September 26th 2017 – Grenoble, France

PAOLO RECH

Radiation Reliability Issues in Current and Future Supercomputers

INSTITUTO DE INFORMÁTICA

UFRGS

Sponsors

Los Alamos NATIONAL LABORATORY

CAPES

HPC reliability importance

IEEE SPECTRUM

Follow on:

Engineering Topics | Special Reports | Blogs | Multime

Feature | Computing | Hardware

How To Kill A Supercomputer: Dirty Power, Cosmic Rays, and Bad Solder

Will future exascale supercomputers be able to withstand the steady onslaught of routine faults?

By Al Geist
Posted 23 Feb 2016 | 16:00 GMT

As a child, were you ever afraid that a monster lurking in your bedroom would leap out of the dark and get you? My job at Oak Ridge National

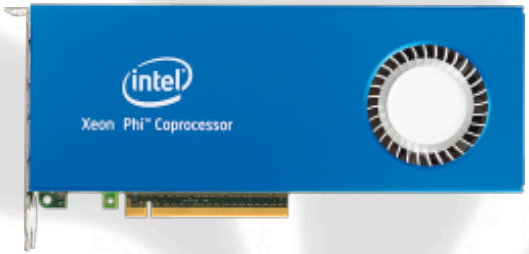
Available Accelerators

Modern parallel accelerators offer:

- Low cost
- Flexible platform
- High efficiency (low per-thread consumption)
- High computational power and frequency
- Huge amount of resources

NVIDIA Kepler K40

intel Xeon-Phi



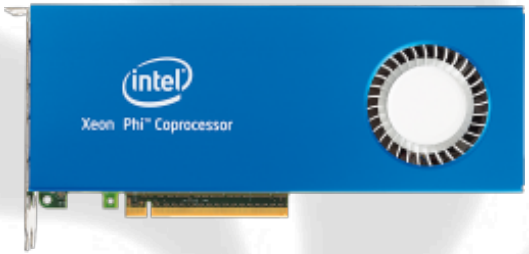
Available Accelerators

Modern parallel accelerators offer:

- Low cost
- Flexible platform
- High efficiency (low per-thread consumption)
- High computational power and frequency
- Huge amount of resources
- **Reliability?**

NVIDIA Kepler K40

intel Xeon-Phi



Available Accelerators

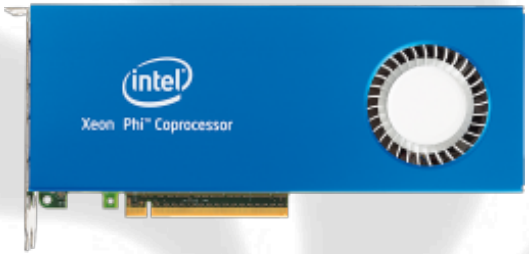
Modern parallel accelerators offer:

- Low cost
- Flexible platform
- High efficiency (low per-thread consumption)
- High computational power and frequency
- Huge amount of resources
- **Reliability?**

