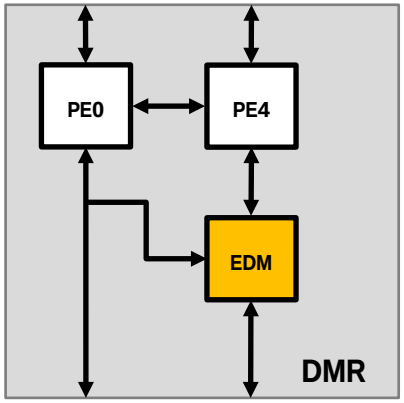
- DMR (Dual Modular Redundancy)
- TMR (Triple Modular Redundancy)

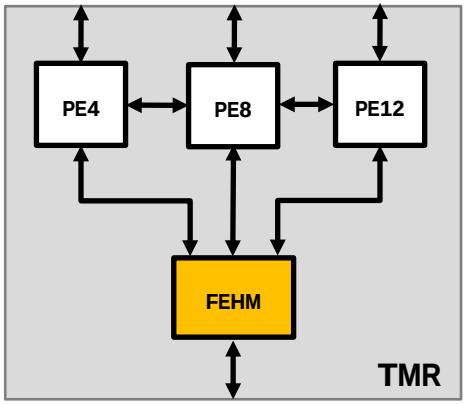


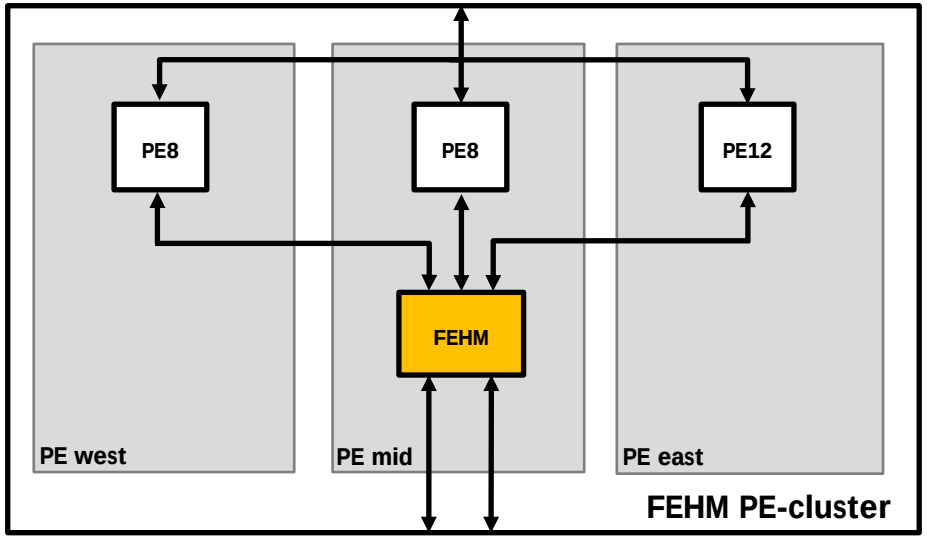
T. Schweizer, A. Kuster, S. Eisenhardt, T. Kuhn, and W. Rosenstiel, “Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures,” in IPDPSW, 2012.

HW based techniques - area overhead

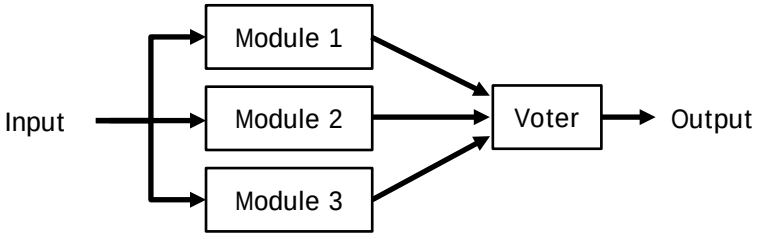






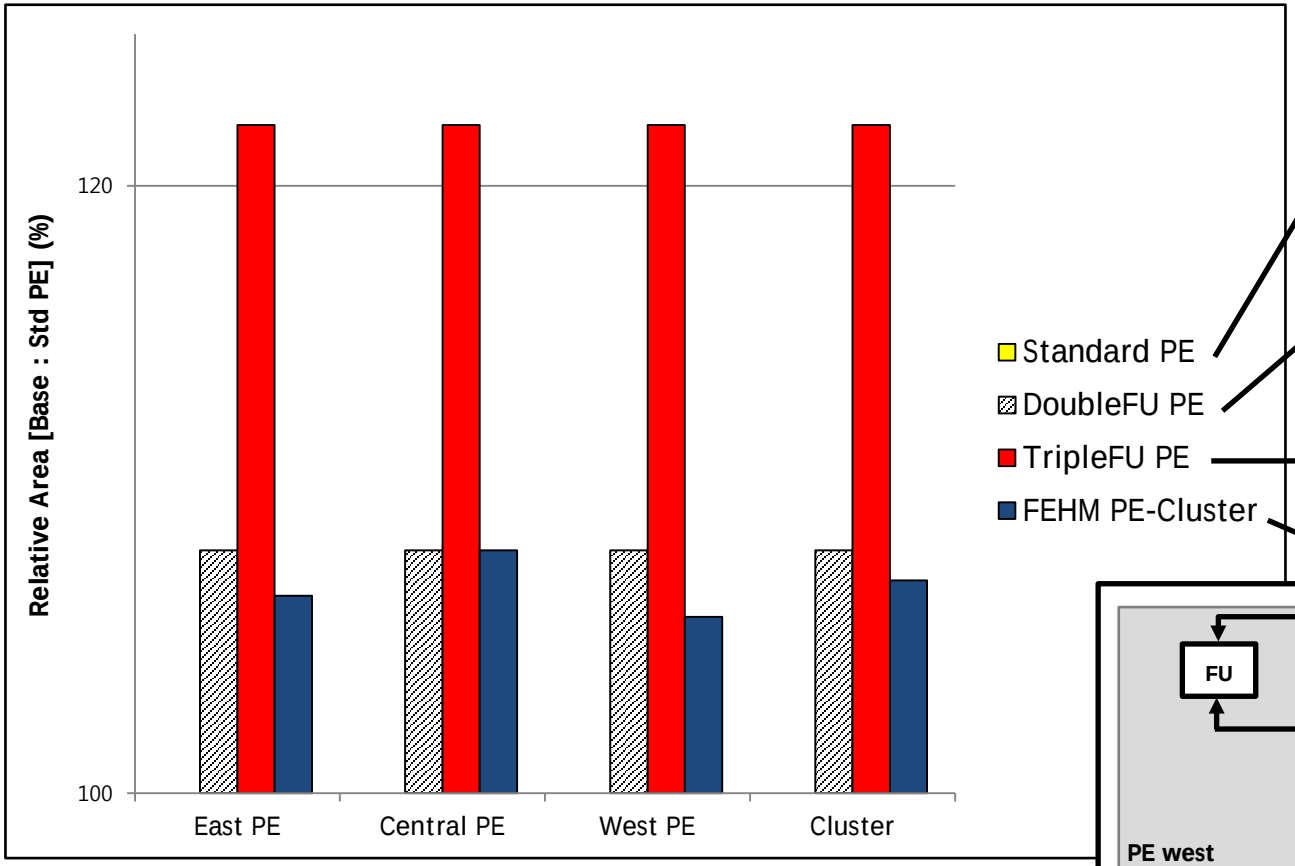


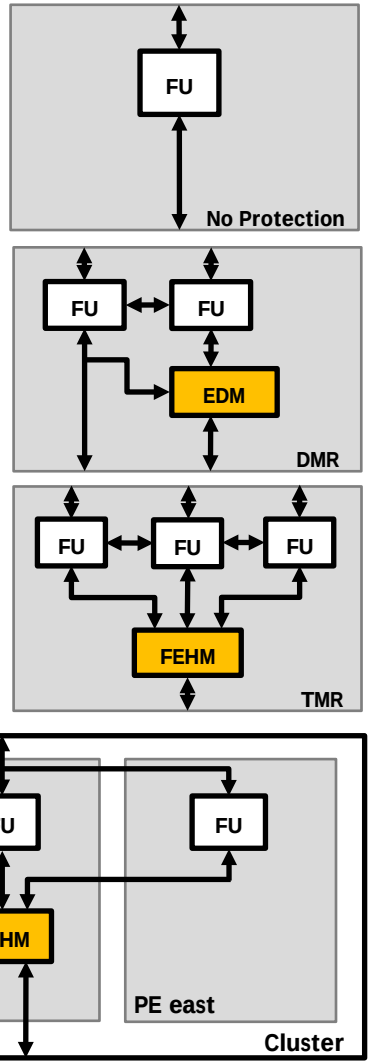
- DMR (Dual Modular Redundancy)
- TMR (Triple Modular Redundancy)



T. Schweizer, A. Kuster, S. Eisenhardt, T. Kuhn, and W. Rosenstiel, “Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures,” in IPDPSW, 2012.

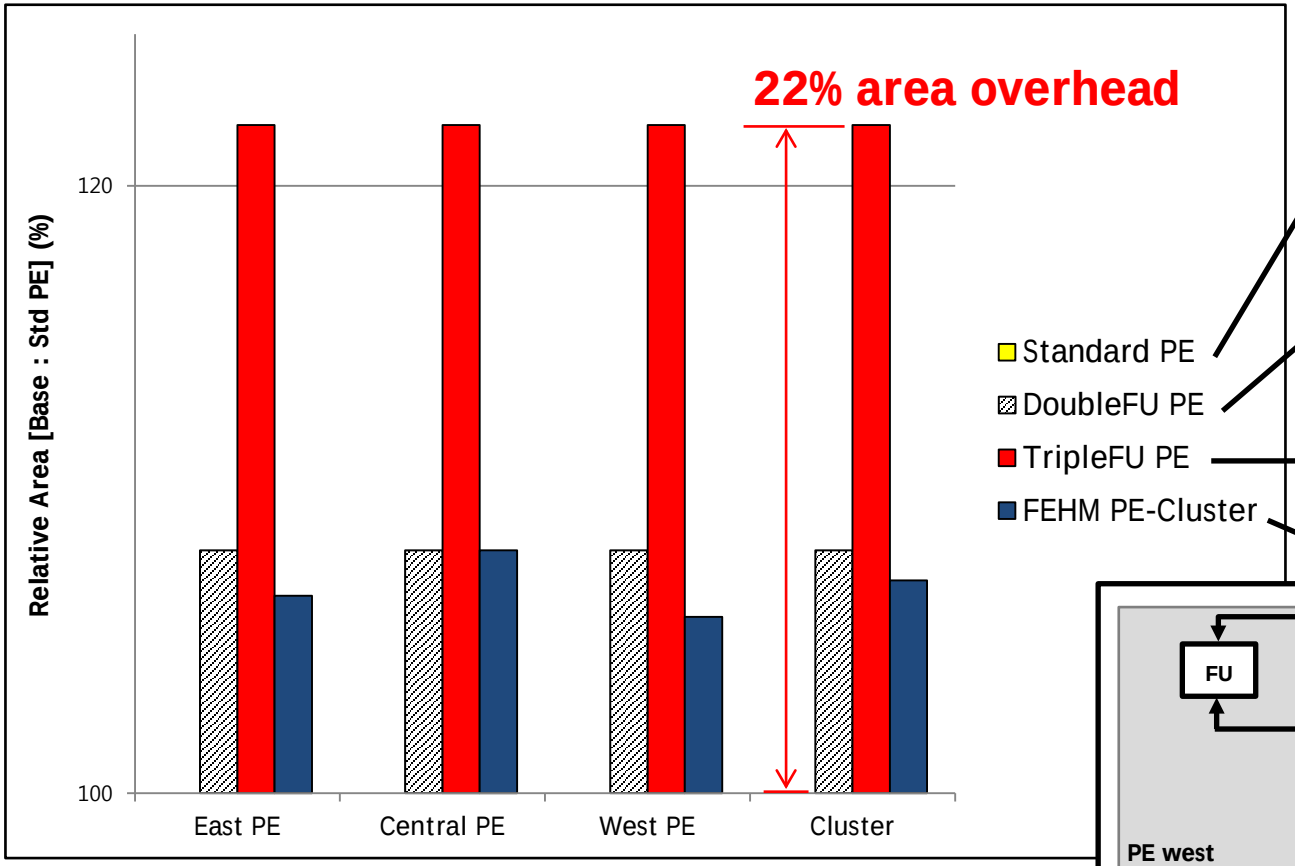
HW based techniques - area overhead

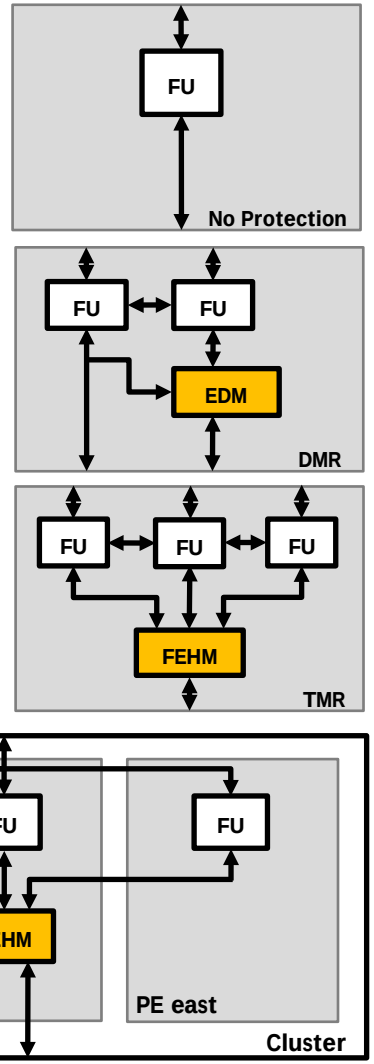




T. Schweizer, A. Kuster, S. Eisenhardt, T. Kuhn, and W. Rosenstiel, "Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures," in IPDPSW, 2012.

HW based techniques - area overhead

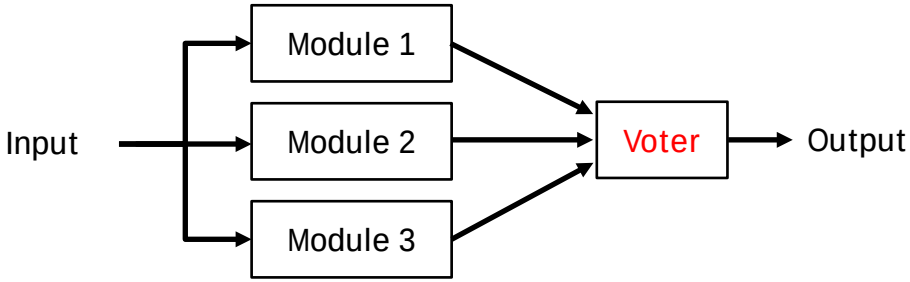




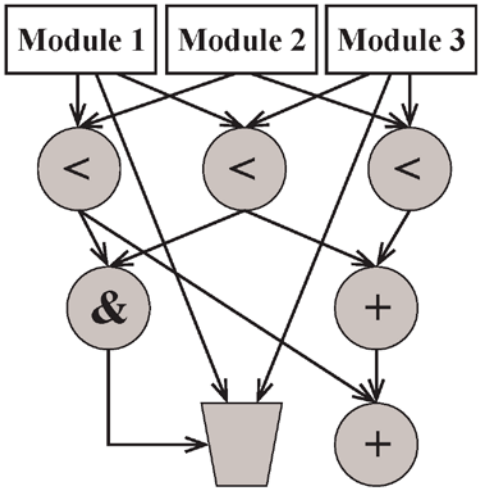
T. Schweizer, A. Kuster, S. Eisenhardt, T. Kuhn, and W. Rosenstiel, "Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures," in IPDPSW, 2012.

SW based techniques - performance overhead

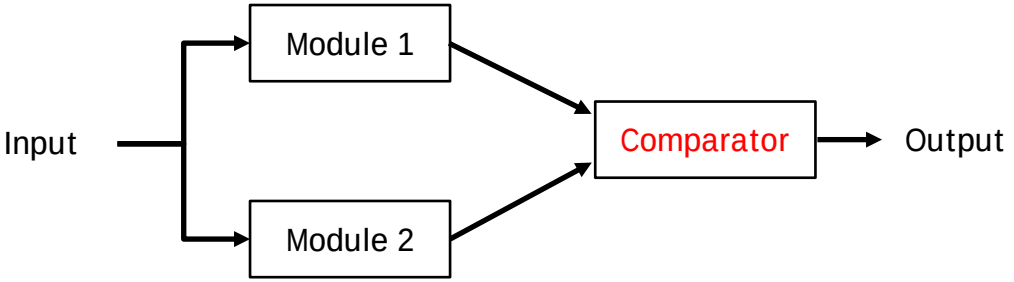
- TMR (Triple Modular Redundancy)



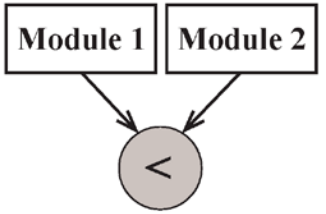
TMR voter implementation



- DMR (Dual Modular Redundancy)

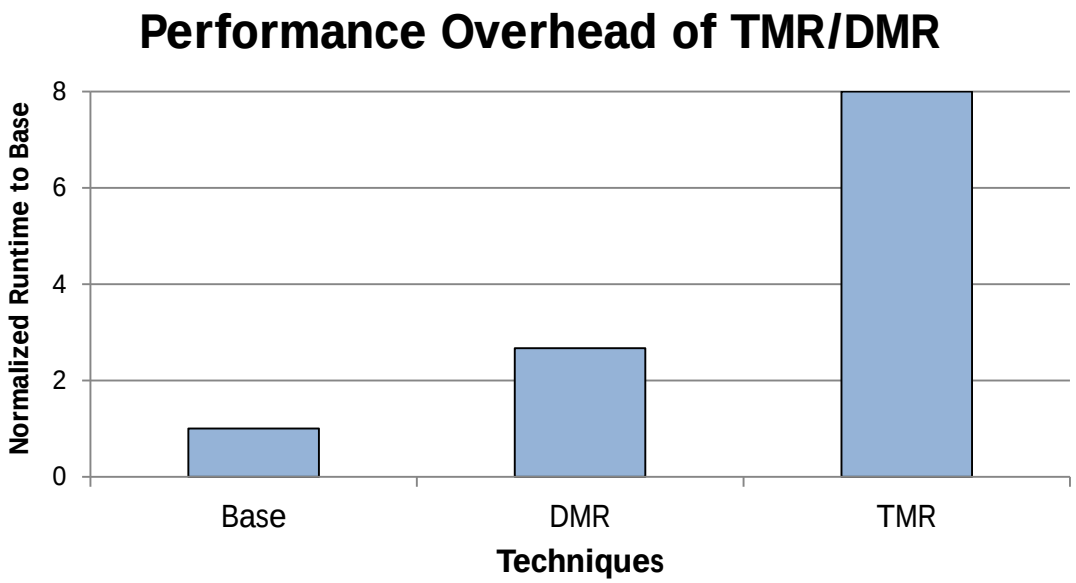


DMR comparator implementation



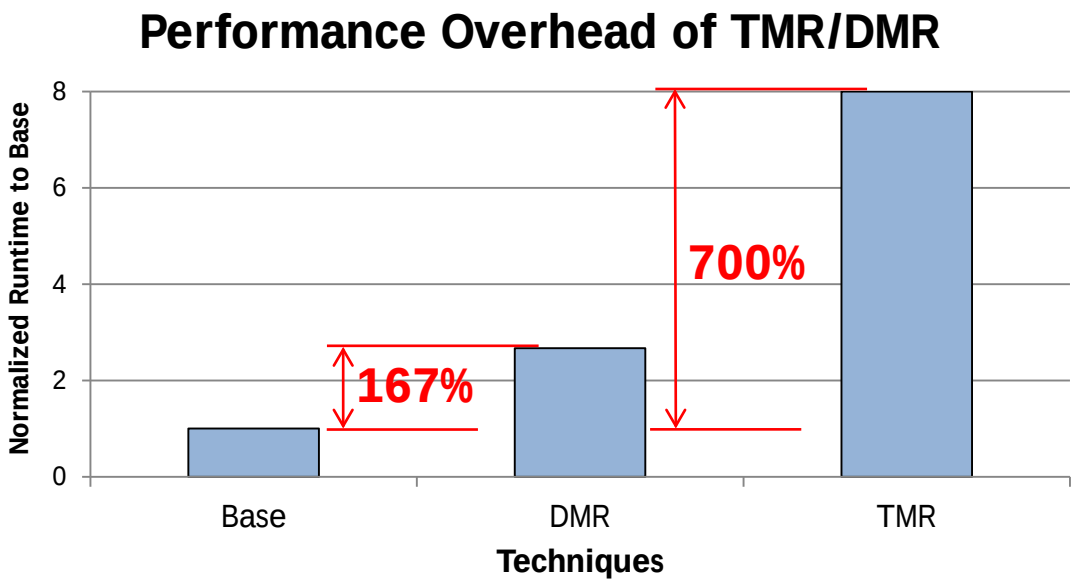
G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead



G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead



G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

- **DFG is generated for loop kernel**

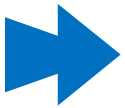
```
...
  for ( i = 0; i < iteration; i ++) {
      a[i] = ( b[i] – X ) / Y;    /* X and Y are constants */
  }
...
```

G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

- **DFG is generated for loop kernel**


```
...
  for ( i = 0; i < iteration; i ++) {
      a[i] = ( b[i] – X ) / Y;    /* X and Y are constants */
  }
...
```

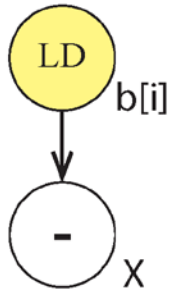


G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

- **DFG is generated for loop kernel**

```
...
  for ( i = 0; i < iteration; i ++) {
      a[i] = ( b[i] – X ) / Y;    /* X and Y are constants */
  }
...
```