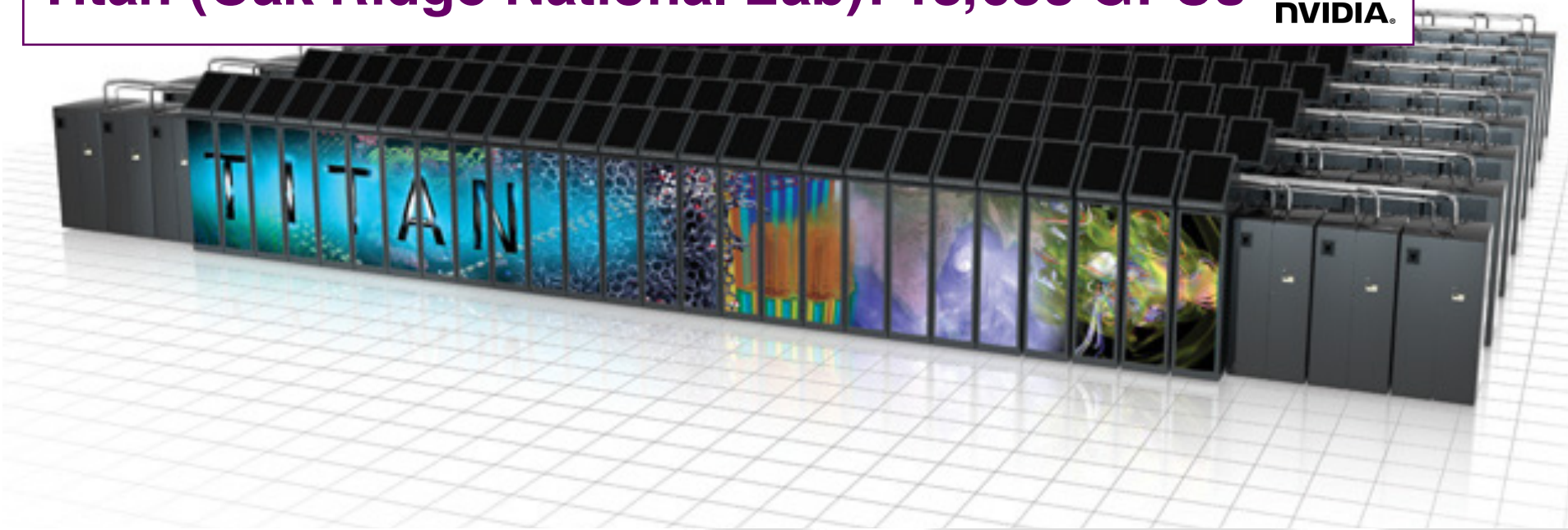
Error Rate

NVIDIA Kepler K40

intel Xeon-Phi



Titan (Oak Ridge National Lab): 18,688 GPUs



High probability of having a GPU corrupted

Titan Detected Uncorrectable Errors MTBF is ~44h*

*(field and experimental data from HPCA'15)

HPC bad stories

Virginia Tech's Advanced Computing facility built a supercomputer called Big Mac in 2003

- 1,100 Apple Power Mac G5
- Couldn't boot because of the failure rate
- Power Mac G5 did not have error-correcting code (ECC) memory
- Big Mac was broken apart and sold on-line

Jaguar – (2009 #1 Top500 list) ● 360 terabytes of main memory ● 350 ECC errors per minute

ASCI Q – (2002 #2 in Top500 list)

- Built with AlphaServers
- 7 Teraflops
- Couldn't run more than 1h without crash
- After putting metal side it could last 6h before crash
- Address bus on the microprocessors were unprotected (causing the crashes)

Outline

The origins of the issue:

- Radiation Effects Essentials
- Error Criticality in HPC

Understand the issue:

- Experimental Procedure
- K40 vs Xeon Phi

Toward the solution of the issue:

- ECC – ABFT – Duplication
- Selective Hardening

What's the Plan?

Outline

The origins of the issue:

- **Radiation Effects Essentials**
- **Error Criticality in HPC**

Understand the issue:

- Experimental Procedure
- K40 vs Xeon Phi

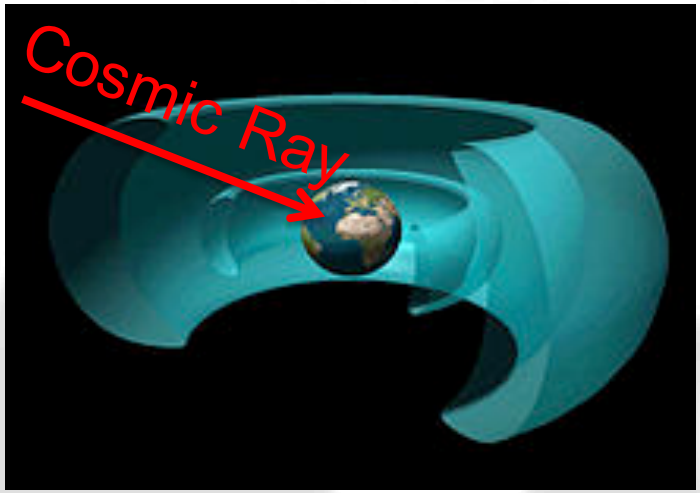
Toward the solution of the issue:

- ECC – ABFT – Duplication
- Selective Hardening

What's the Plan?

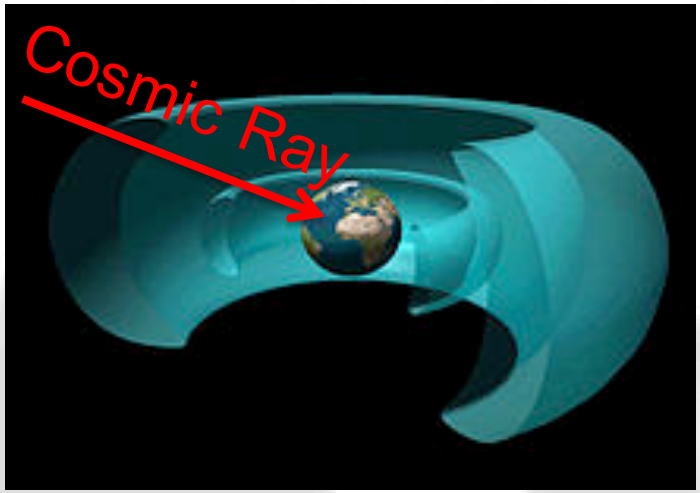
Terrestrial Radiation Environment

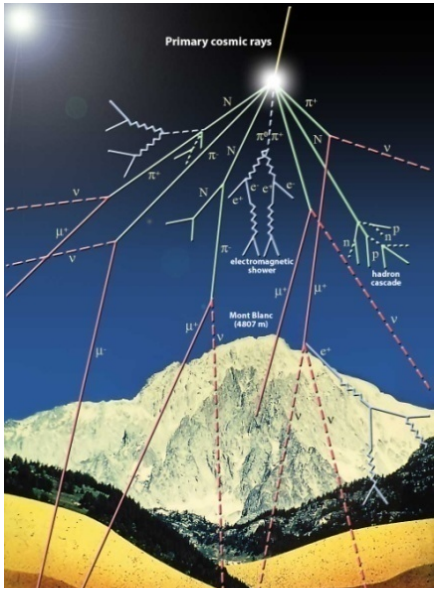
Cosmic rays could be so energetic to pass the Van Allen belts



Terrestrial Radiation Environment

Cosmic rays could be so energetic to pass the Van Allen belts





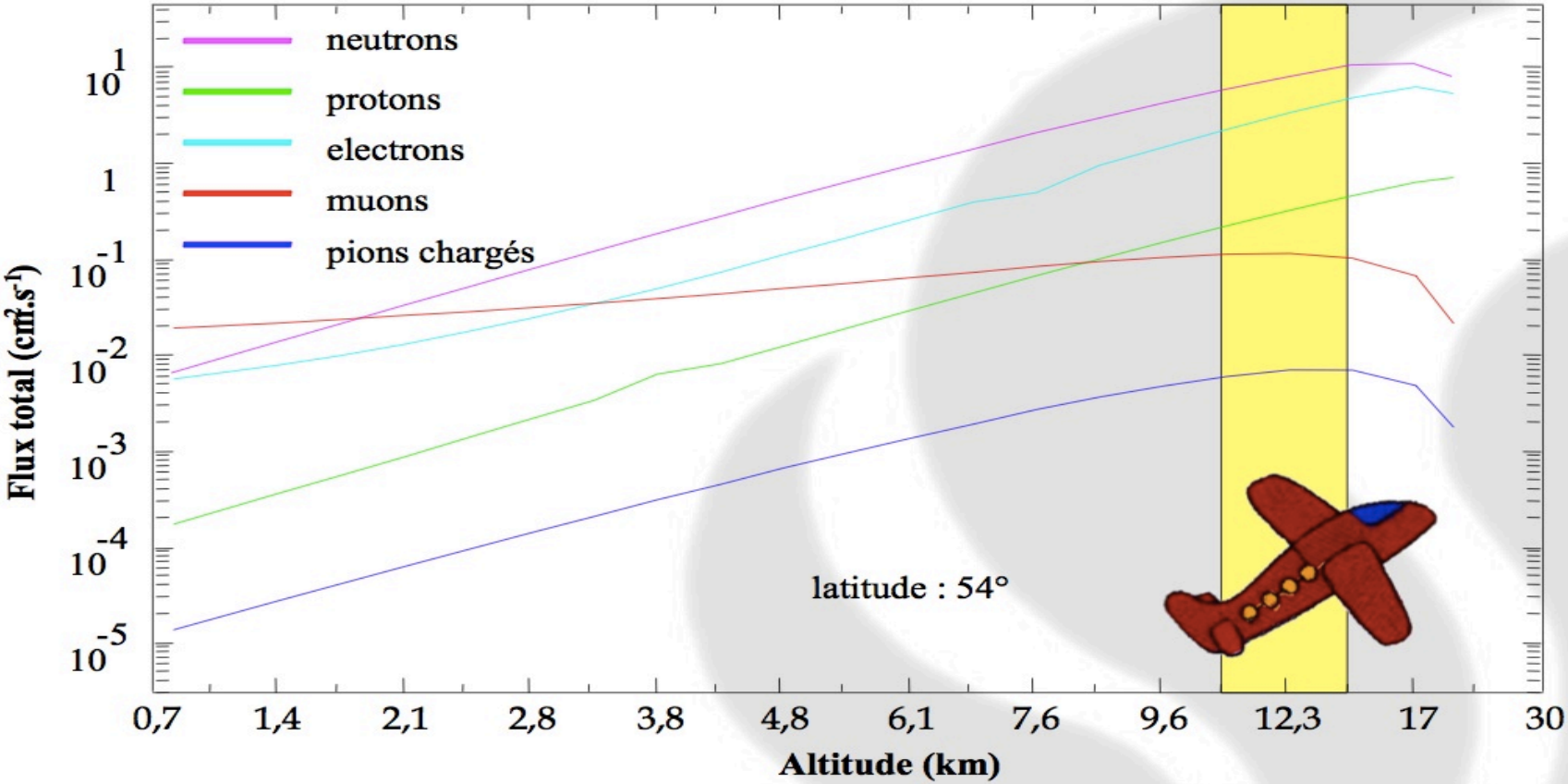
Galactic cosmic rays interact with atmosphere