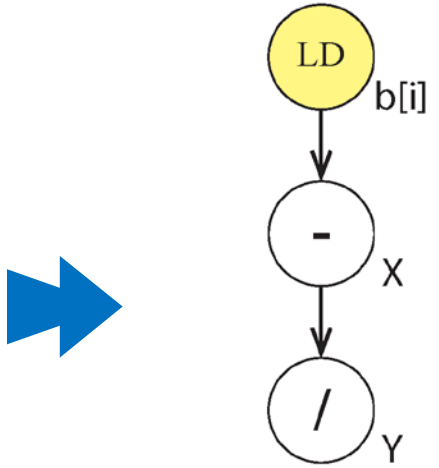


G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

- **DFG is generated for loop kernel**

```
...
  for ( i = 0; i < iteration; i ++) {
      a[i] = ( b[i] – X ) / Y;    /* X and Y are constants */
  }
...
```

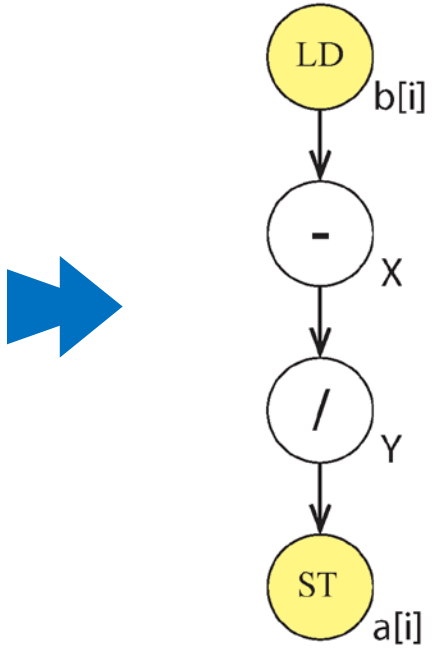


G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

- **DFG is generated for loop kernel**

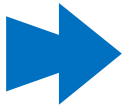
```
...
  for ( i = 0; i < iteration; i ++) {
      a[i] = ( b[i] – X ) / Y;    /* X and Y are constants */
  }
...
```

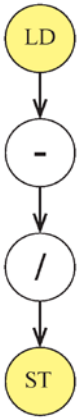


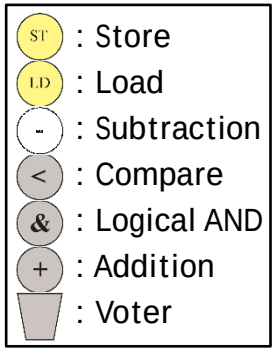
G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

```
...
   for ( i = 0; i < iteration; i ++) {
        a[i] = ( b[i] – X )  / Y;    /* X and Y are constants */
   }
...
```



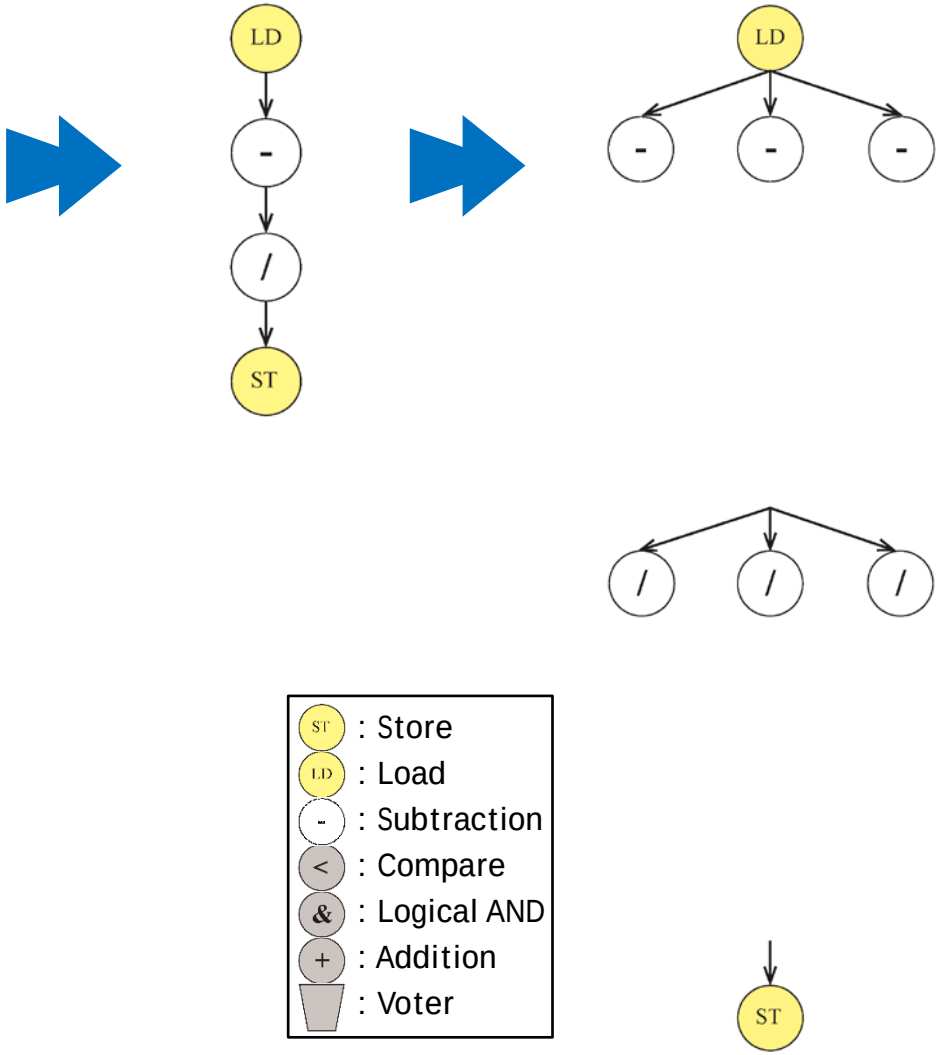




G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

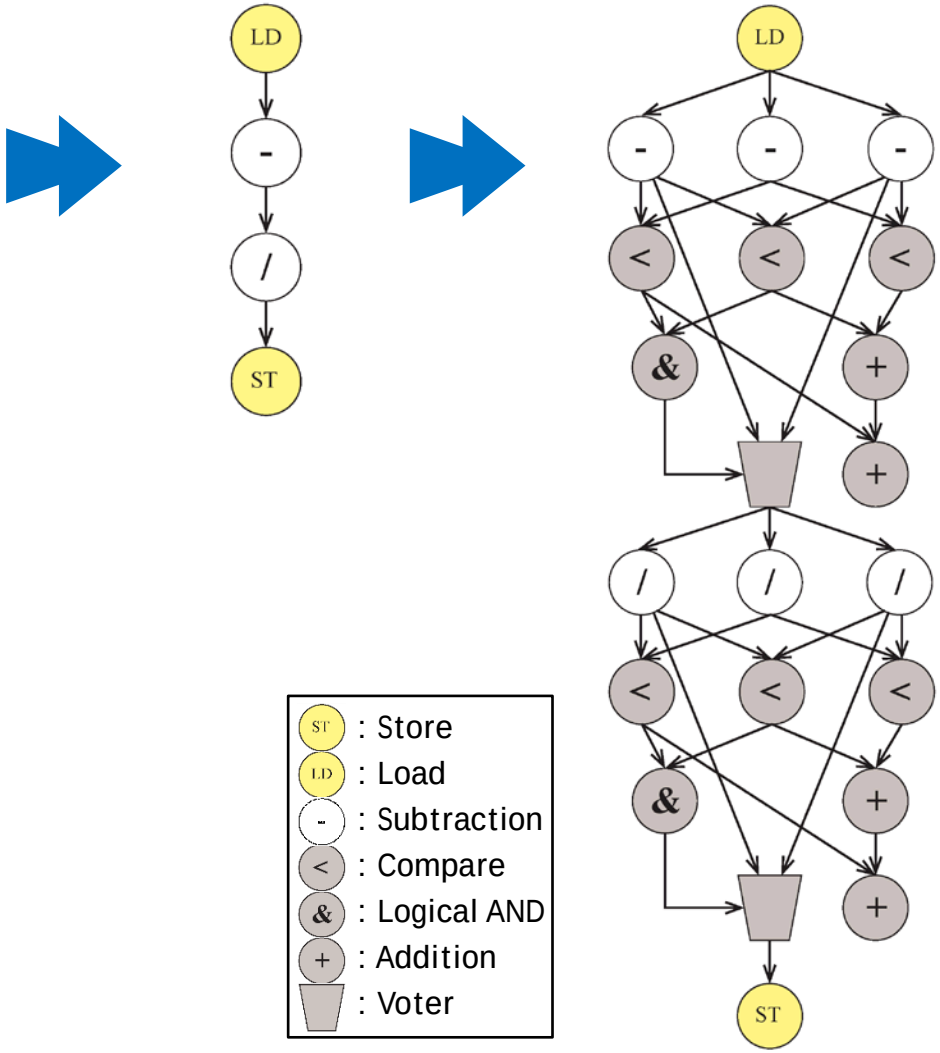
```
...
   for ( i = 0; i < iteration; i ++) {
        a[i] = ( b[i] – X )  / Y;    /* X and Y are constants */
   }
...
```



G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

SW based techniques - performance overhead

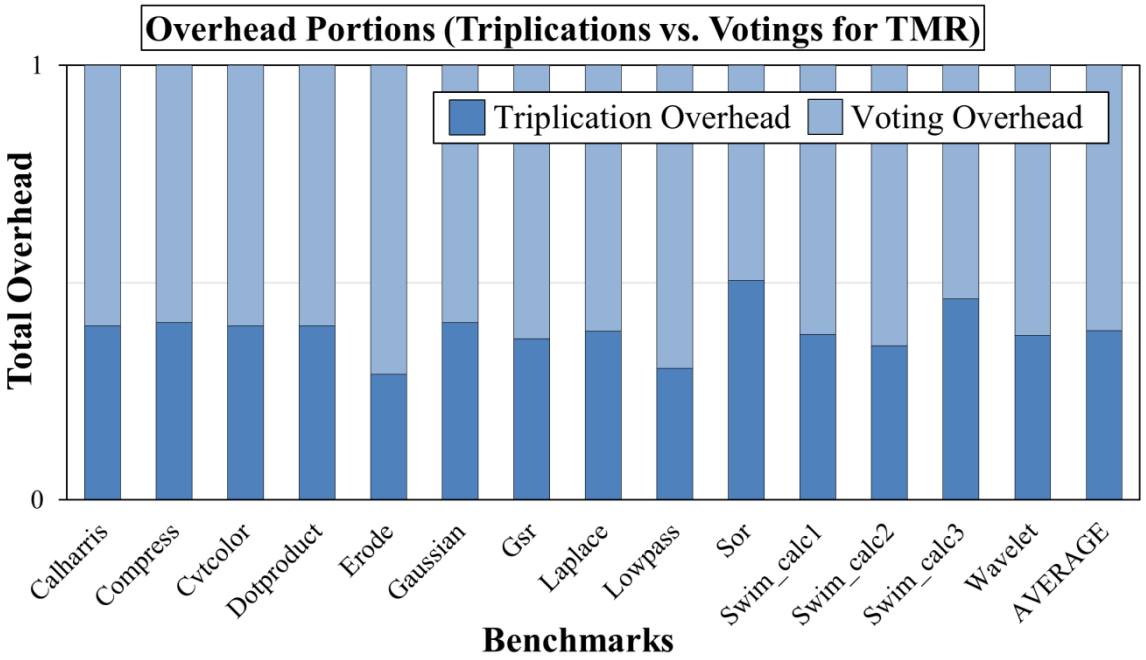
```
...
   for ( i = 0; i < iteration; i ++) {
        a[i] = ( b[i] – X )  / Y;    /* X and Y are constants */
   }
...
```



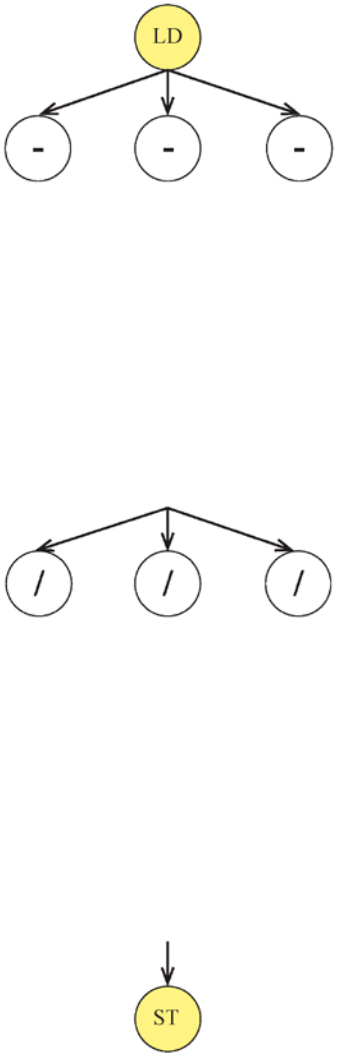
G. Lee and K. Choi, "Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture," in AHS, 2010.

3. Problem definition

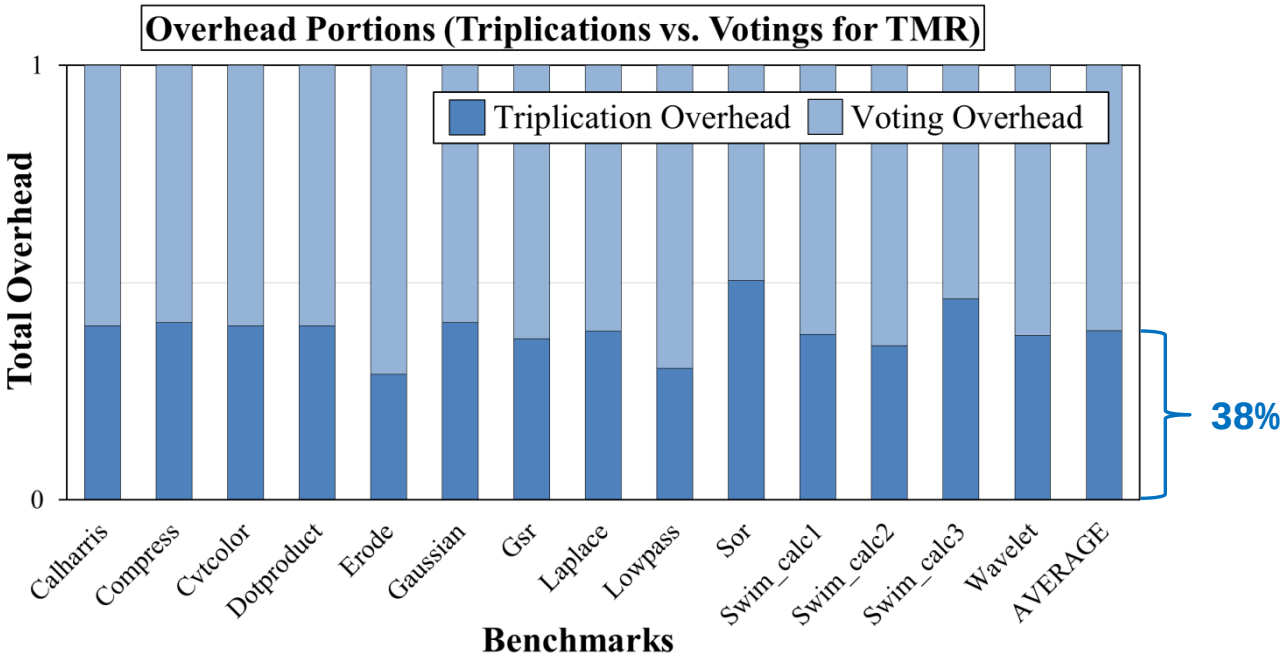
Significantly expensive voting mechanism



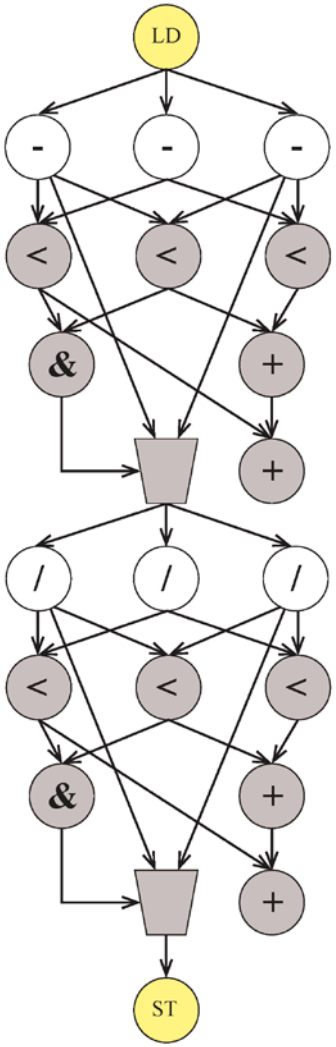
Significantly expensive voting mechanism



Triplication

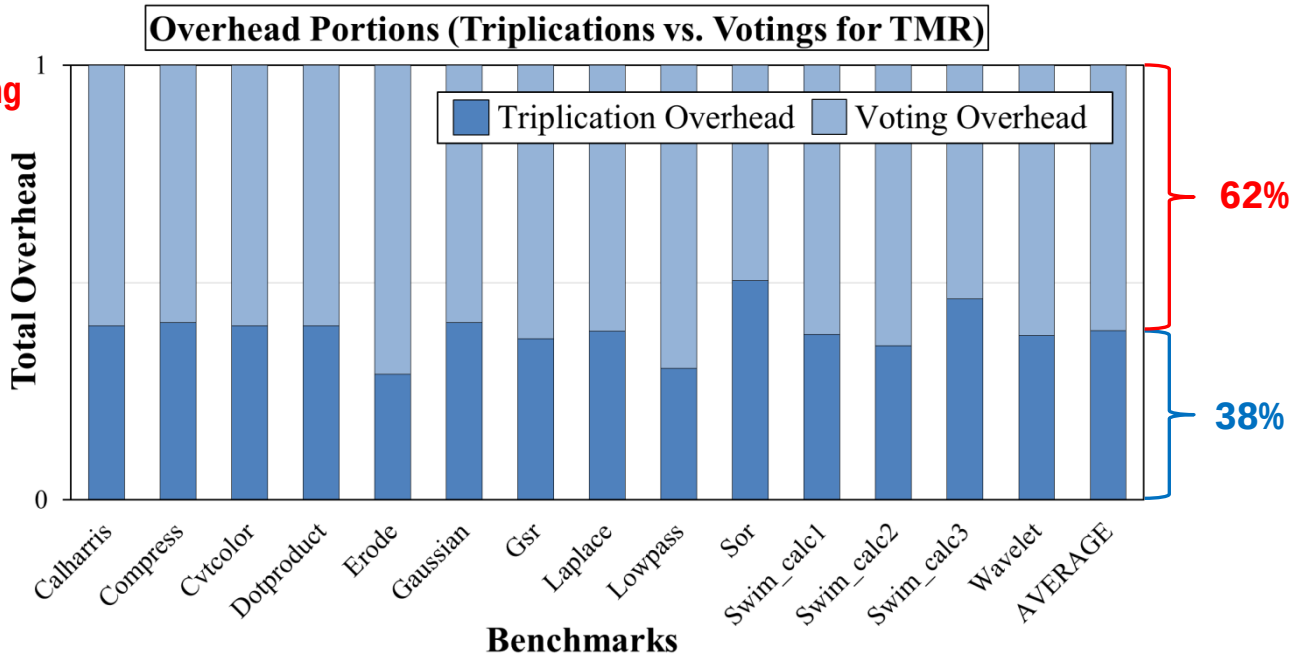


Significantly expensive voting mechanism



Triplication

Voting



4. Thesis proposal: Selective validation for efficient protection

```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
...
```

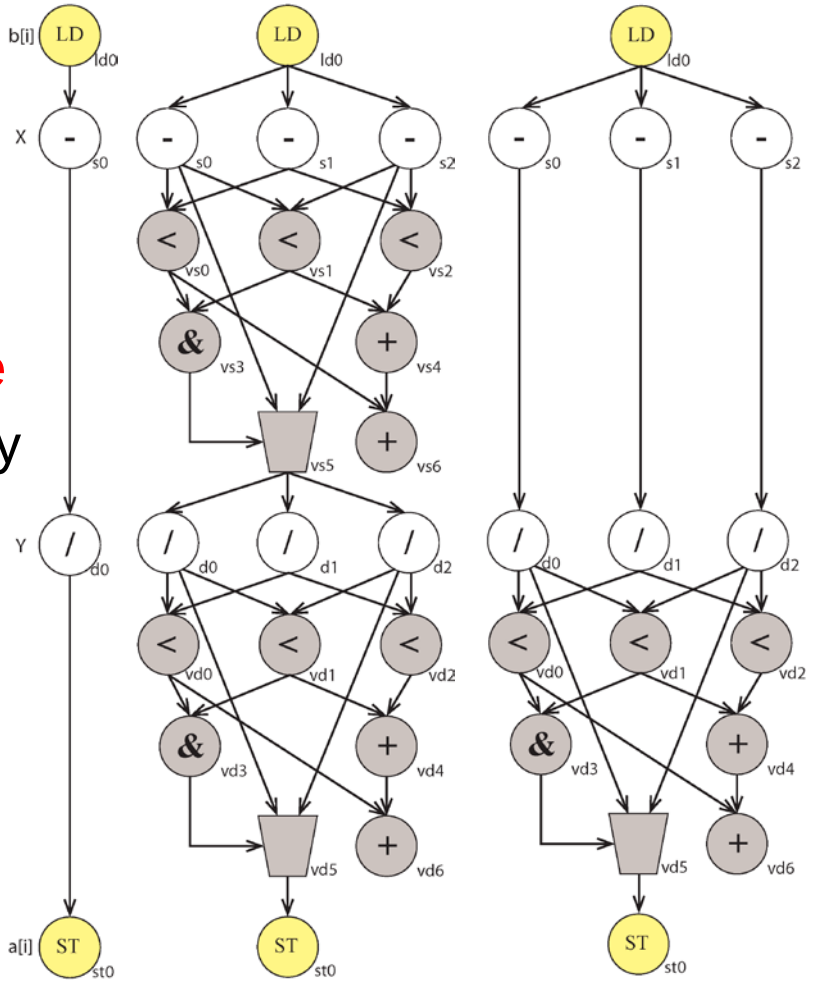
Selective validation : reduce expensive voting

- Suppose that memory of CGRA is protected against errors
  - Traditional protection methodologies for memory are inexpensive as compared to maintaining execution cores
- Do not replicate the memory operation
- Synchronization point where corrupt data can propagate and cause failure
  - erroneous store operations can eventually cause incorrect outputs
- The program will be executed correctly if corrupted data is not stored in the main memory

```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
...
```