shower of energetic particles:
Muons, Pions, Protons, Gamma rays, Neutrons

13 n/(cm^2·h) @sea level*

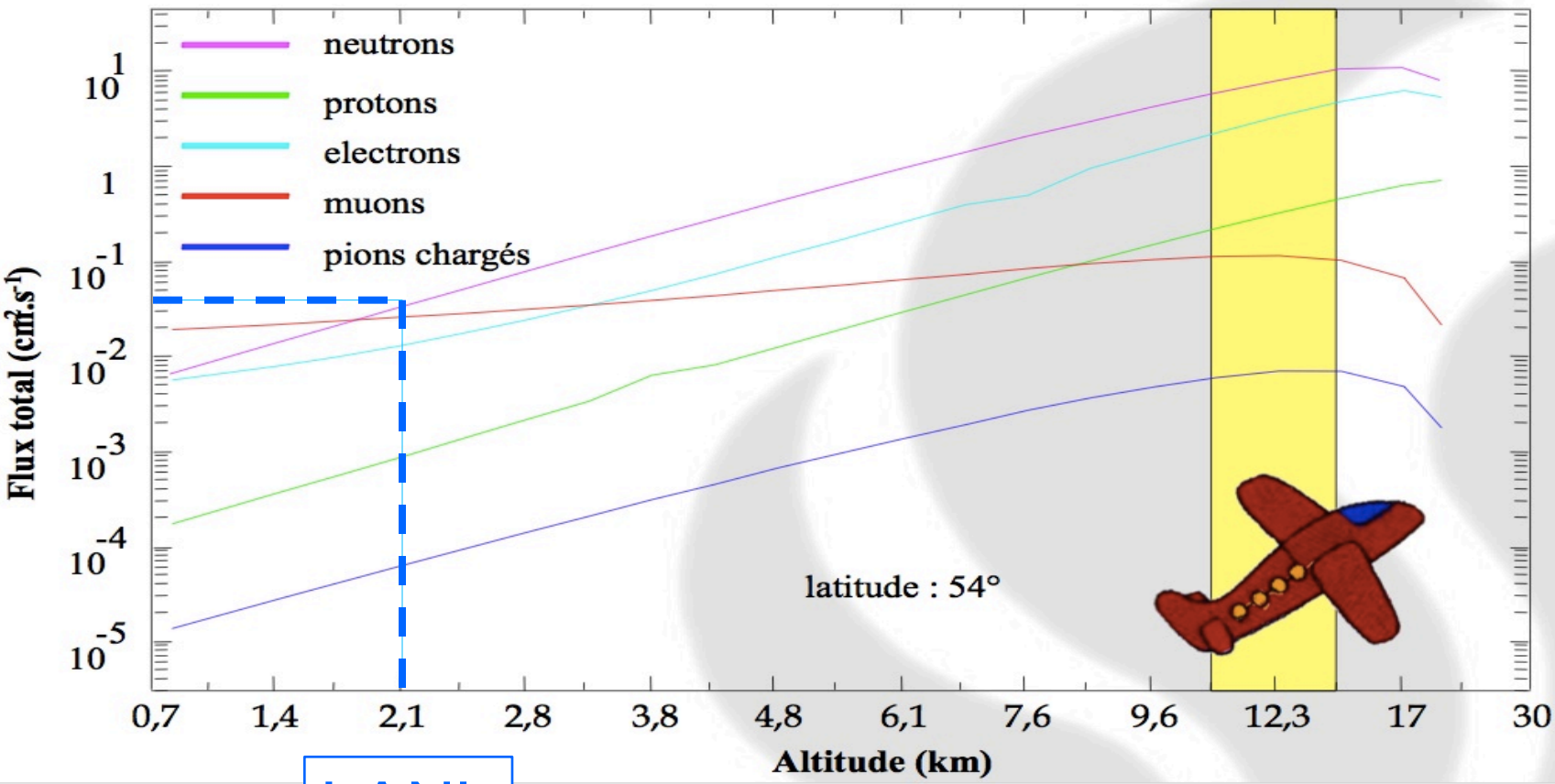
*JEDEC JESD89A Standard

Altitude and Radiation



Maximum ionization @ ~13KM above sea level

Altitude and Radiation



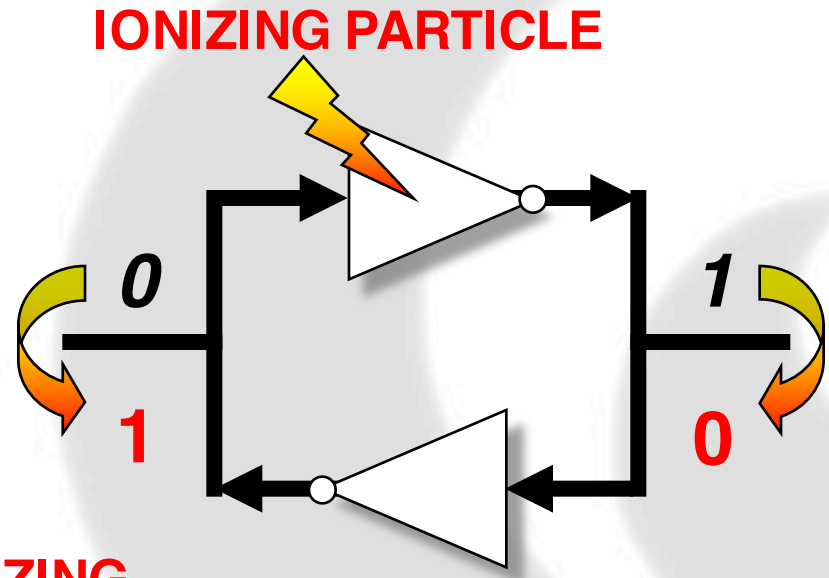
LANL

Maximum ionization @ ~13KM above sea level

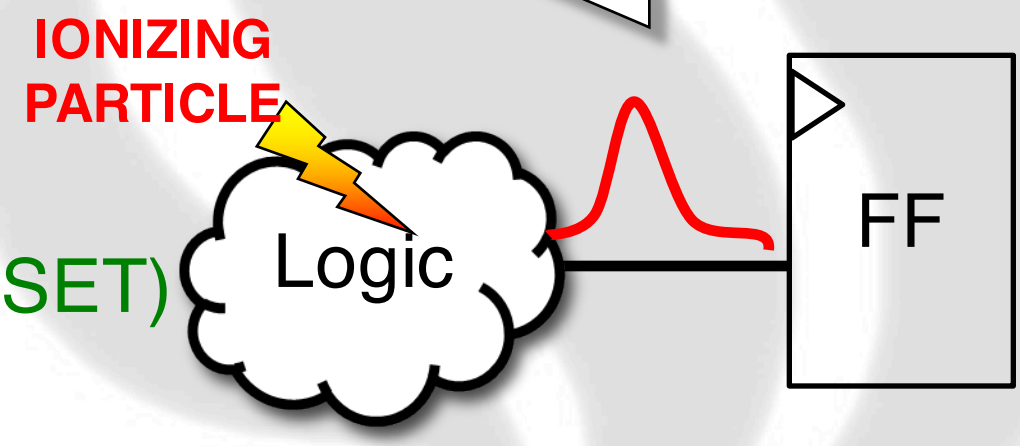
Radiation Effects - Soft Errors

Soft Errors: the device is not permanently damaged, but the particle may generate:

- One or more bit-flips
  - Single Event Upset (SEU)
  - Multiple Bit Upset (MBU)



- Transient voltage pulse
  - Single Event Transient (SET)



Silent Data Corruption vs Crash

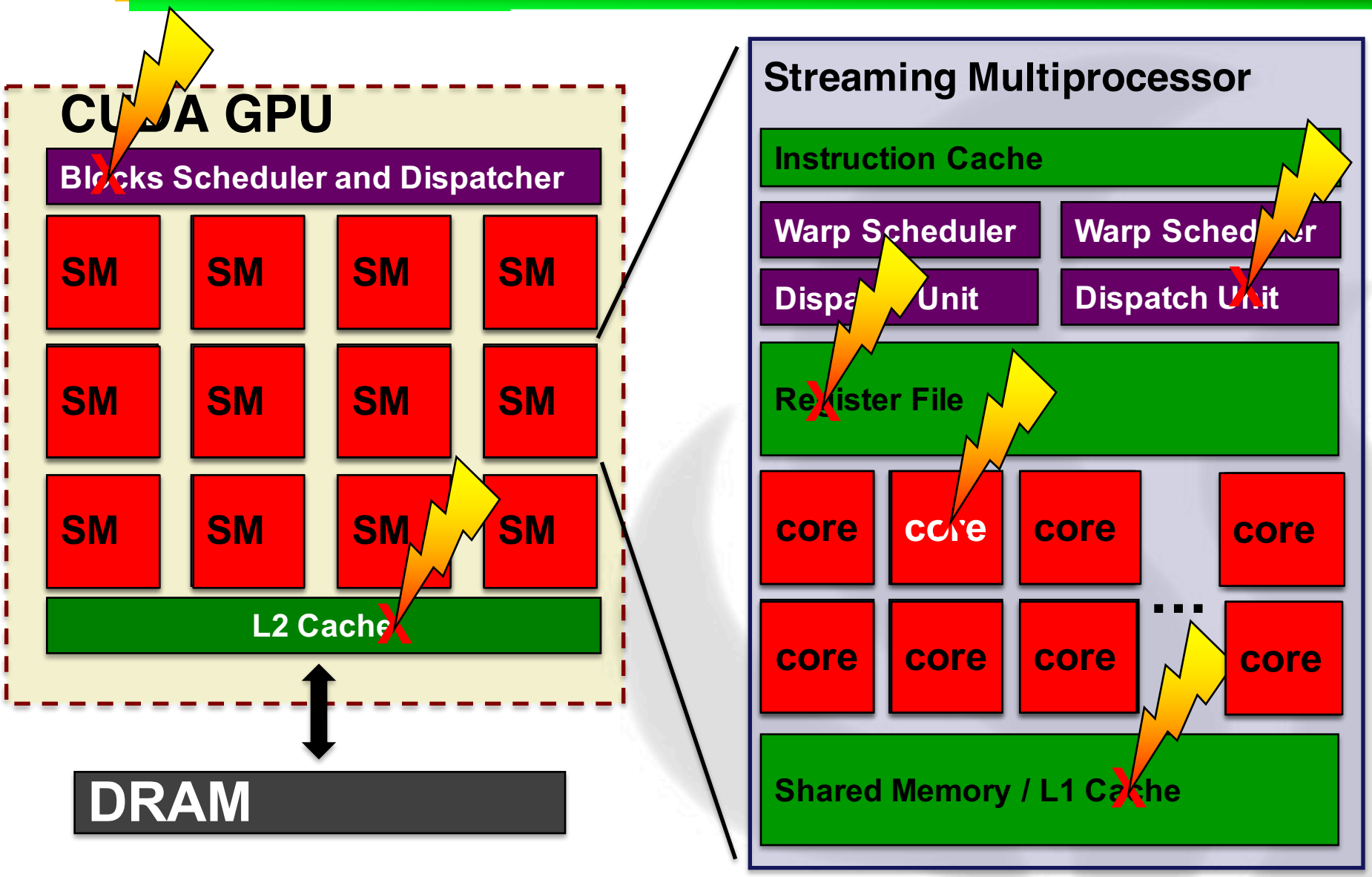
Soft Errors in:
-data cache
-register files
-logic gates (ALU)
-scheduler