Selective validation : reduce expensive voting

- TMR with full voting
  - Voting at every operation (except memory operation)
- TMR with selective voting
  - Voting at operation just before store
  - Memory is protected by ECC or Parity
  - Corrupt data in memory ➔ failure

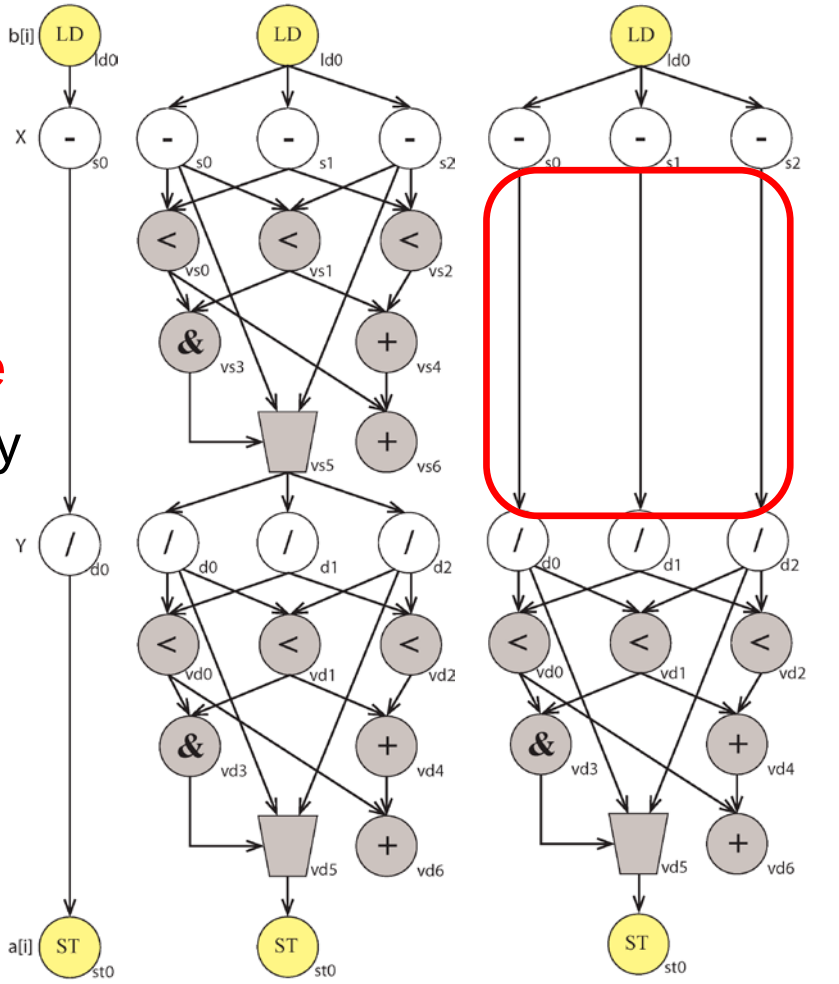


Base TMR with full voting TMR with selective voting

```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
...
```

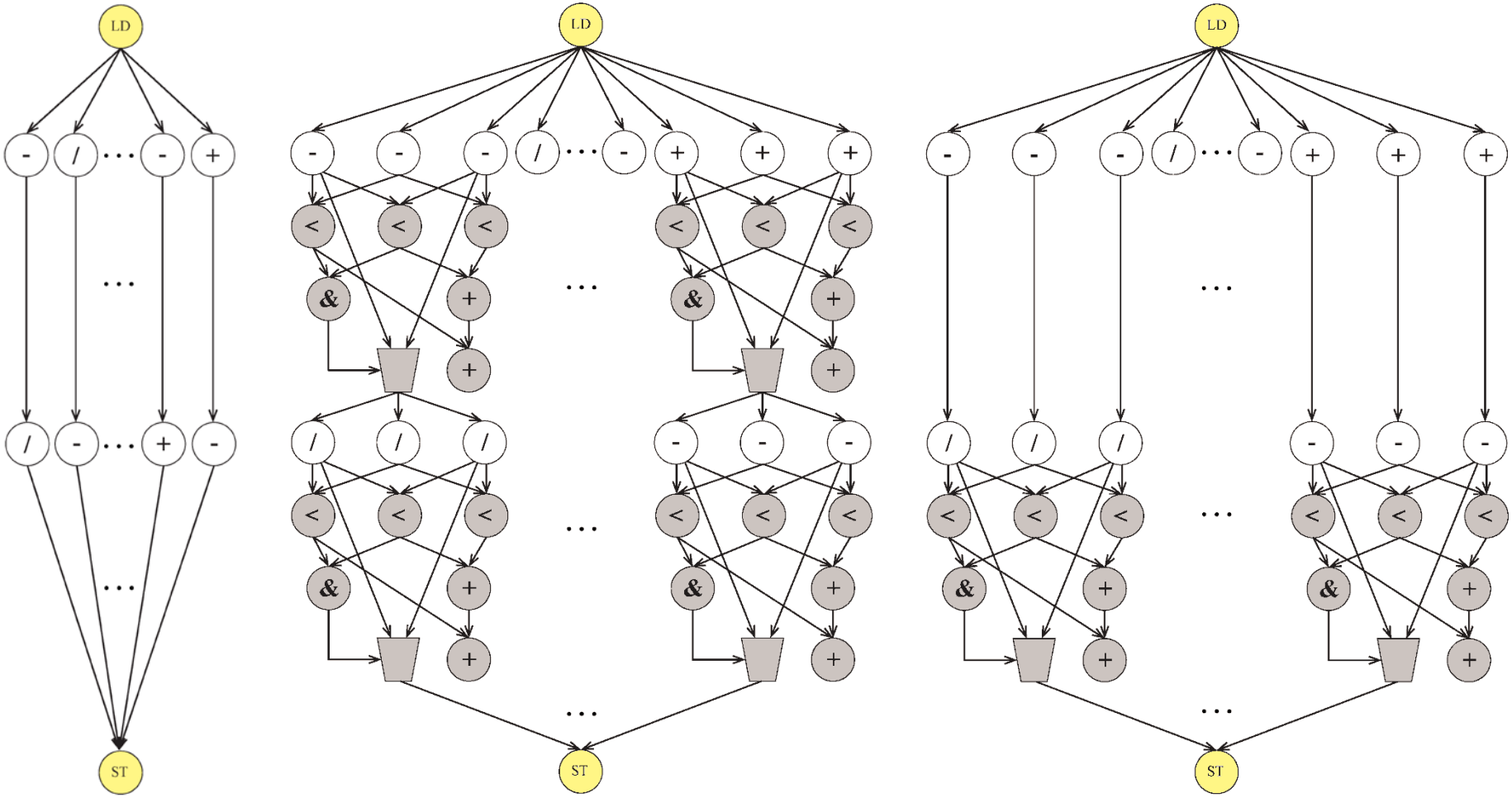
Selective validation : reduce expensive voting

- TMR with full voting
  - Voting at every operation (except memory operation)
- TMR with selective voting
  - Voting at operation just before store
  - Memory is protected by ECC or Parity
  - Corrupt data in memory ➔ failure



Base | TMR with full voting | TMR with selective voting

Ex) Reduce expensive voting

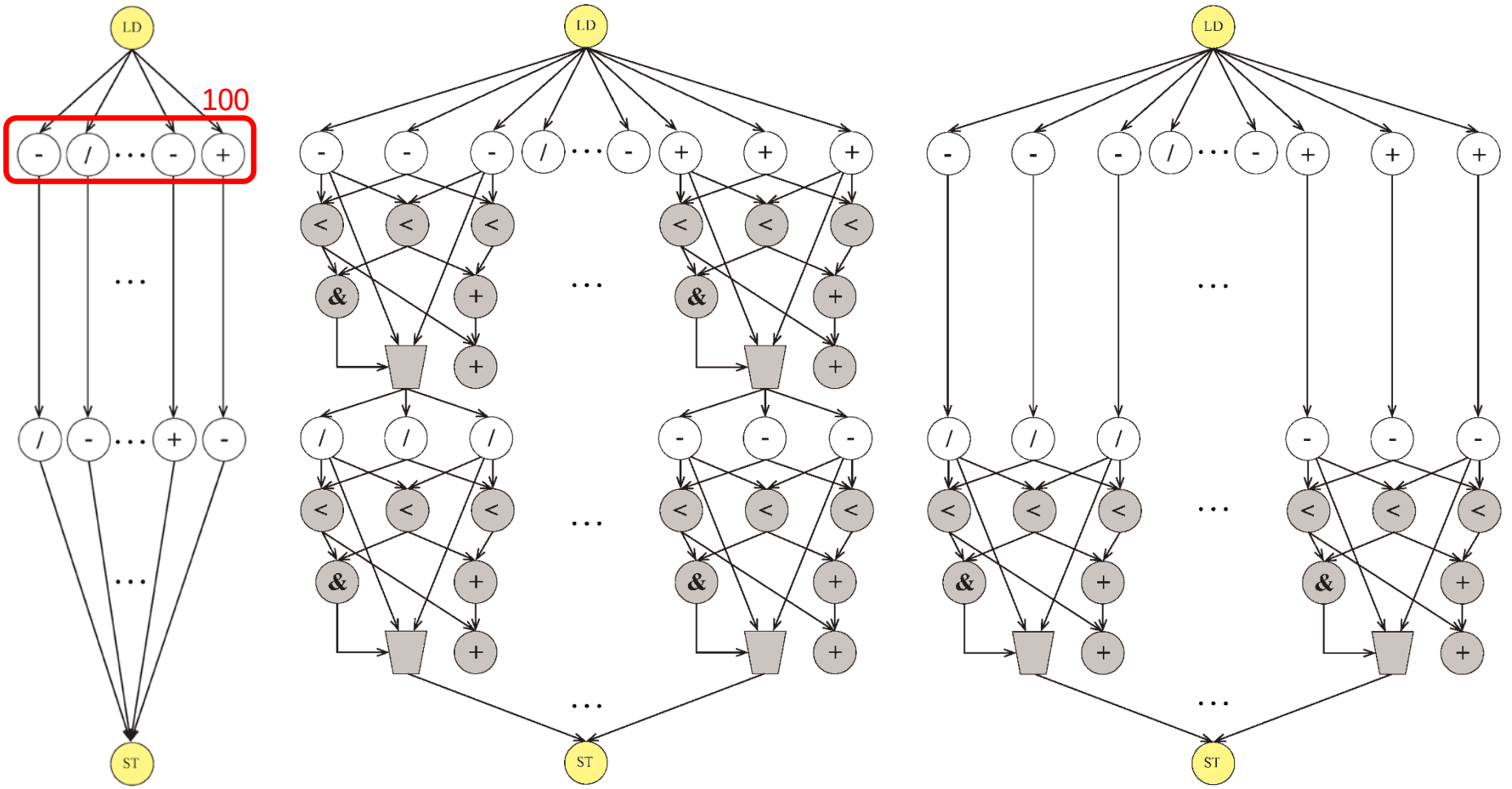


Base

TMR with full voting

TMR with selective voting

Ex) Reduce expensive voting

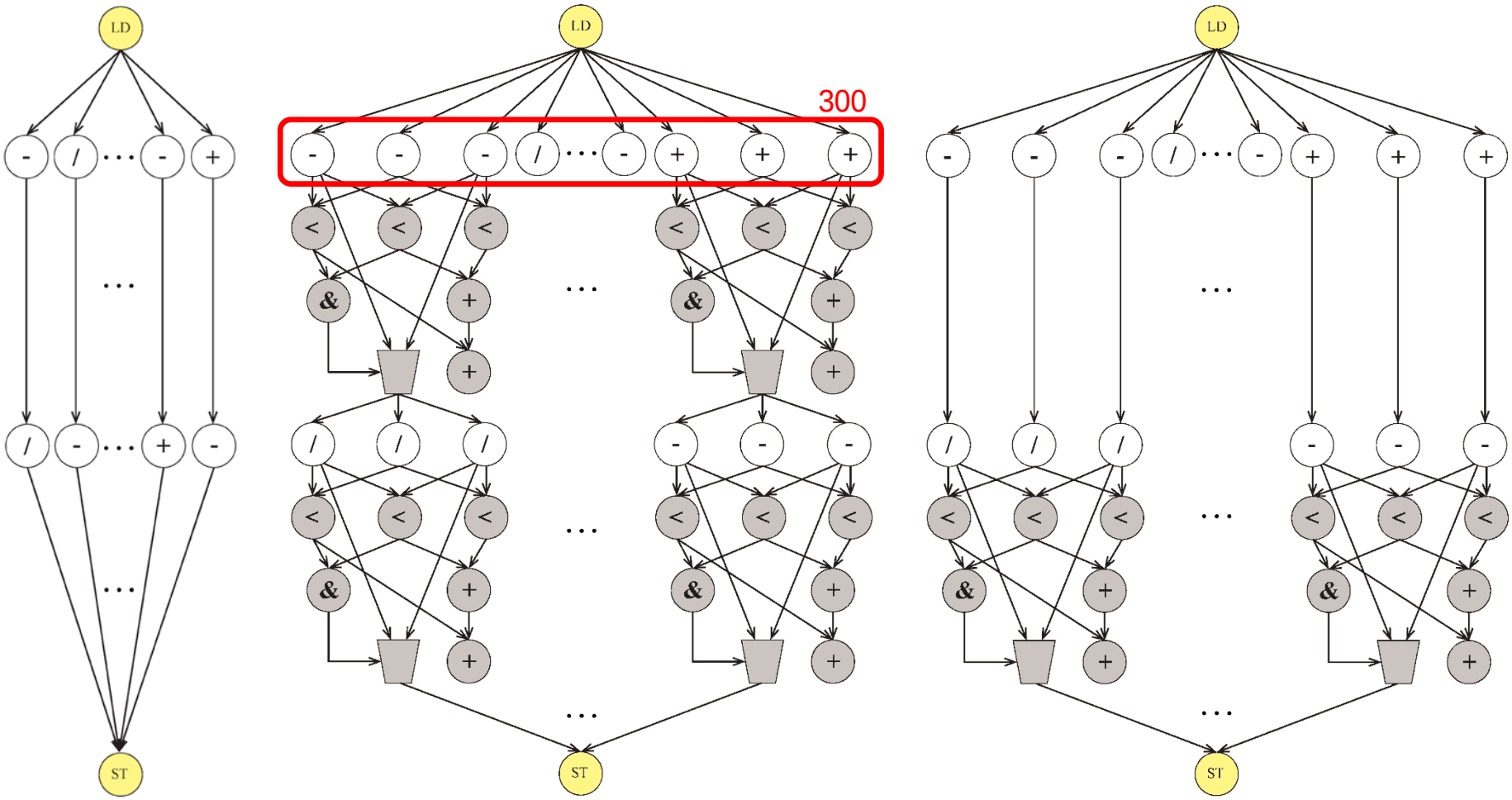


Base

TMR with full voting

TMR with selective voting

Ex) Reduce expensive voting

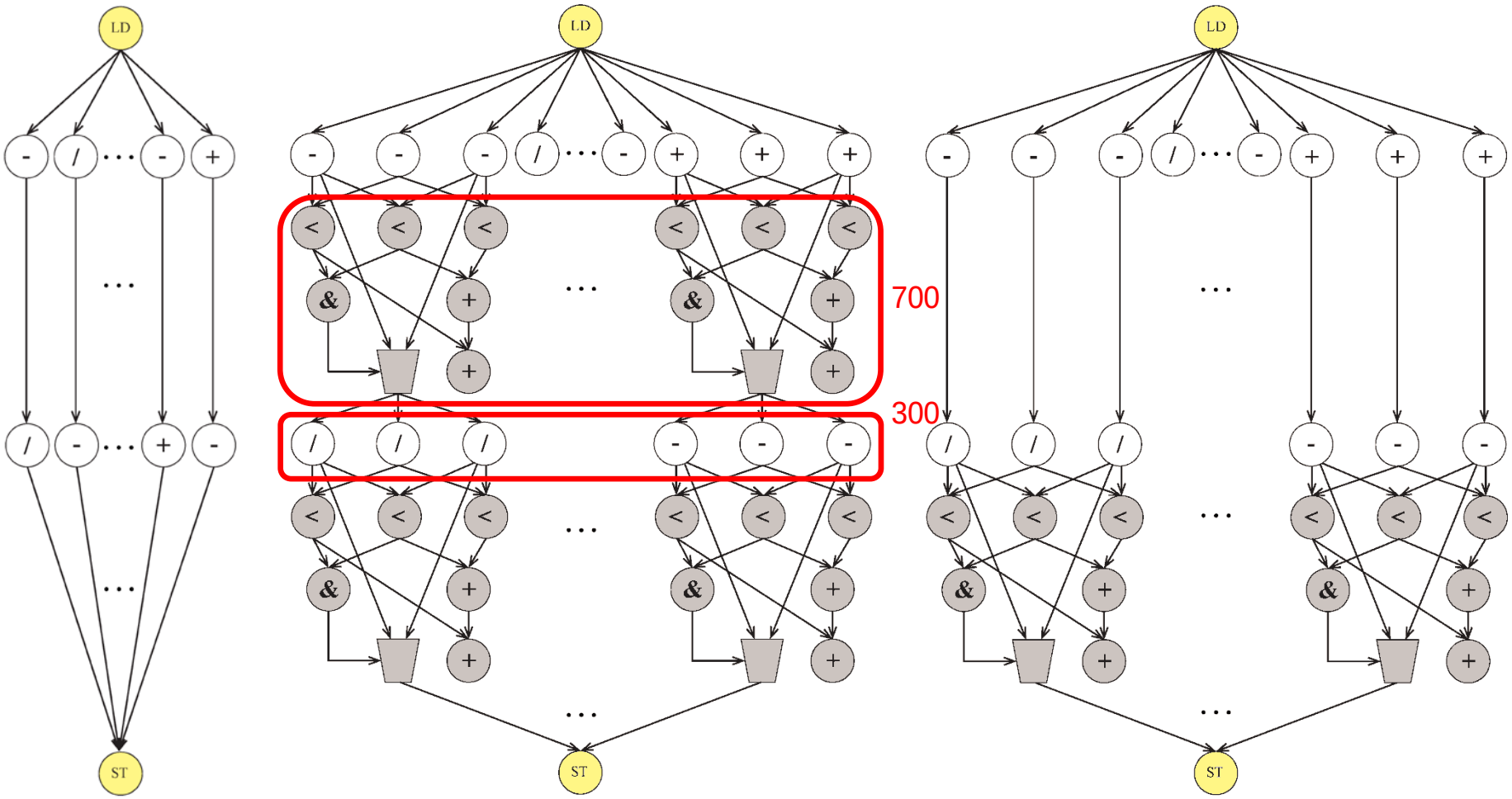


Base

TMR with full voting

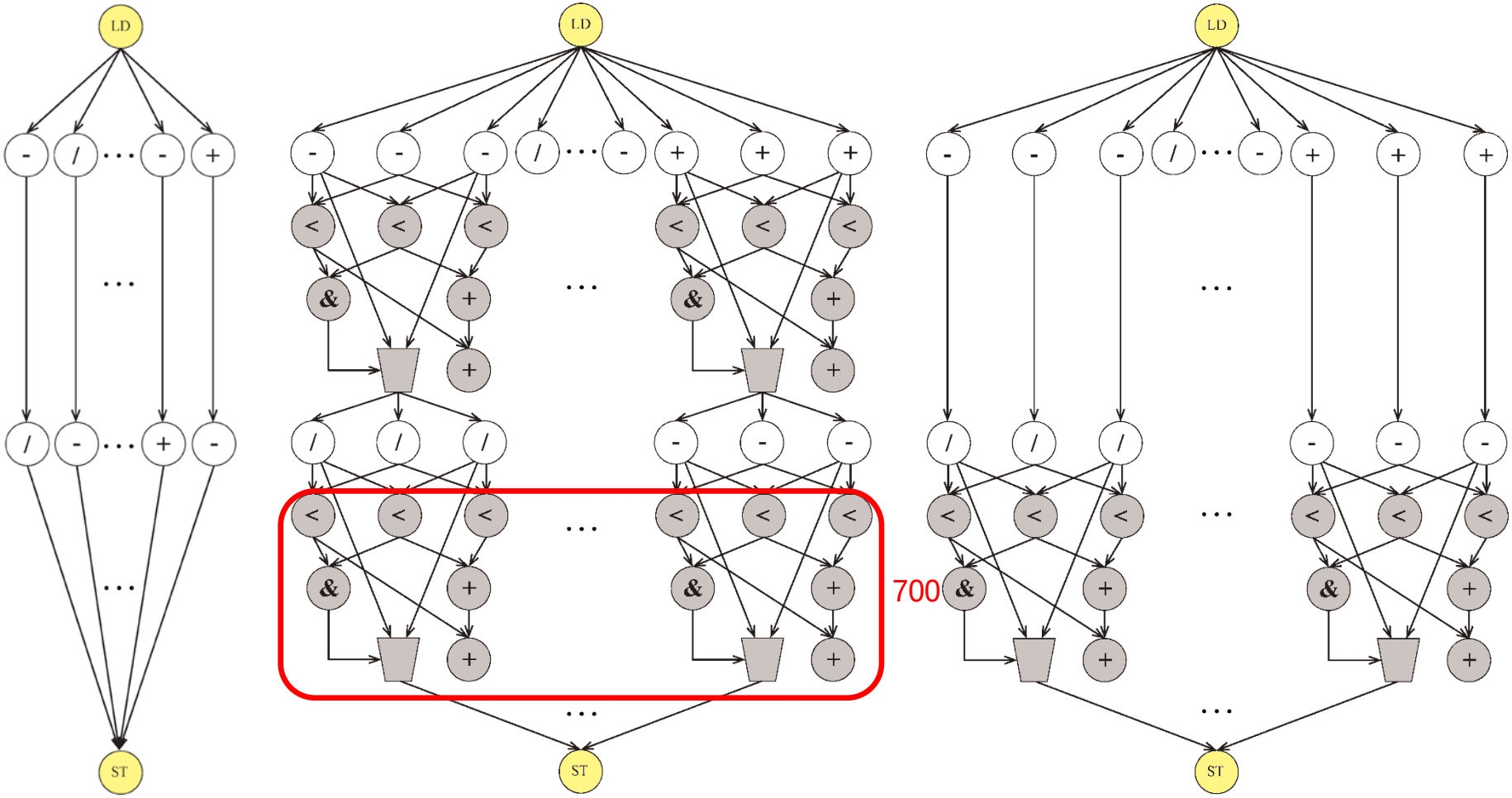
TMR with selective voting

Ex) Reduce expensive voting



Base | TMR with full voting | TMR with selective voting

Ex) Reduce expensive voting

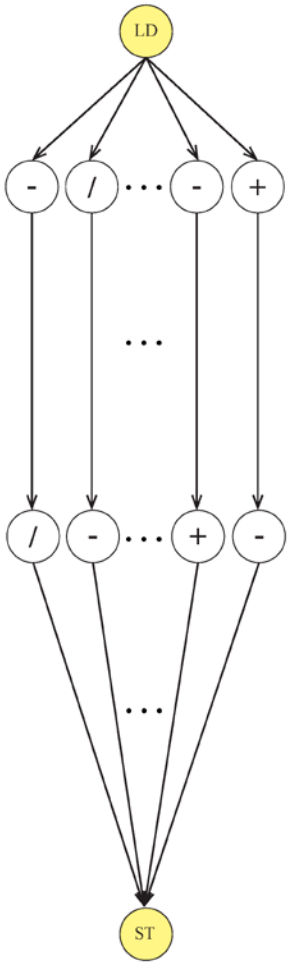


Base

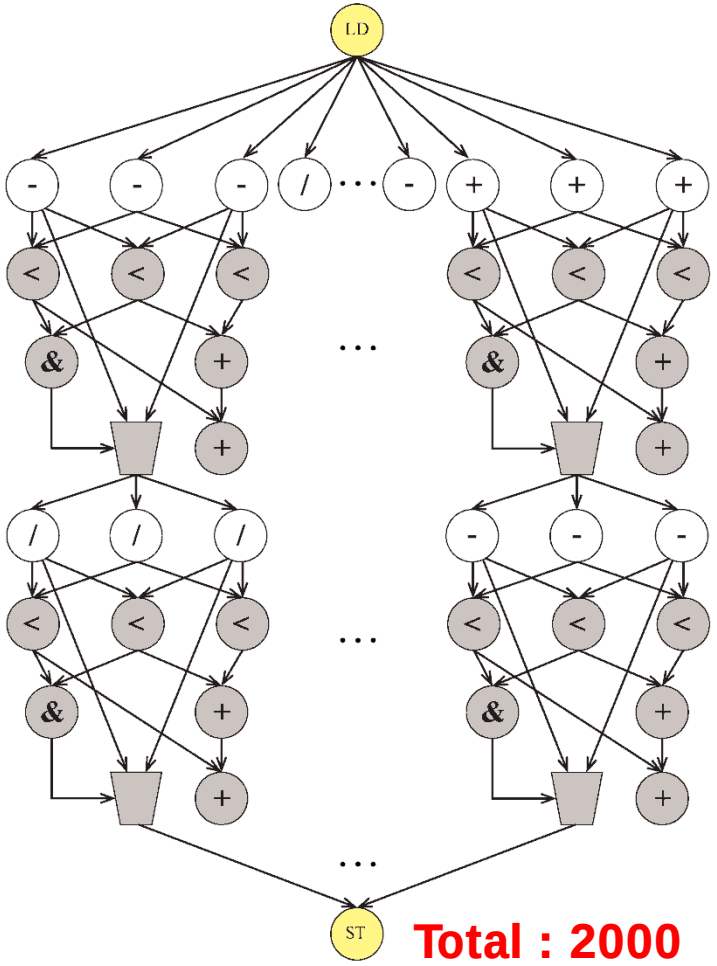
TMR with full voting

TMR with selective voting

Ex) Reduce expensive voting

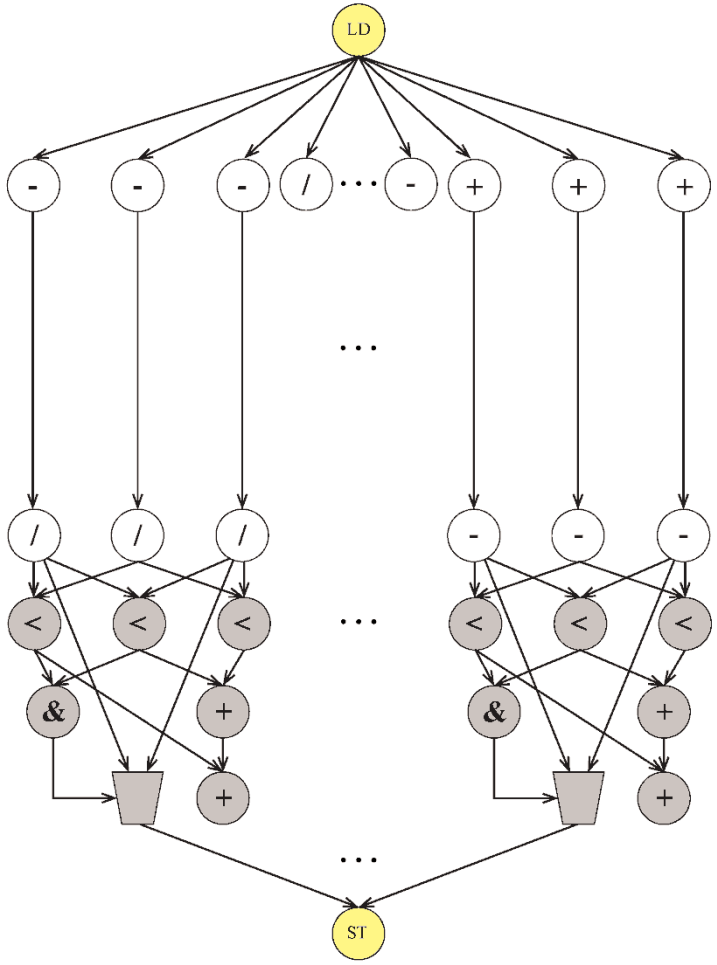


Base



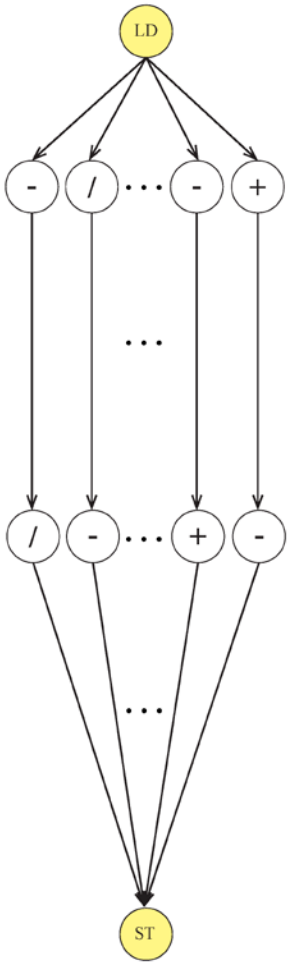
TMR with full voting

Total : 2000

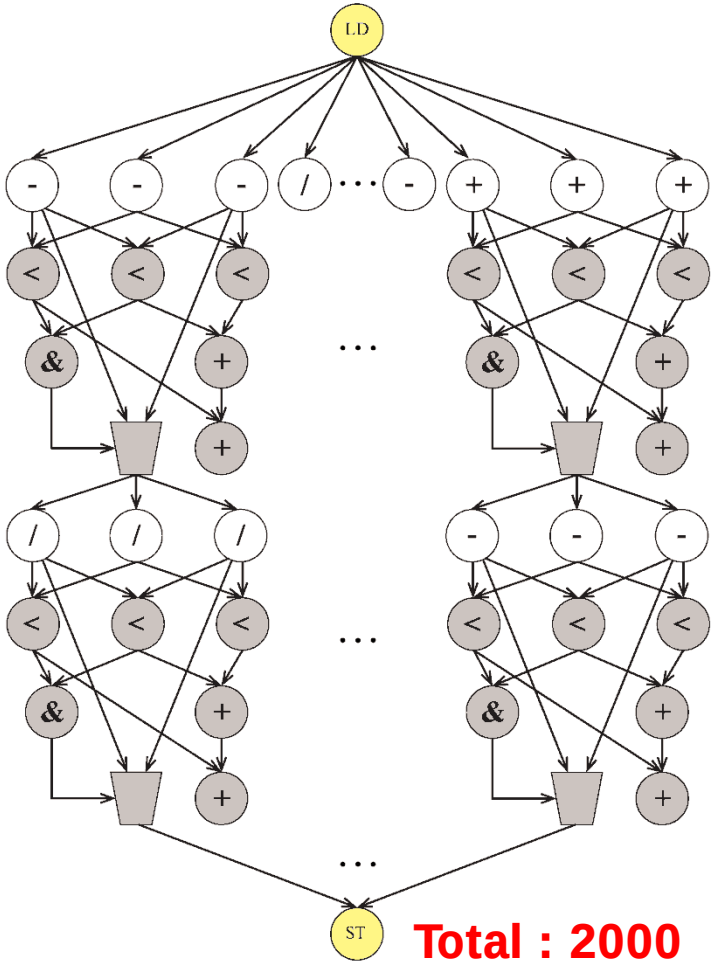


TMR with selective voting

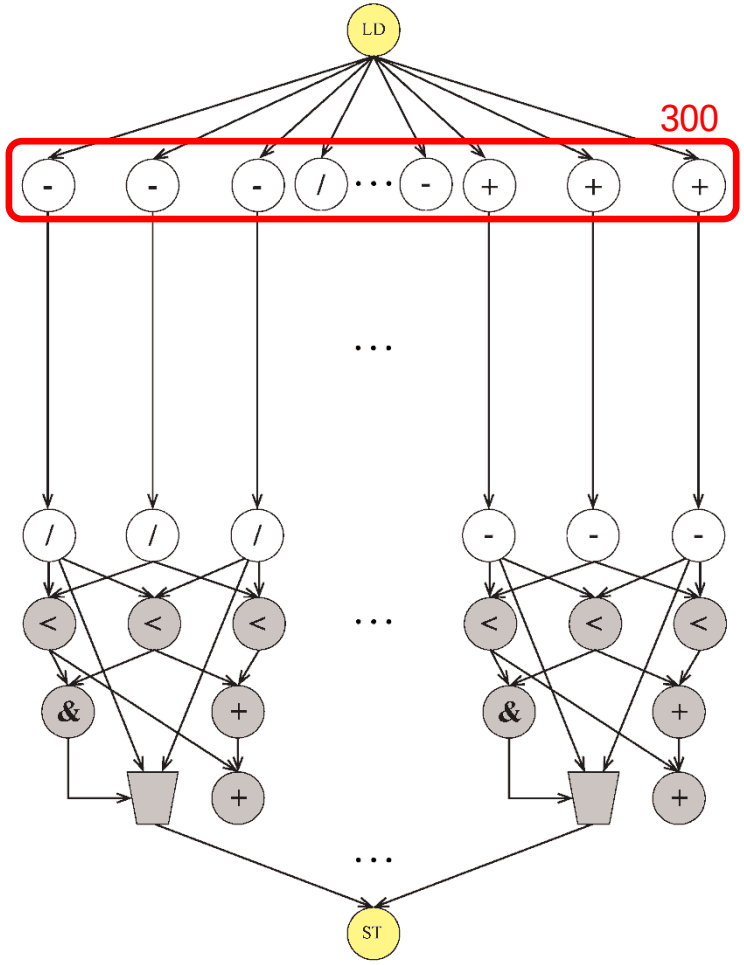
Ex) Reduce expensive voting



Base

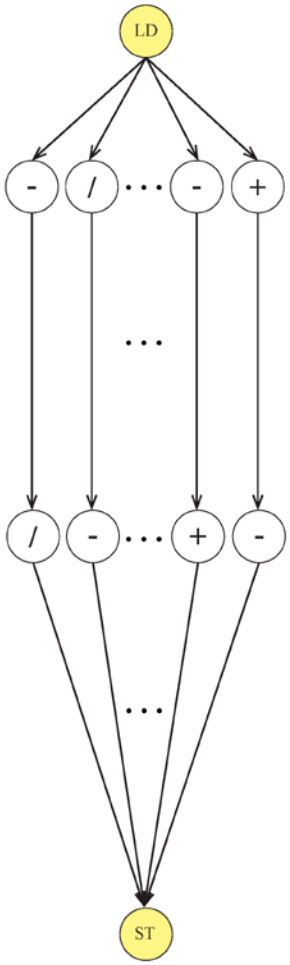


TMR with full voting

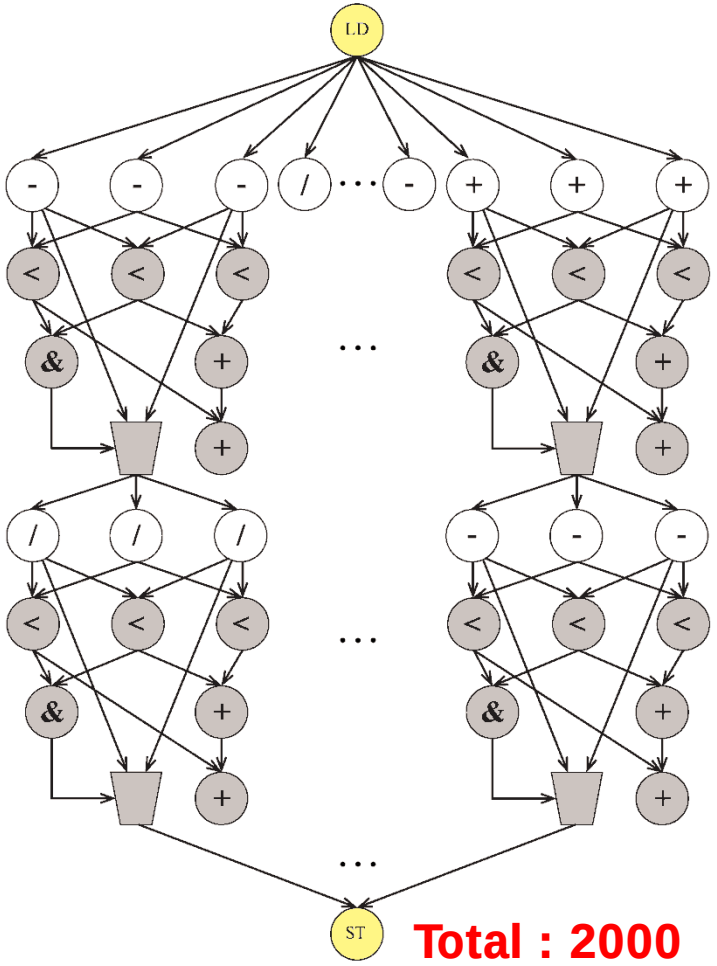


TMR with selective voting

Ex) Reduce expensive voting

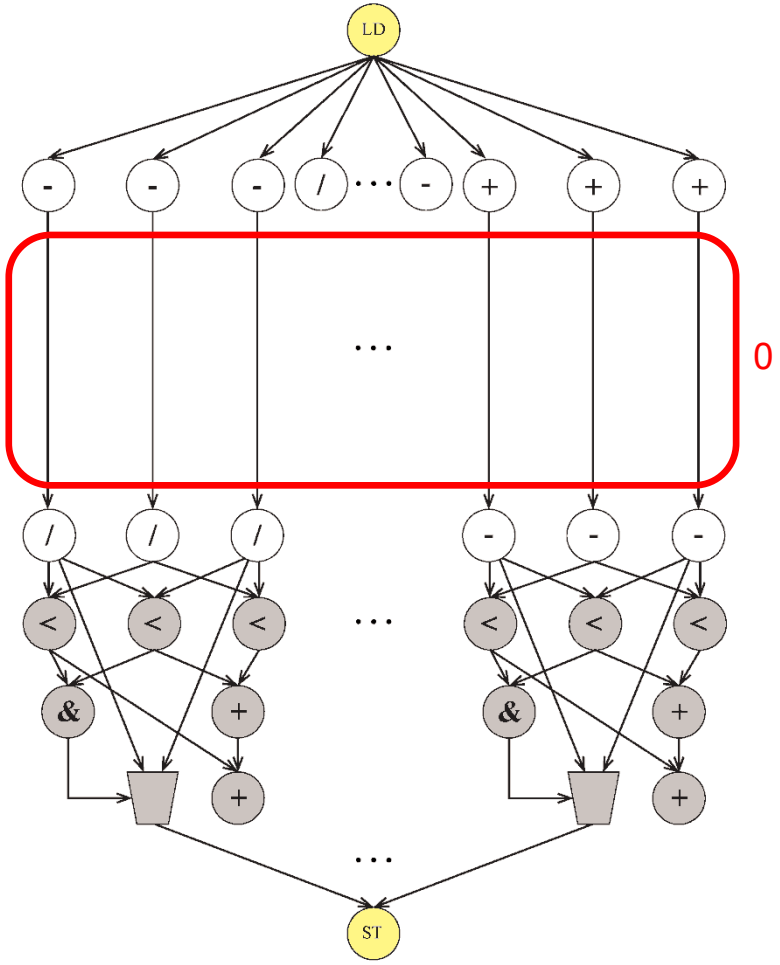


Base



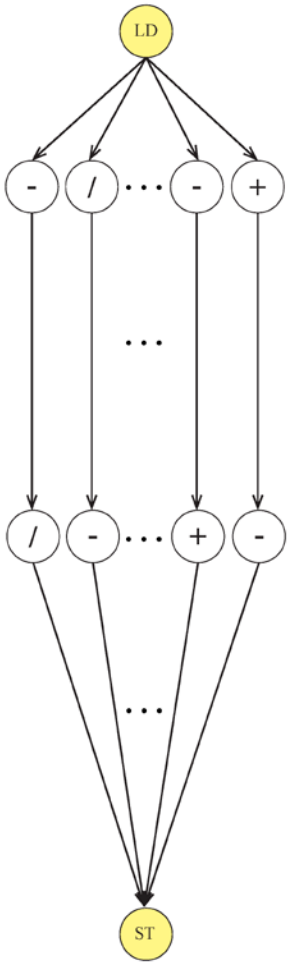
Total : 2000

TMR with full voting

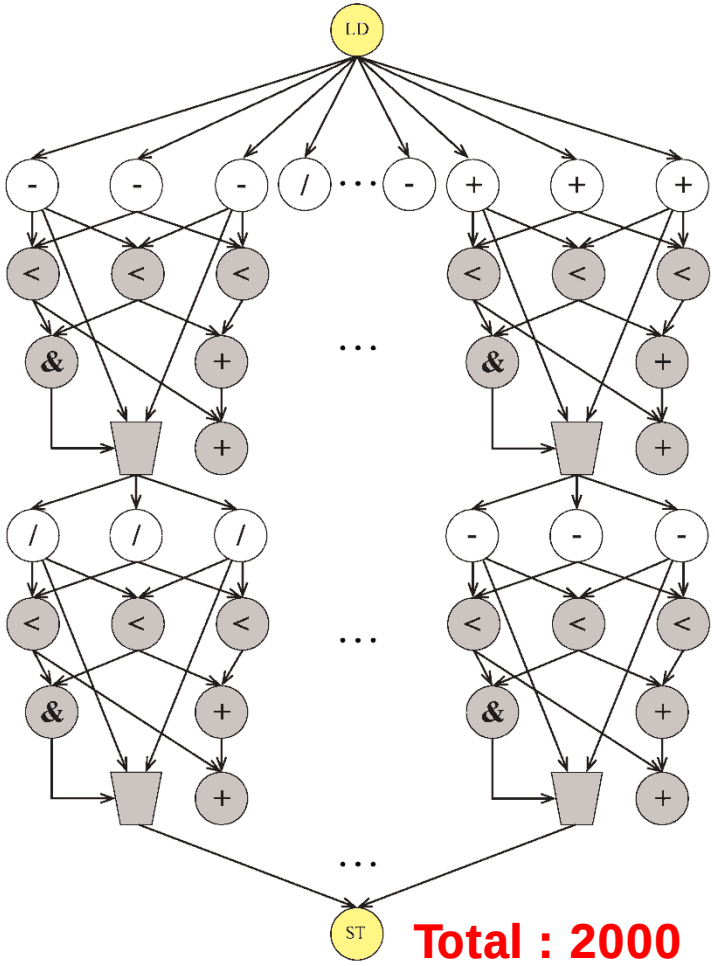


TMR with selective voting

Ex) Reduce expensive voting

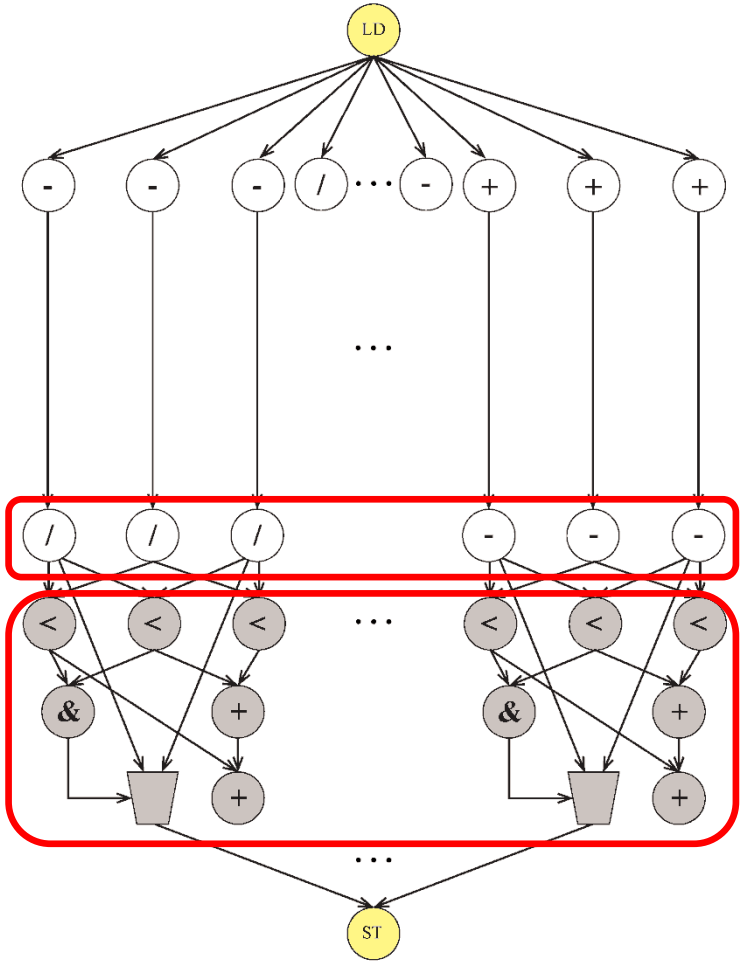


Base



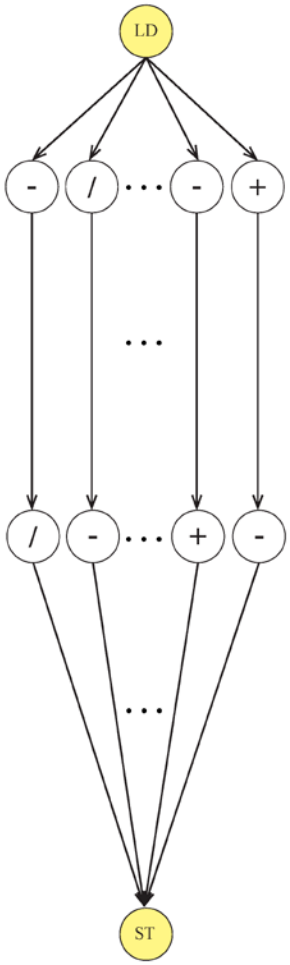
Total : 2000

TMR with full voting

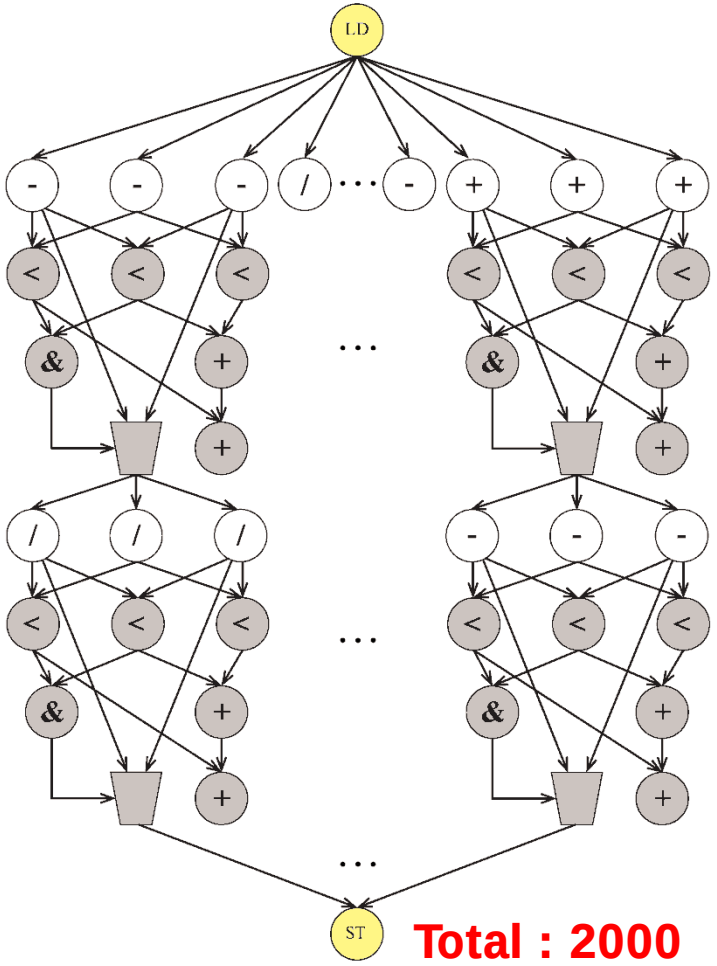


TMR with selective voting