Silent Data Corruption

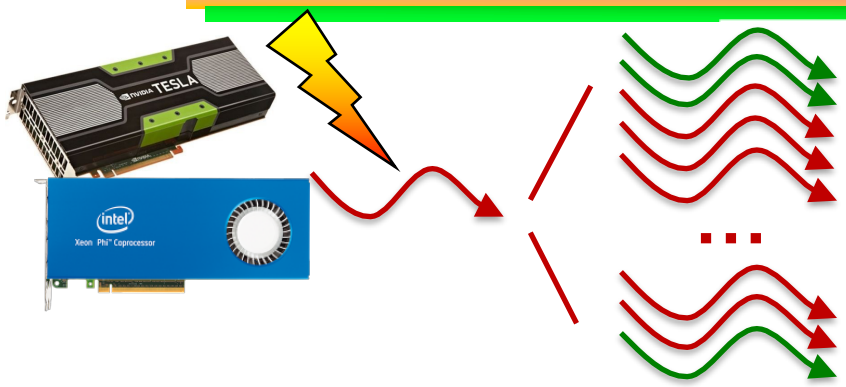
Soft Errors in:
-instruction cache
-scheduler / dispatcher
-PCI-e bus controller

DUE (Crash)

Radiation Effects on Parallel Accelerators

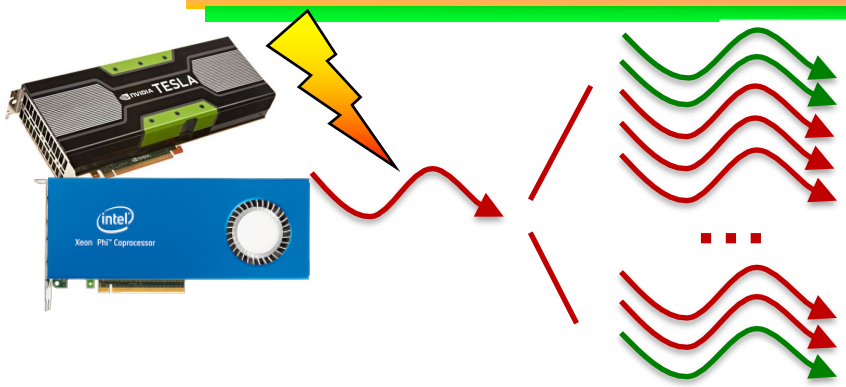


Output Correctness in HPC



A **single fault** can propagate to several parallel threads: **multiple corrupted elements**.