Ex) Reduce expensive voting

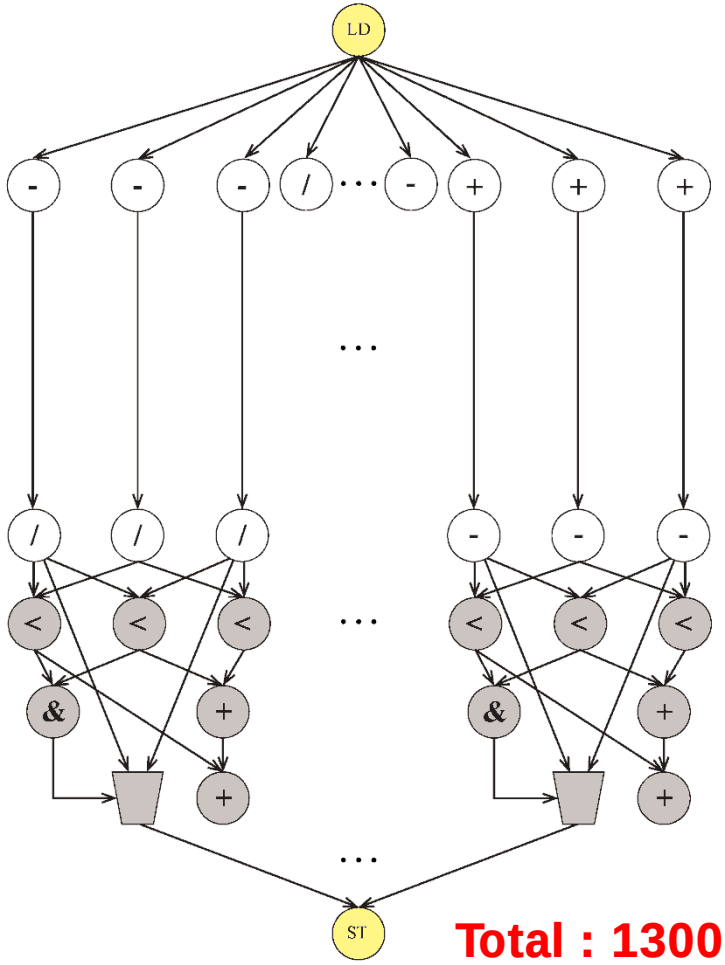


Base



Total : 2000

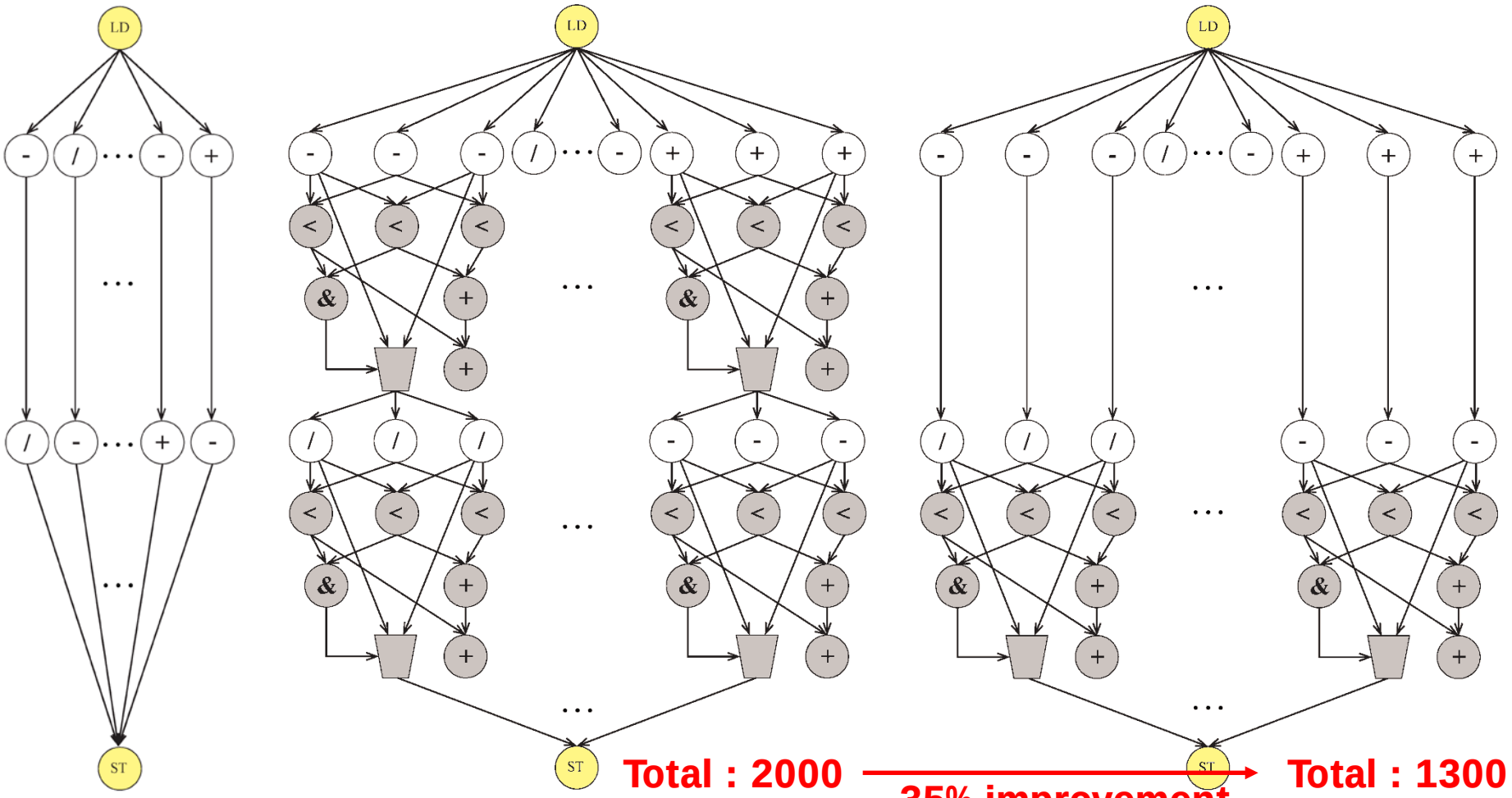
TMR with full voting



Total : 1300

TMR with selective voting

Ex) Reduce expensive voting

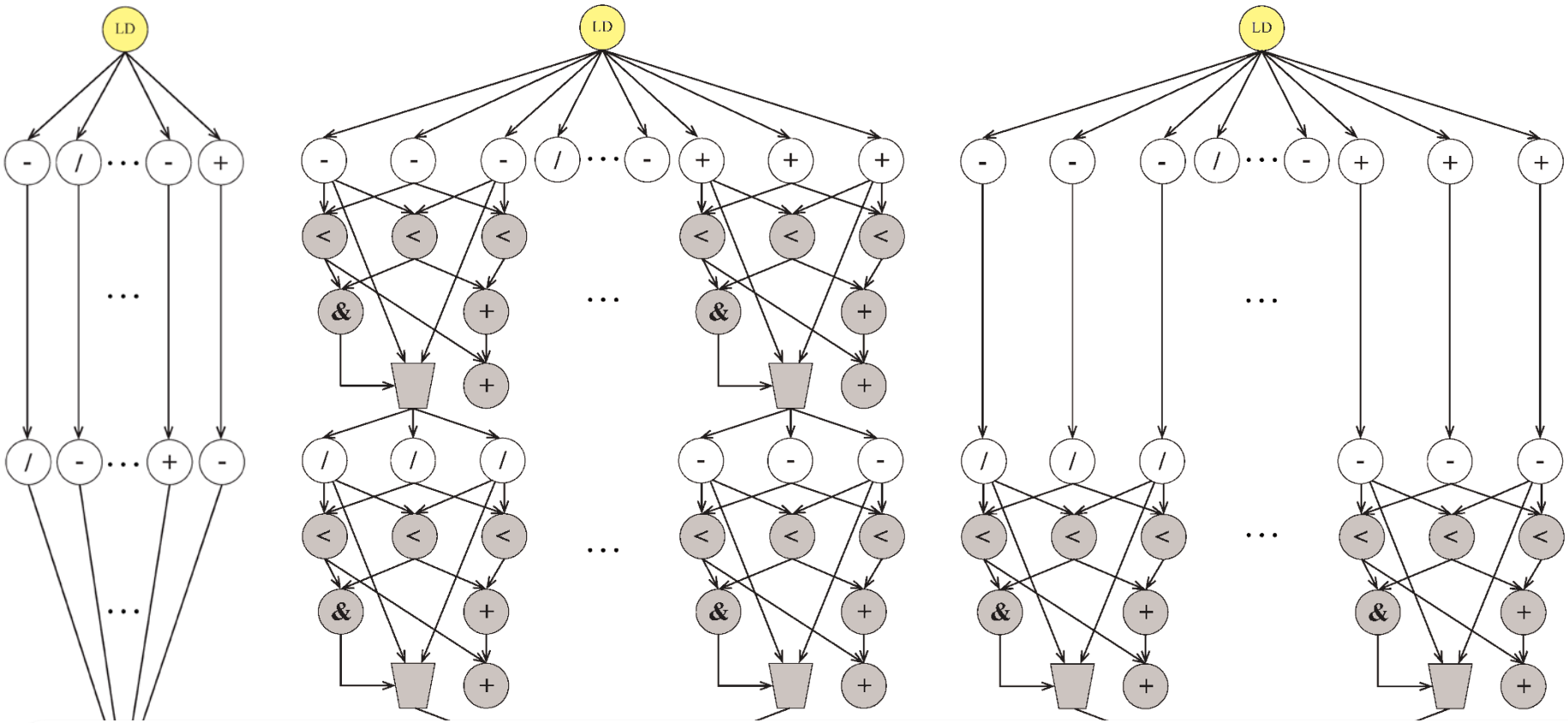


Base

TMR with full voting

TMR with selective voting

Ex) Reduce expensive voting



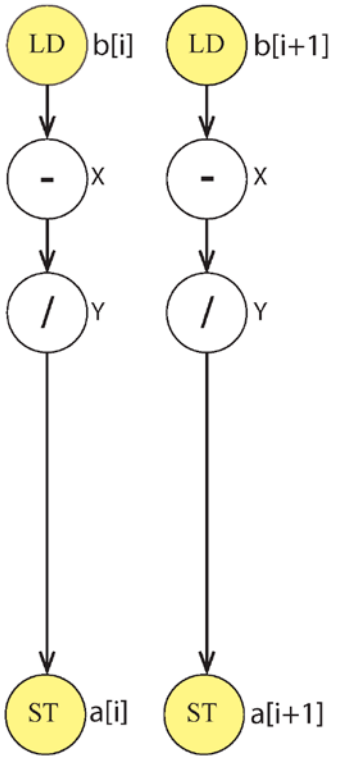
TMR with the selective voting is more efficient than TMR with full voting in terms of performance

```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
```

Optimization by reducing # of store operations

- Our optimization to reduce # of votings
  - Reduce the # of store operations by merging store operations with add and shift operations

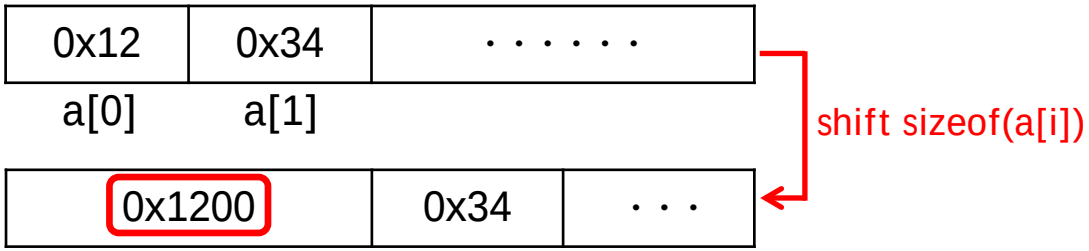
0x12	0x34	· · · · · · ·
a[0]	a[1]	

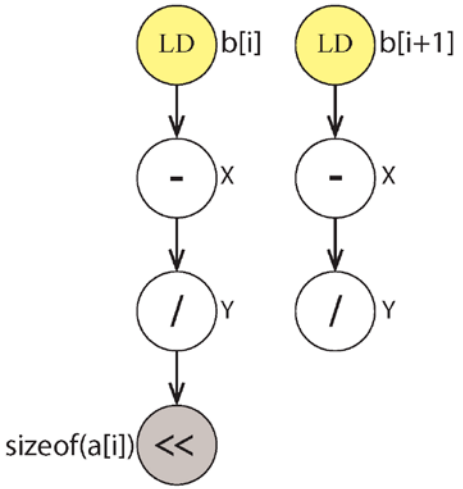


```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
```

Optimization by reducing # of store operations

- Our optimization to reduce # of votings
  - Reduce the # of store operations by merging store operations with add and shift operations

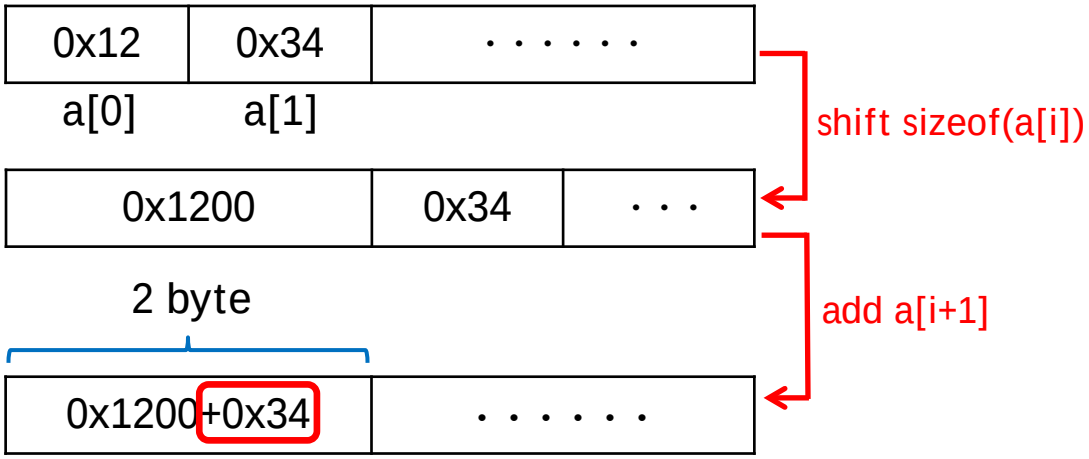


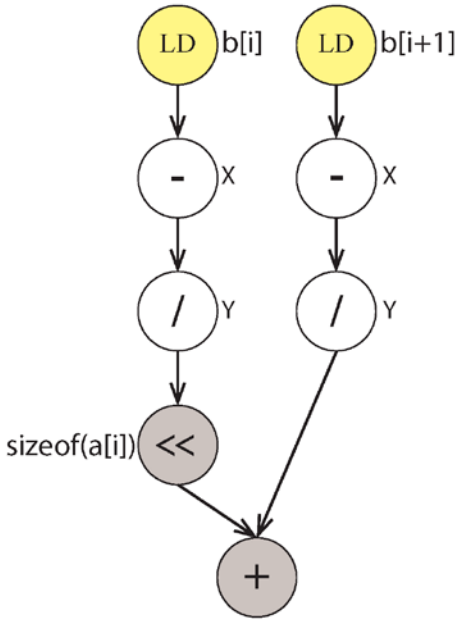


```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
```

Optimization by reducing # of store operations

- Our optimization to reduce # of votings
  - Reduce the # of store operations by merging store operations with add and shift operations

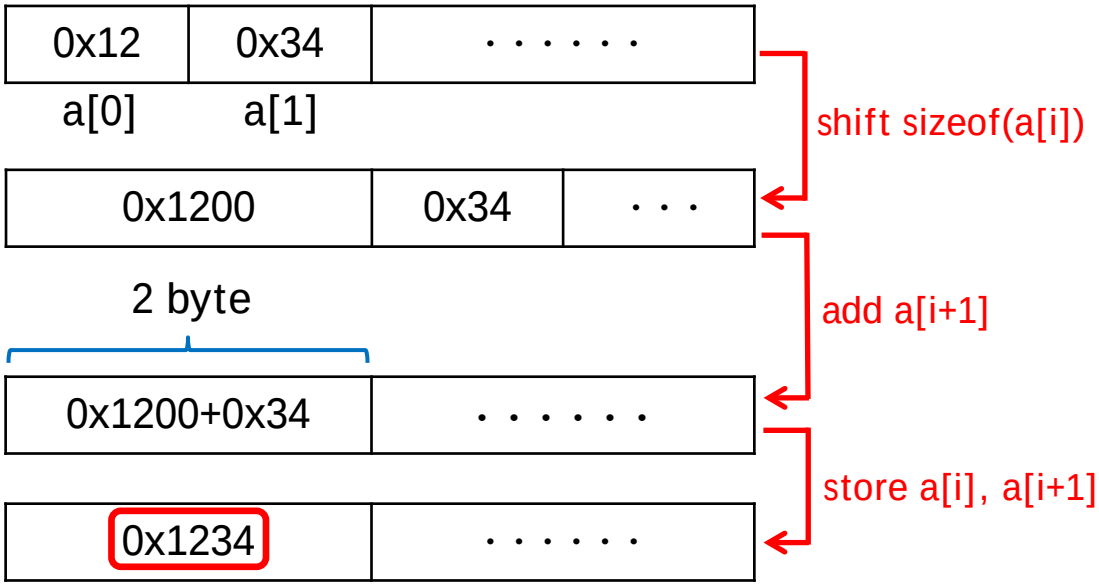


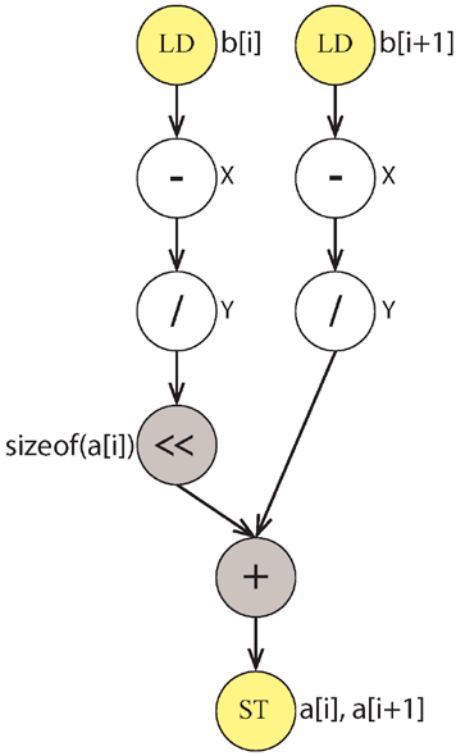


```
...
for ( i = 0; i < iteration; i ++) {
    a[i] = ( b[i] – X )  Y;    /* X and Y are constants */
}
```

Optimization by reducing # of store operations

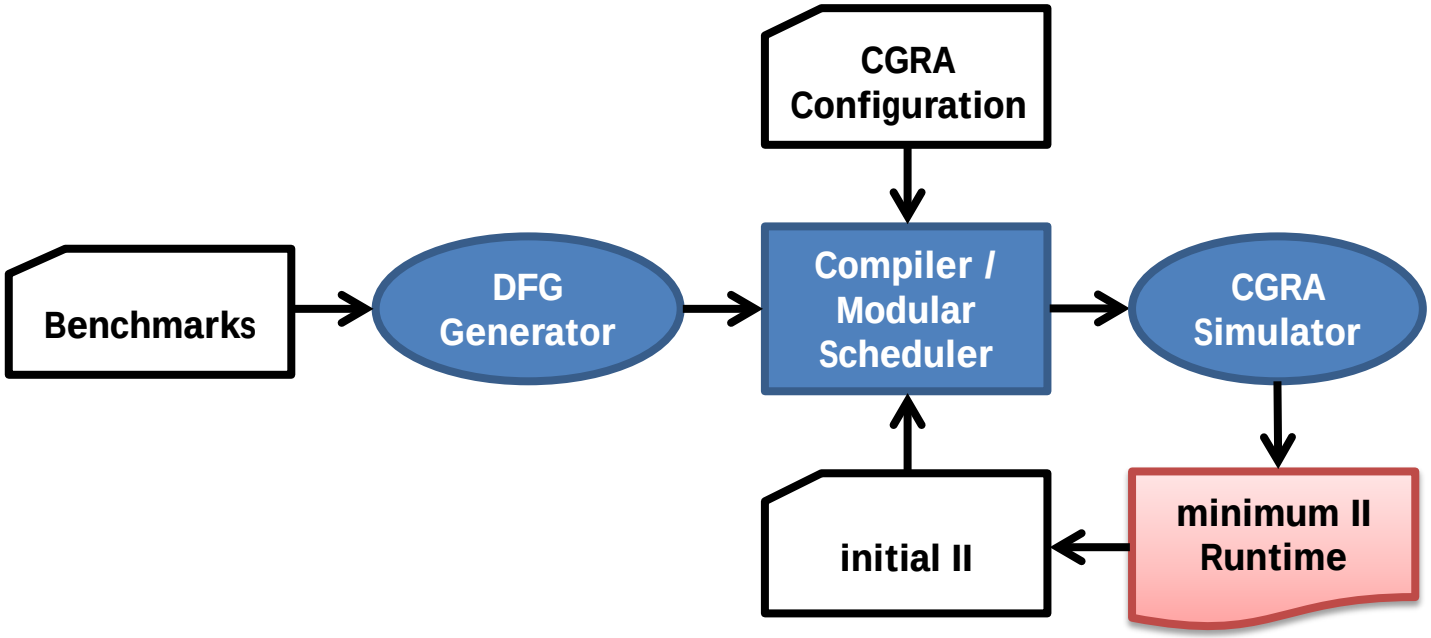
- Our optimization to reduce # of votings
  - Reduce the # of store operations by merging store operations with add and shift operations





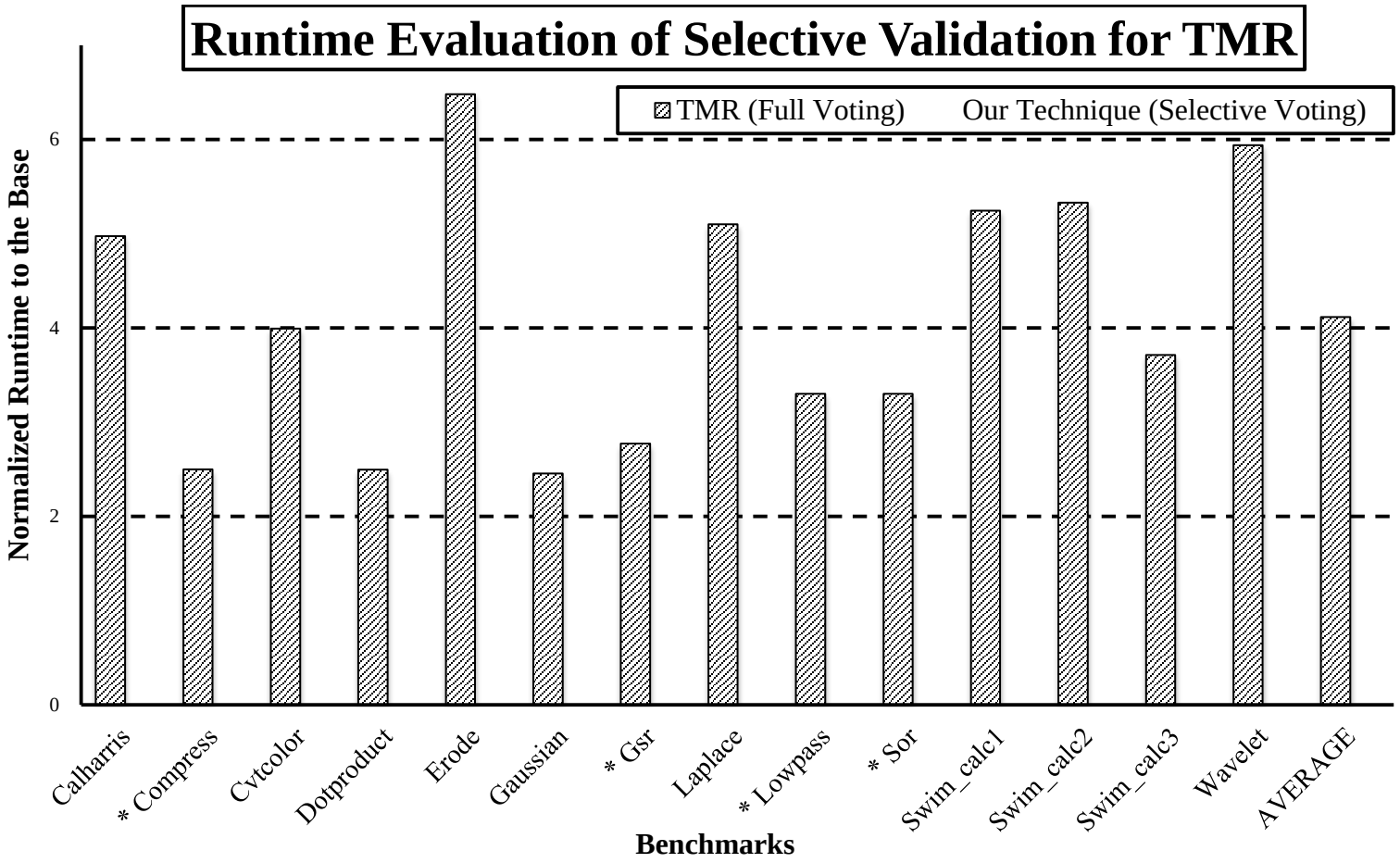
5. Experiments

Experimental setup

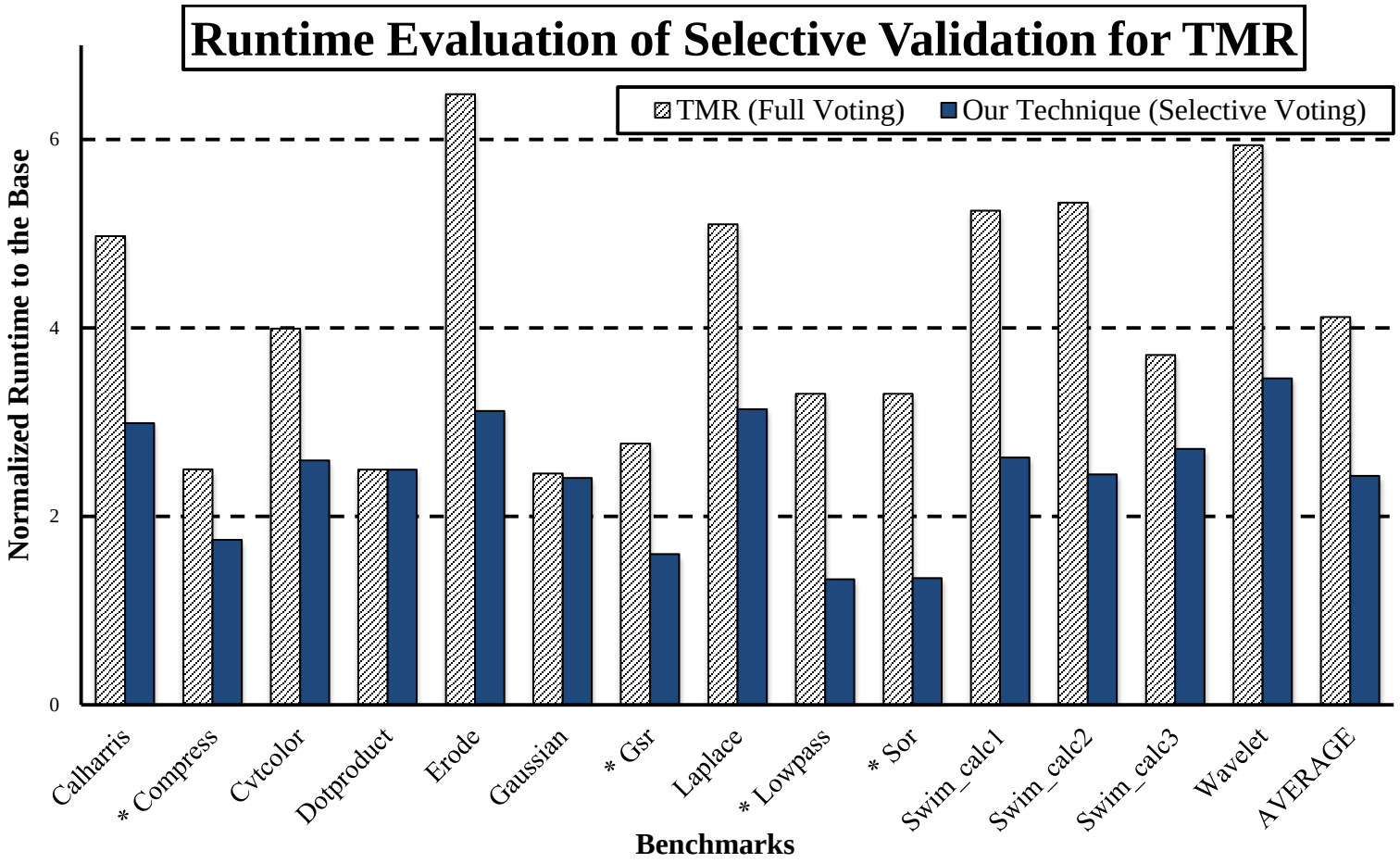


- compile and simulate each benchmark kernel with input parameters
  - Input parameters: DFG information, CGRA configuration, initial II
    - DFG information : generated by DFG Generator
    - CGRA configuration : 4 X 4 CGRA, Local register
    - II : Initiation Interval
- Benchmarks : SPEC 2000, OpenCV
- Scheduler : modulo scheduling (the cost-based scheduling algorithm)

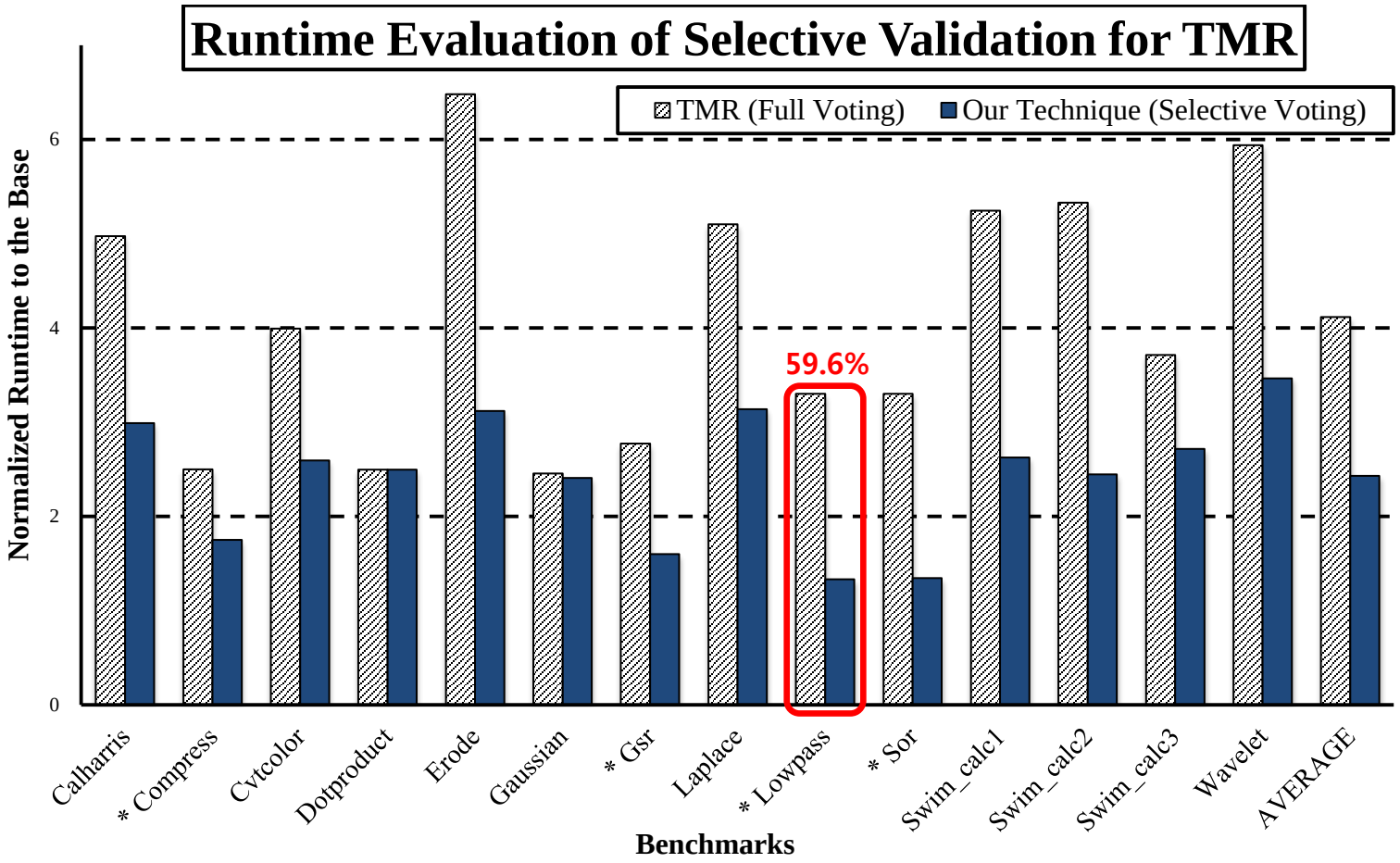
Performance improvement w/ selective voting



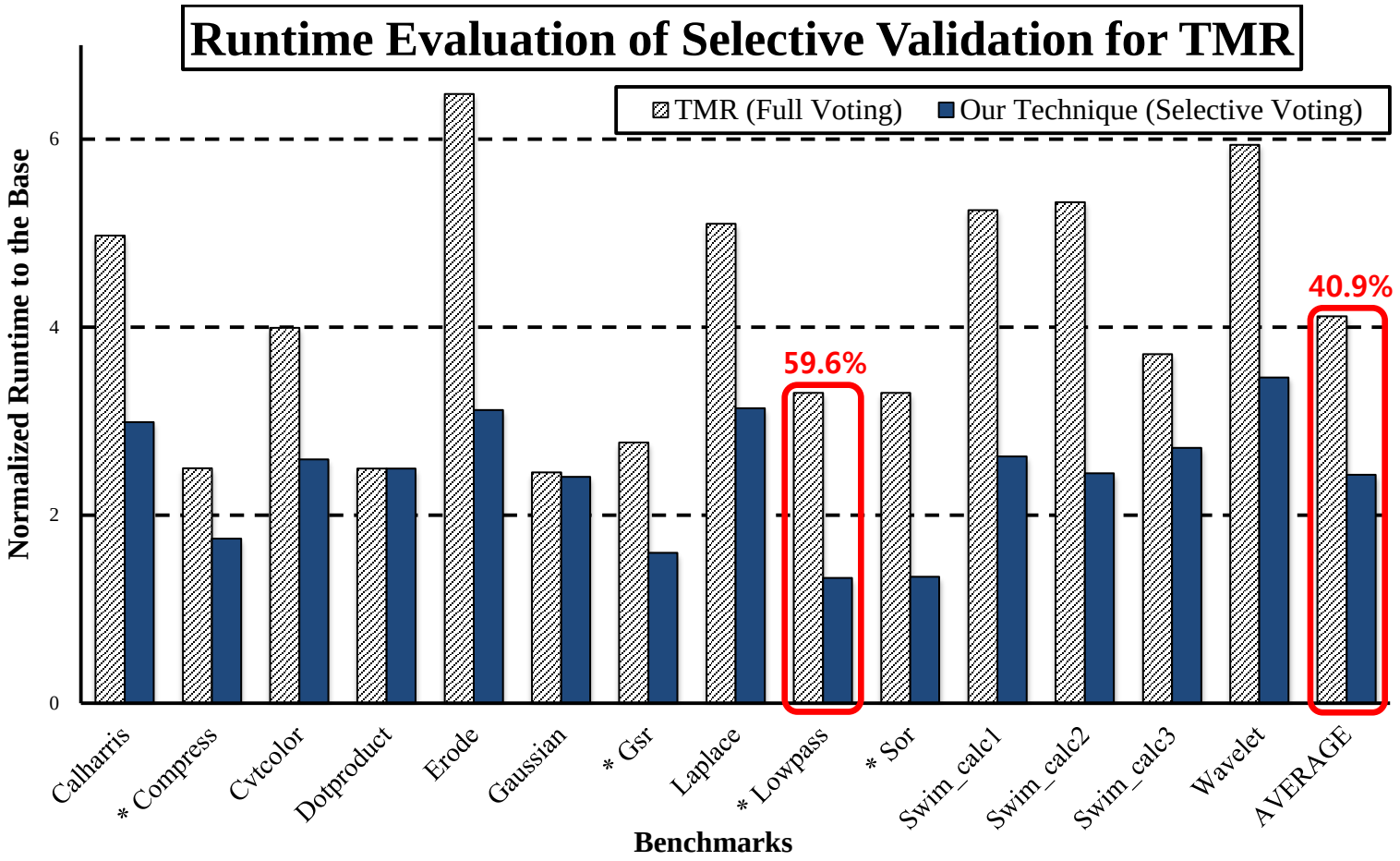
Performance improvement w/ selective voting



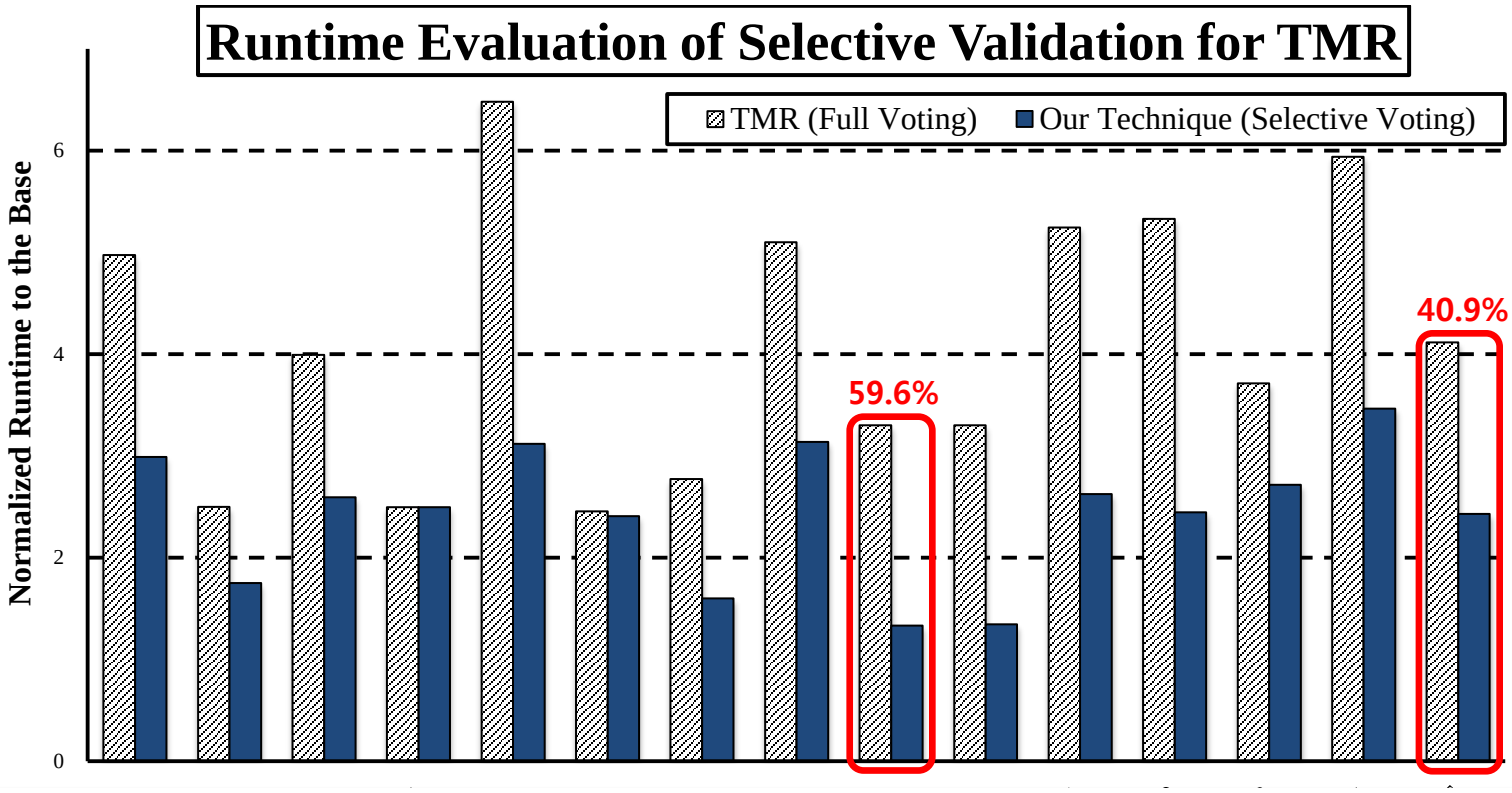
Performance improvement w/ selective voting