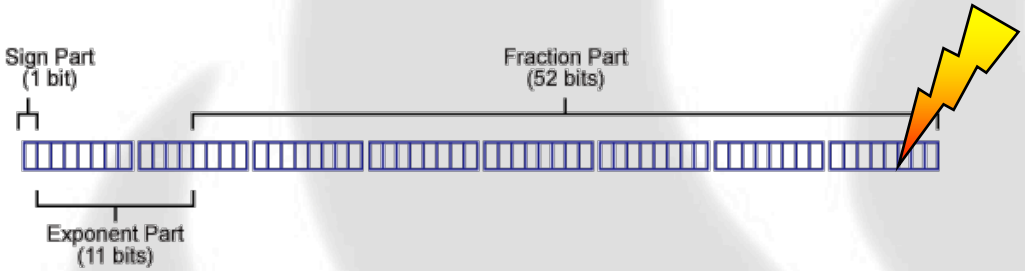
Output Correctness in HPC

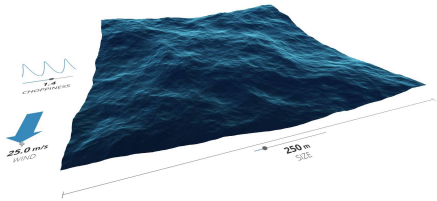


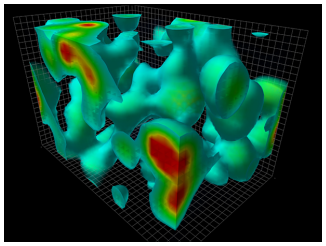
A **single fault** can propagate to several parallel threads: **multiple corrupted elements**.

Not all SDCs are critical for HPC applications

error can be in the float intrinsic variance



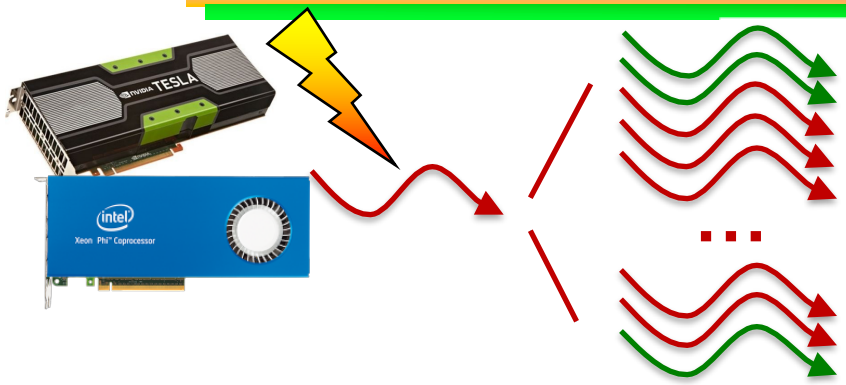




Values in a given range are accepted as correct in physical simulations

Imprecise computation is being applied to HPC

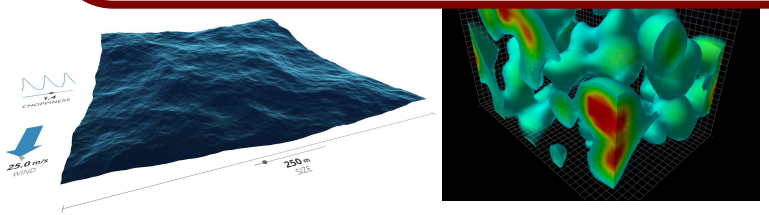
Output Correctness in HPC



A **single fault** can propagate to several parallel threads: **multiple corrupted elements**.

Not all SDCs are critical for HPC applications

Goal: quantify and qualify SDC in **NVIDIA** and **Intel** architectures.



values in a given range are accepted as correct in physical simulations

Imprecise computation is being applied to HPC

Outline

The origins of the issue:

- Radiation Effects Essentials
- Error Criticality in HPC

Understand the issue:

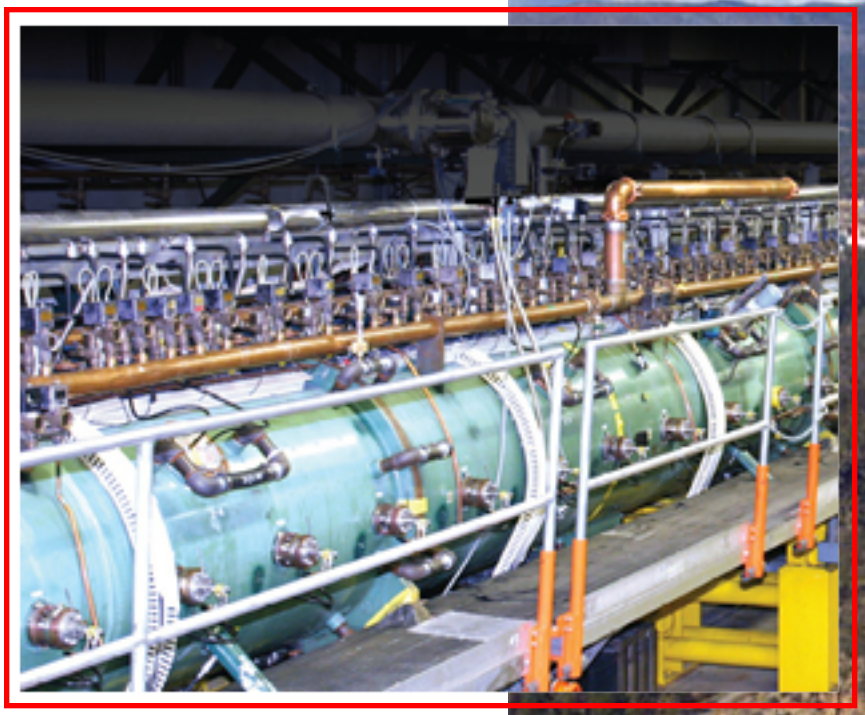
- **Experimental Procedure**
- **K40 vs Xeon Phi**