Performance improvement w/ selective voting

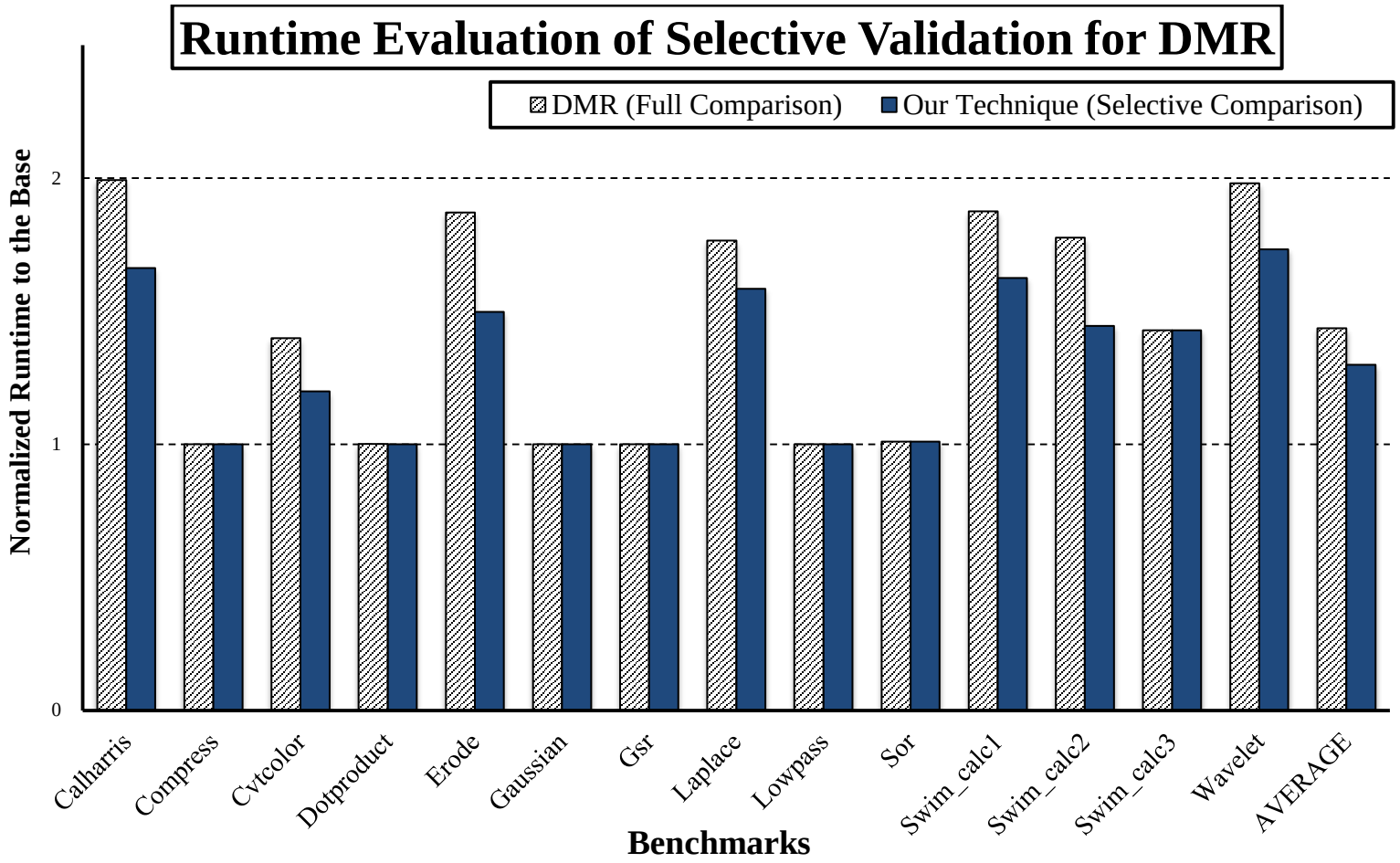


Performance improvement w/ selective voting

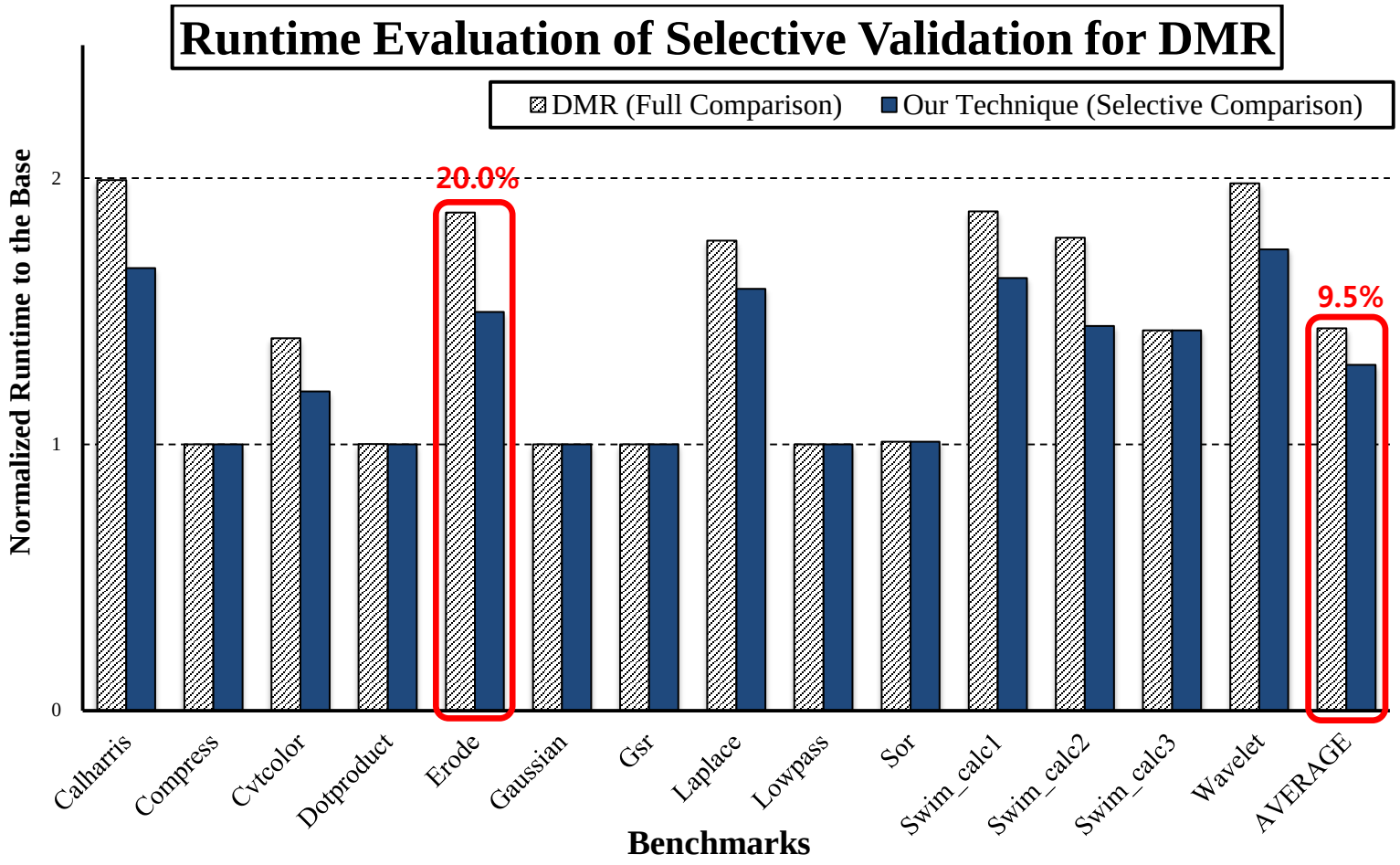


Our selective validation for TMR can improve the performance

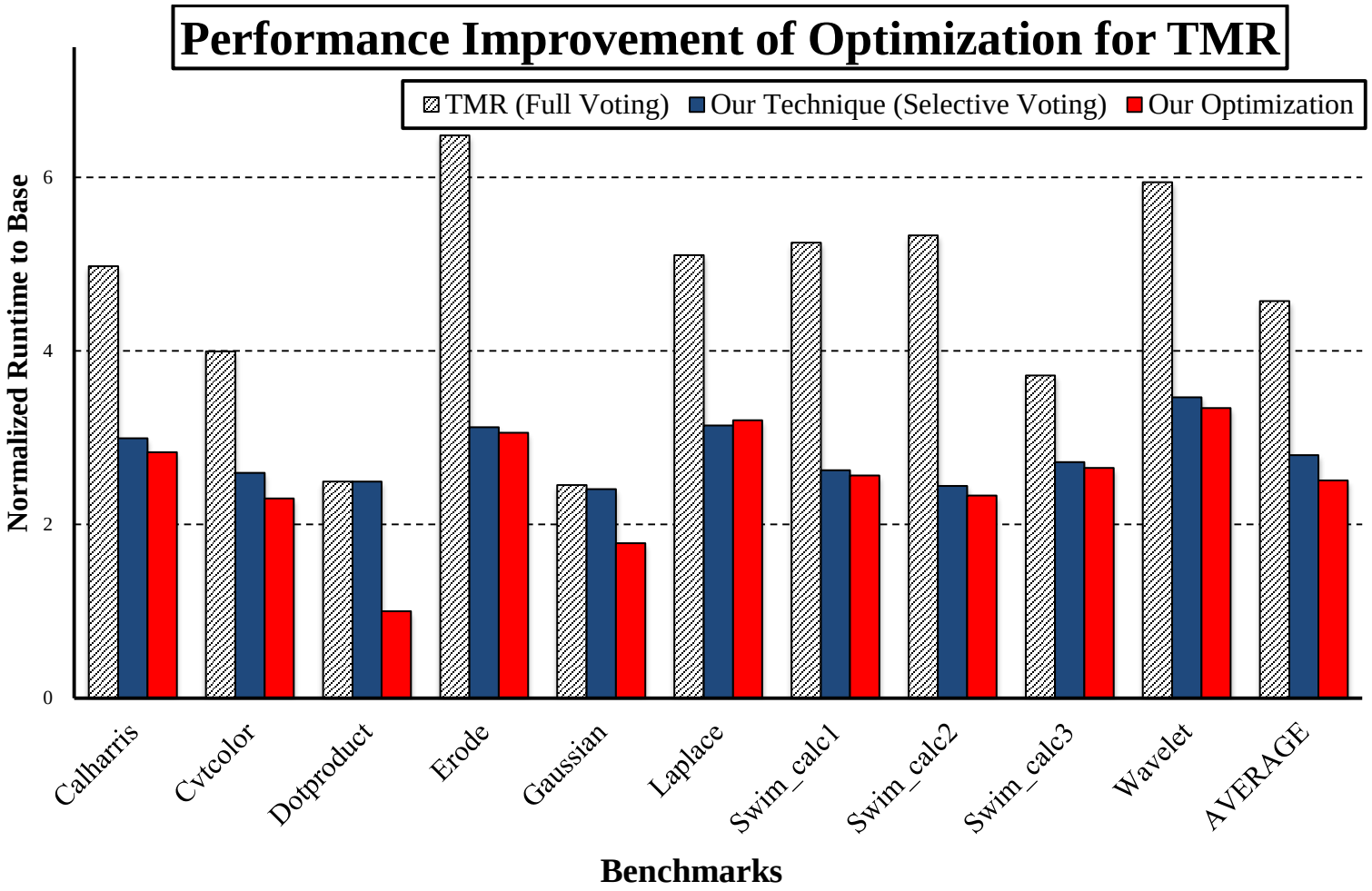
Performance improvement w/ selective voting



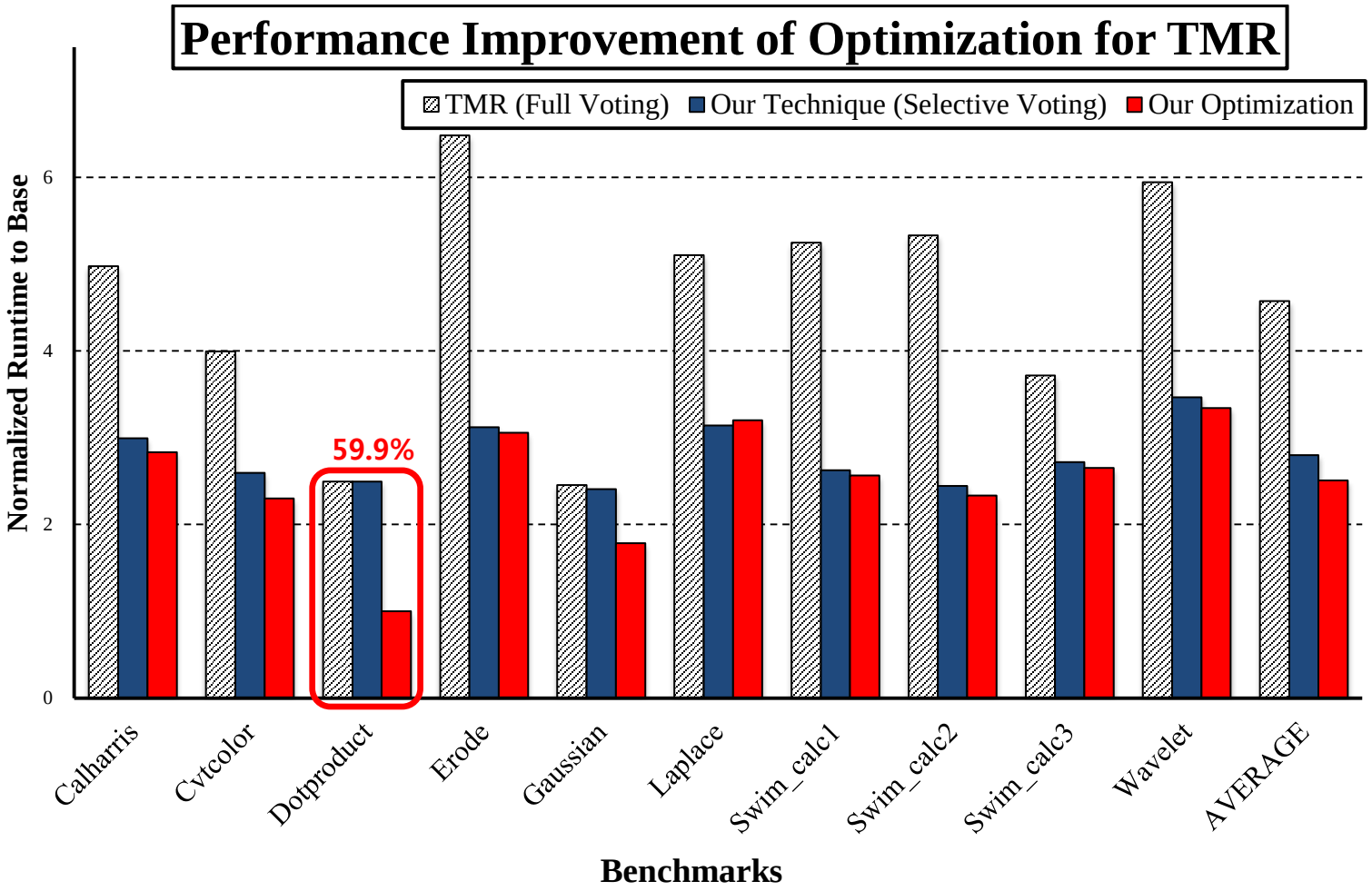
Performance improvement w/ selective voting



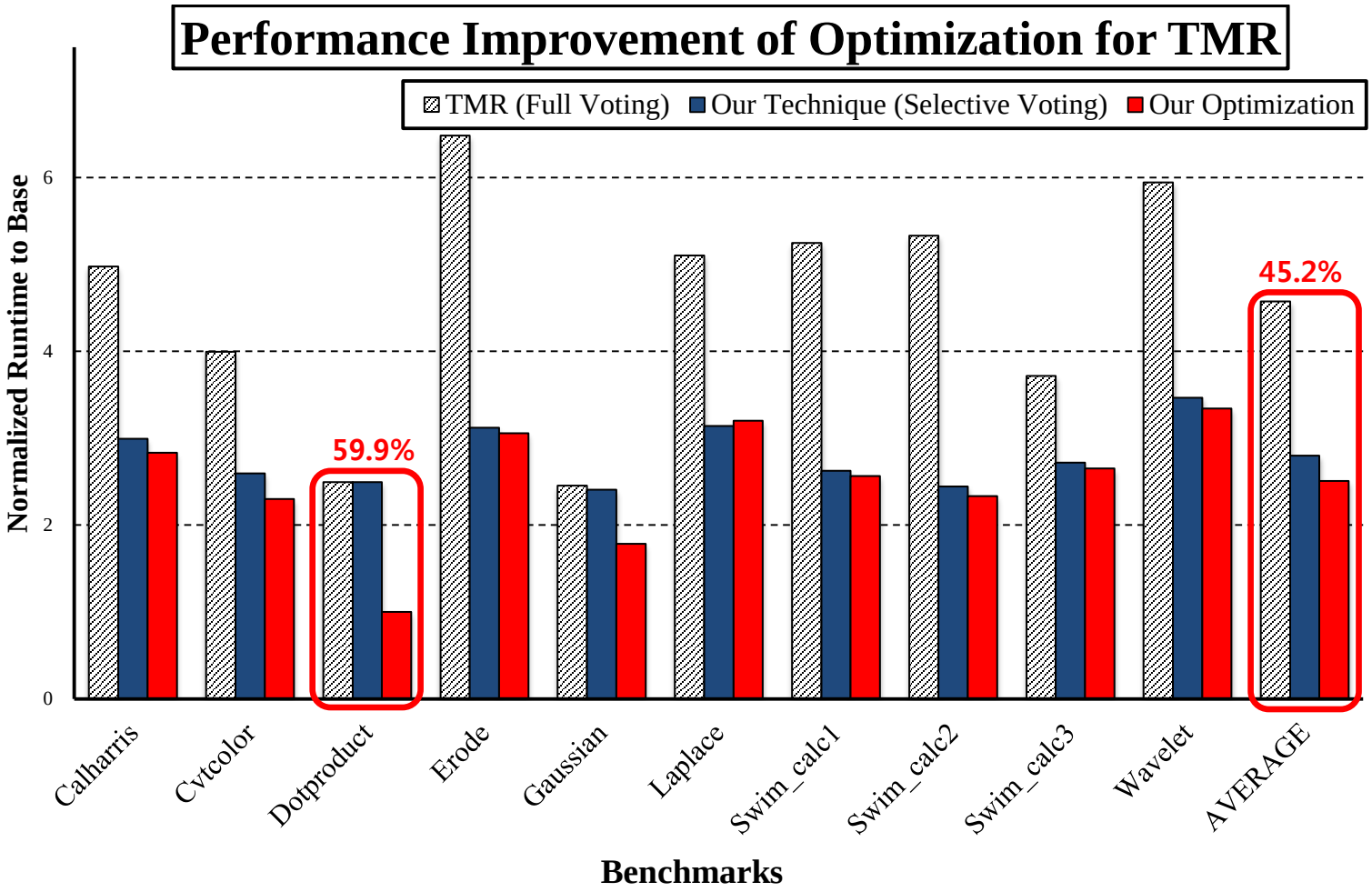
Performance optimization w/ store reduction



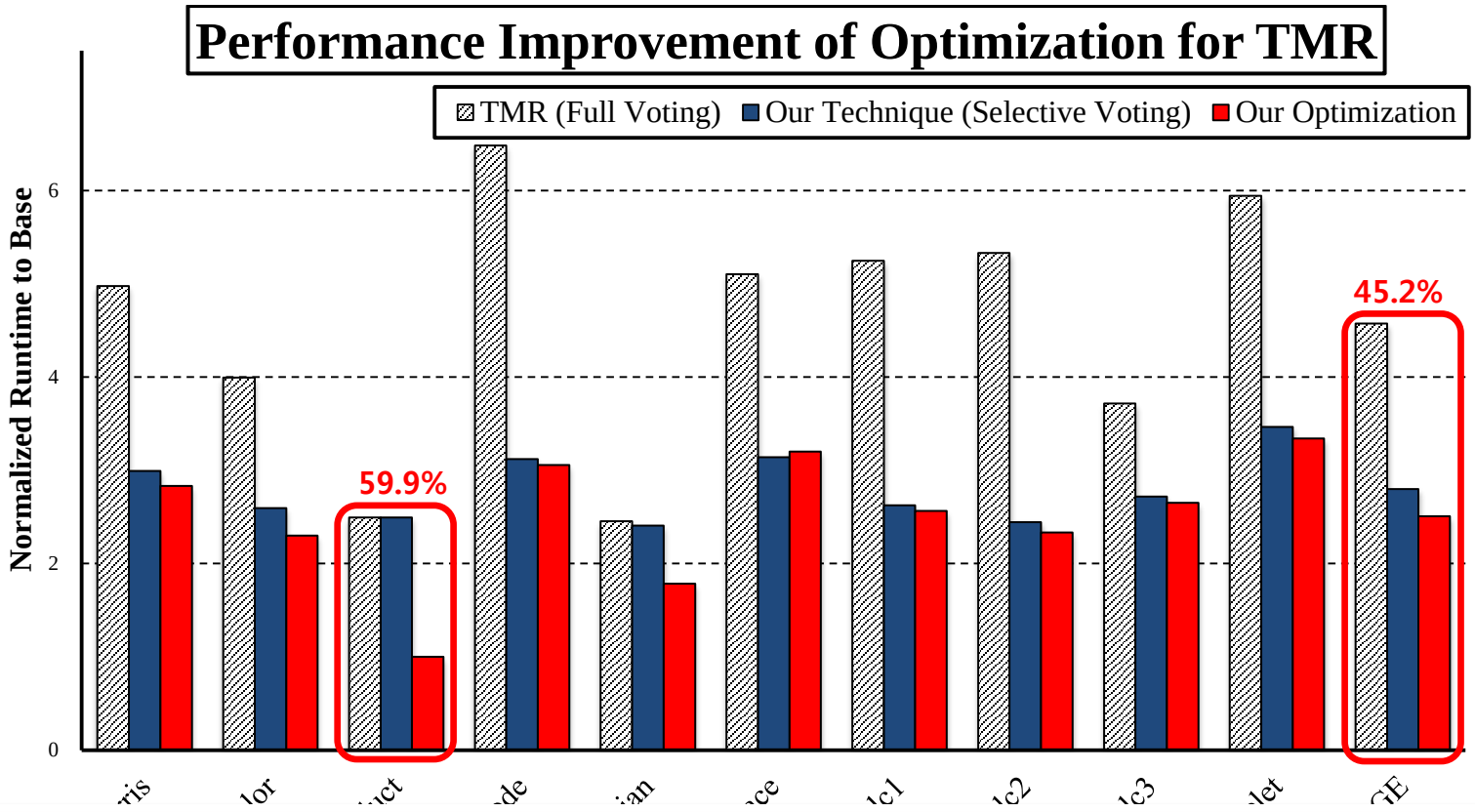
Performance optimization w/ store reduction



Performance optimization w/ store reduction

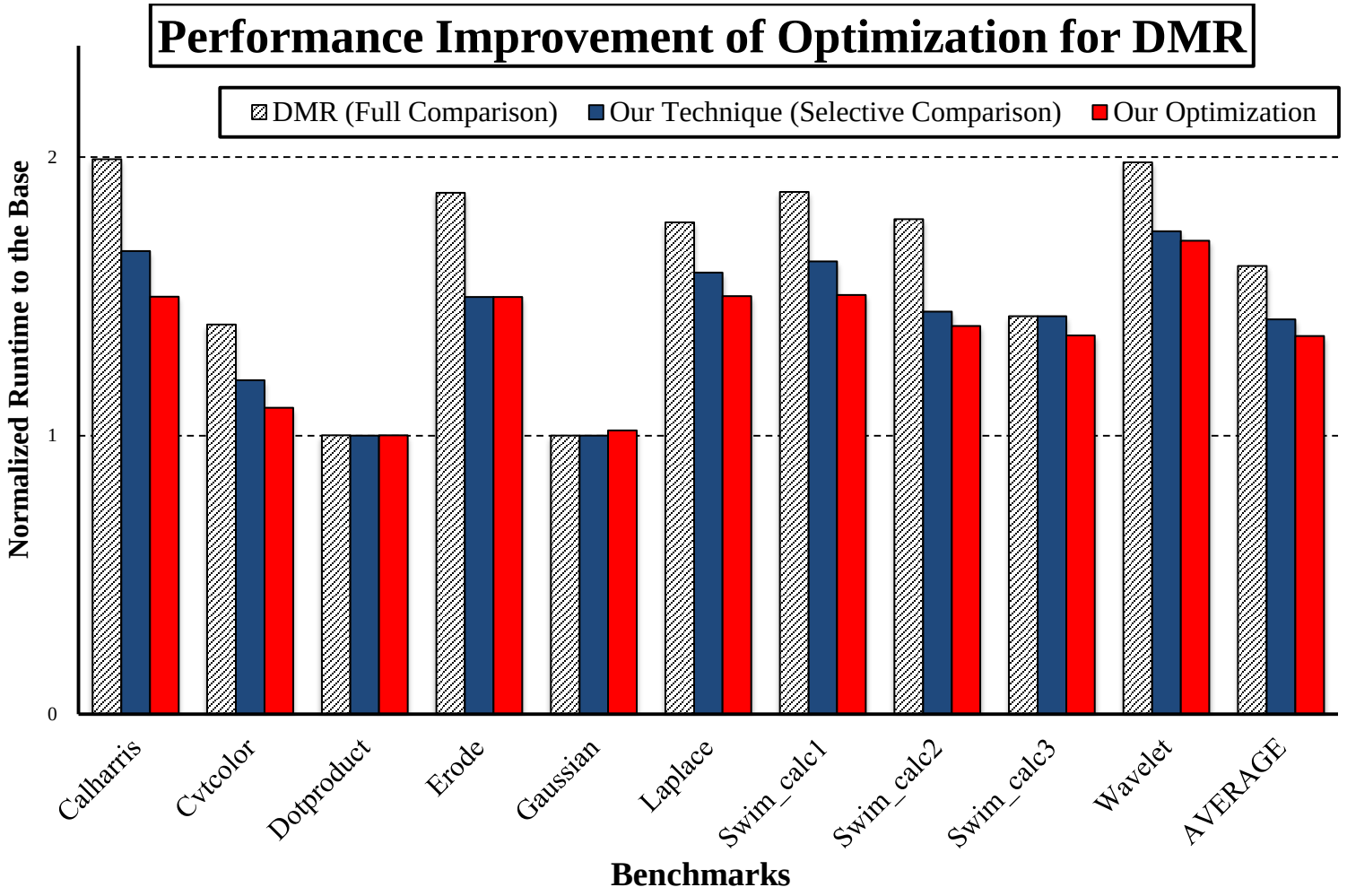


Performance optimization w/ store reduction

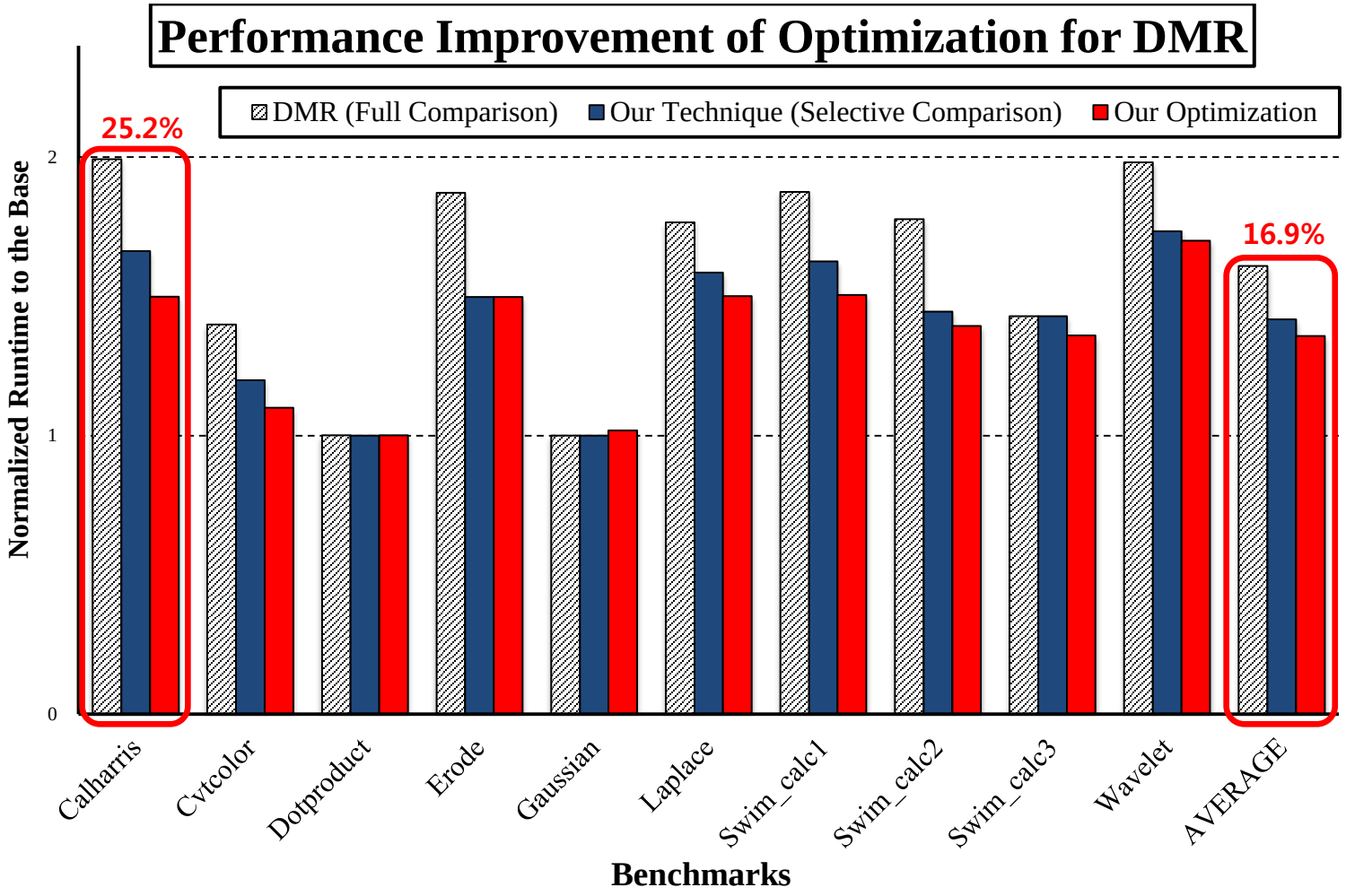


Our optimization with store reduction can further improve the performance as compared to our selective validation

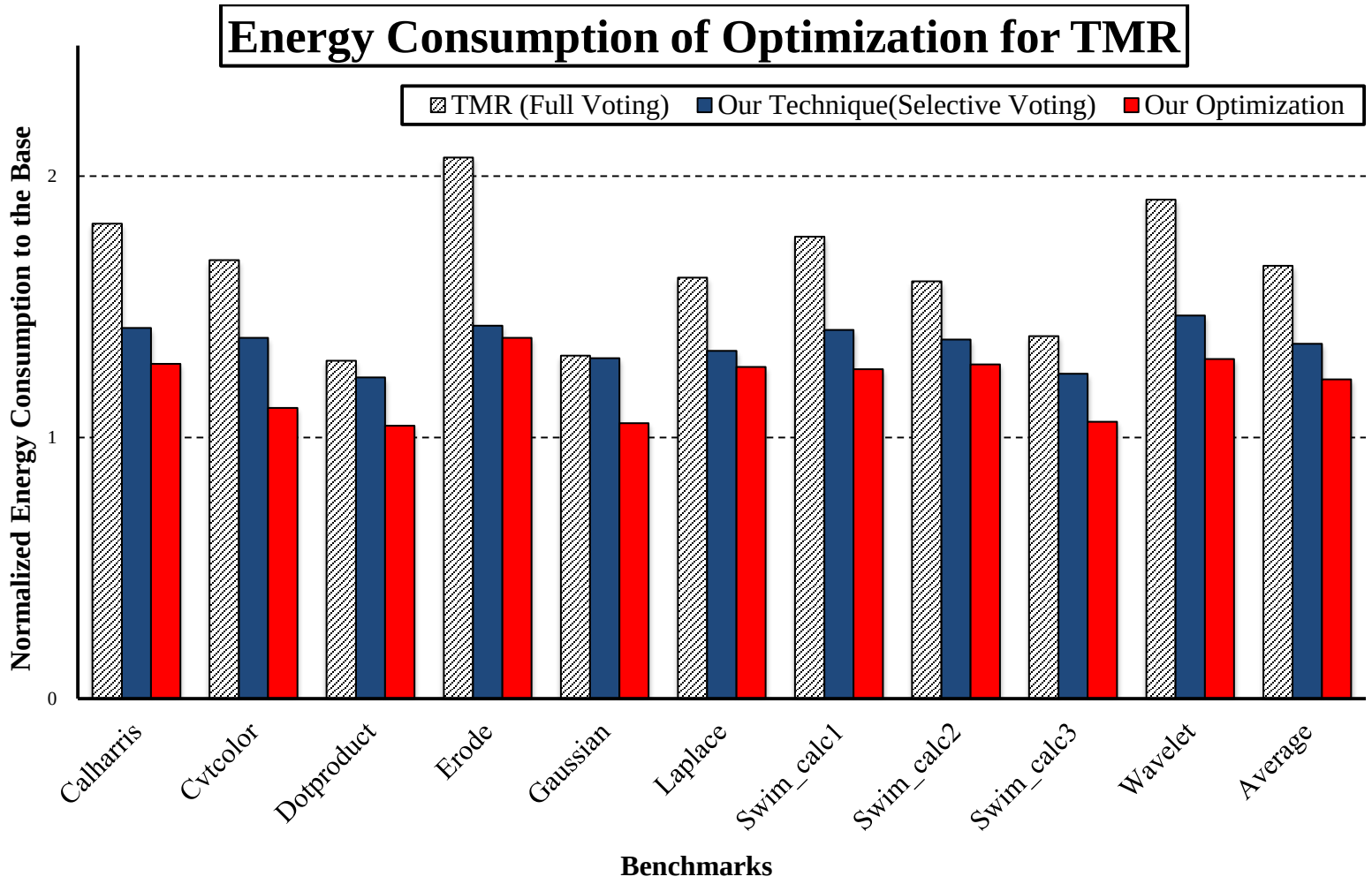
Performance optimization w/ store reduction



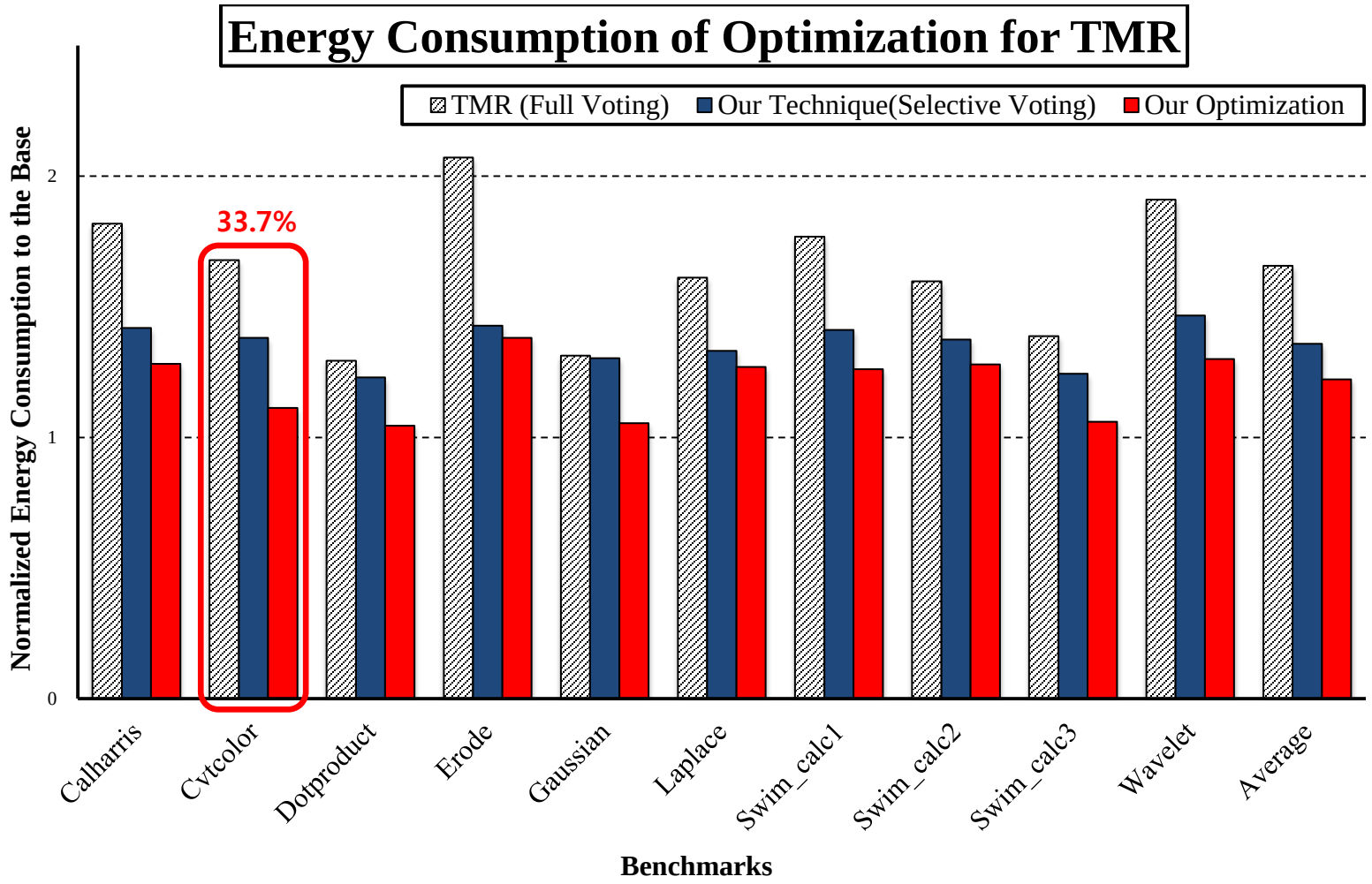
Performance optimization w/ store reduction



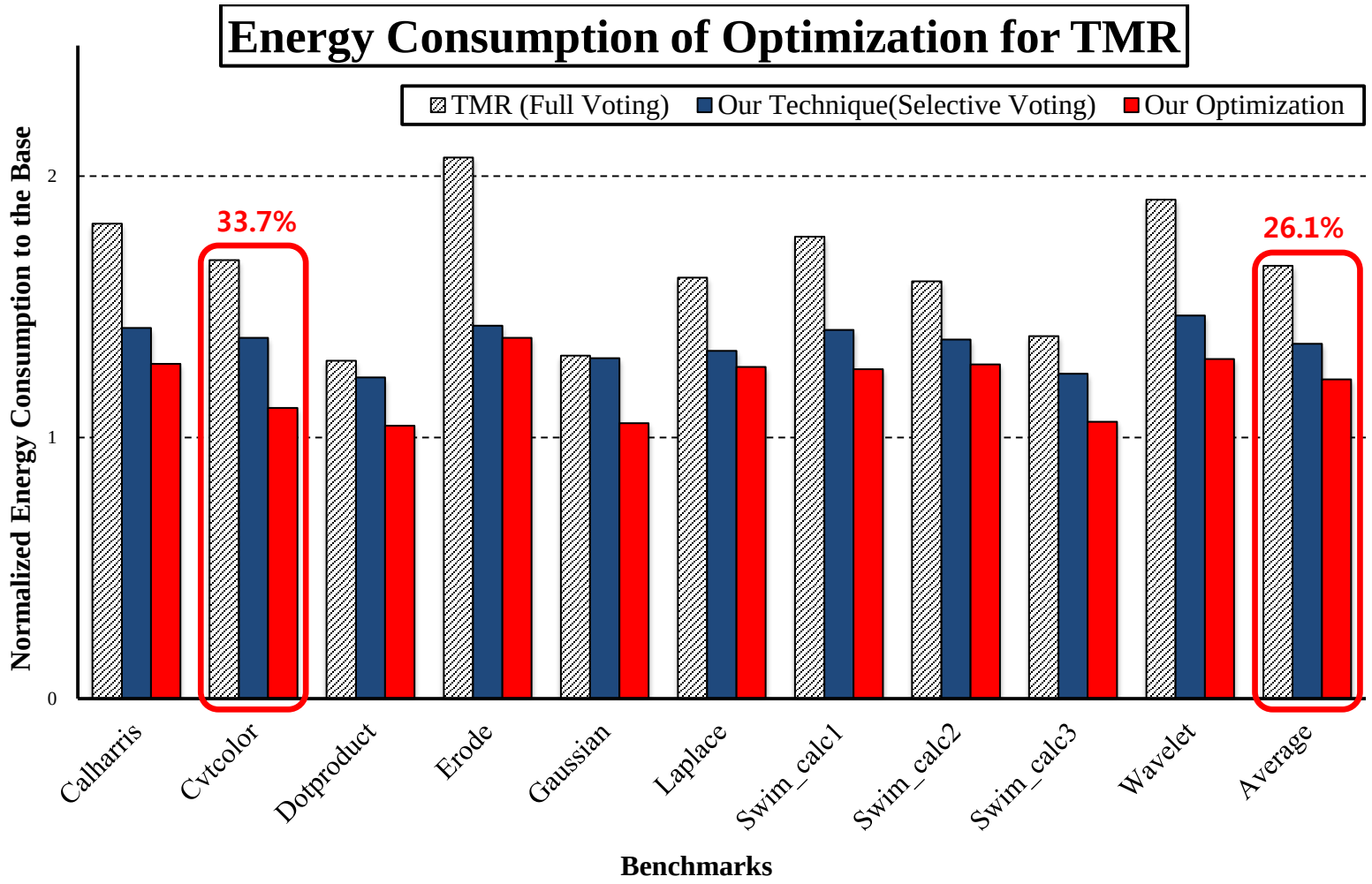
Energy consumption w/ optimization



Energy consumption w/ optimization



Energy consumption w/ optimization



Conclusions

- Reliability is the critical design concern
- CGRA becomes more popular
- Previous proposed techniques have limitations in terms of area and performance
- Software-based selective validation techniques
  - Validation at operation just before store
- Performance improvement
  - TMR/DMR with selective validation : 40.9%/9.5%
  - TMR/DMR with selective validation + optimization : 45.2%/16.9%

  ※ Previous proposed DMR/TMR techniques : 167%/700%
- Energy consumption savings
  - TMR/DMR with selective validation + optimization : 26.1%/13.8%

Future work

- Continue energy consumption
- Design space exploration
  - Tradeoff among reliability, energy consumption and performance
- Further optimization
  - CGRA scheduling algorithm

Q & A

Publication

- Paper accept
  - Selective validations for efficient protections on coarse-grained reconfigurable architectures
  - Application-specific Systems, Architectures and Processors(ASAP)
  - The 24th IEEE International Conference, June 5-7 2013
- Plan to submit extended version journal(TECS)
  - Energy consumption estimation
  - Fault coverage
- Plan to submit a paper KCC Conference

Reference

- Baumann et al. “Determining the Impact of Alpha-Particle-Emitting Contamination From the Fukushima-Daiichi Disaster on Japanese Semiconductor Manufacturing Sites” 2012 RADECS.
- Toshinori et al. “Radiation Effect Mitigation Methods for Electronic Systems” 2012 IEEE SII .
- Jafri et al. “Design of a faulttolerant coarse-grained reconfigurable architecture: a case study,” in ISQED, 2010.
- Alnajiar el al. “Coarse-grained dynamically reconfigurable architecture with flexible reliability,” in FPL, 2009.
- Schweizer el al. “Using run-time reconfiguration to implement fault-tolerant coarse grained reconfigurable architectures,” in IPDPSW, 2012.
- Schweizer el al “Low-cost TMR for fault-tolerance on coarse-grained reconfigurable architectures,” in ReConFig, 2011.
- Reis el al. “SWIFT: Software implemented fault tolerance,” in CGO, 2005.
- K.Lee el al. “Mitigating the impact of hardware defects on multimedia applications: a cross-layer approach,” in ACM Multimedia, 2008.
- G. Lee et al. “Thermal-aware fault-tolerant system design with coarse-grained reconfigurable array architecture,” in AHS, 2010.
- G. Bradski, “The OpenCV library,” Doctor Dobbs Journal, 2000.
- J. L. Henning, “SPEC CPU2000: Measuring CPU performance in the new millennium,” Computer, 2000.
- http://technology-and-science.lawyers.com/blogs/archives/4461-Are-Cosmic-Rays-a-Factor-in-Toyota-Acceleration-Problems.html
- http://www.chipdesignmag.com/payne/2010/04/14/alpha-particles-dram-seu-toyota-acceleration/