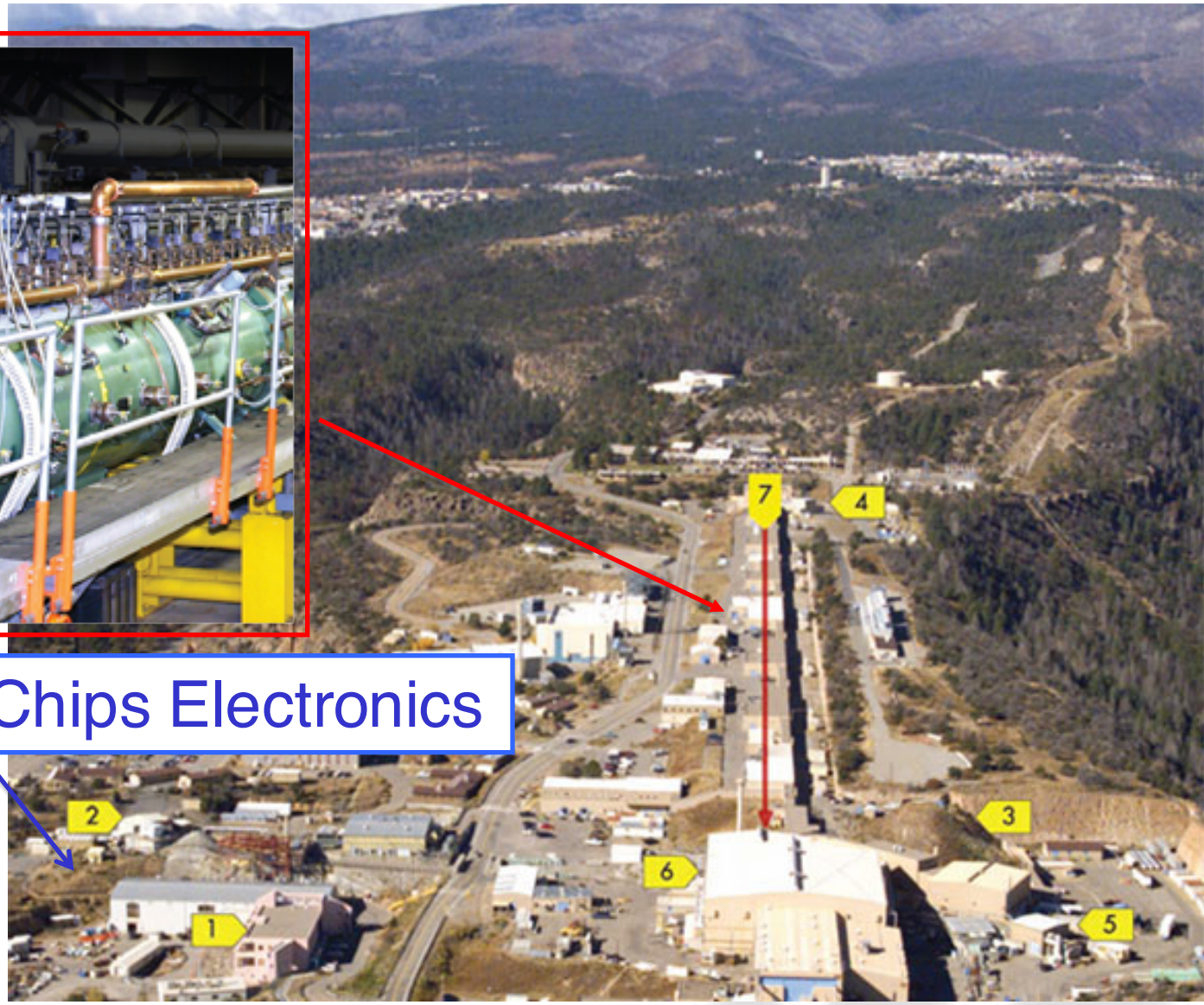
Toward the solution of the issue:

- ECC – ABFT – Duplication
- Selective Hardening

What's the Plan?

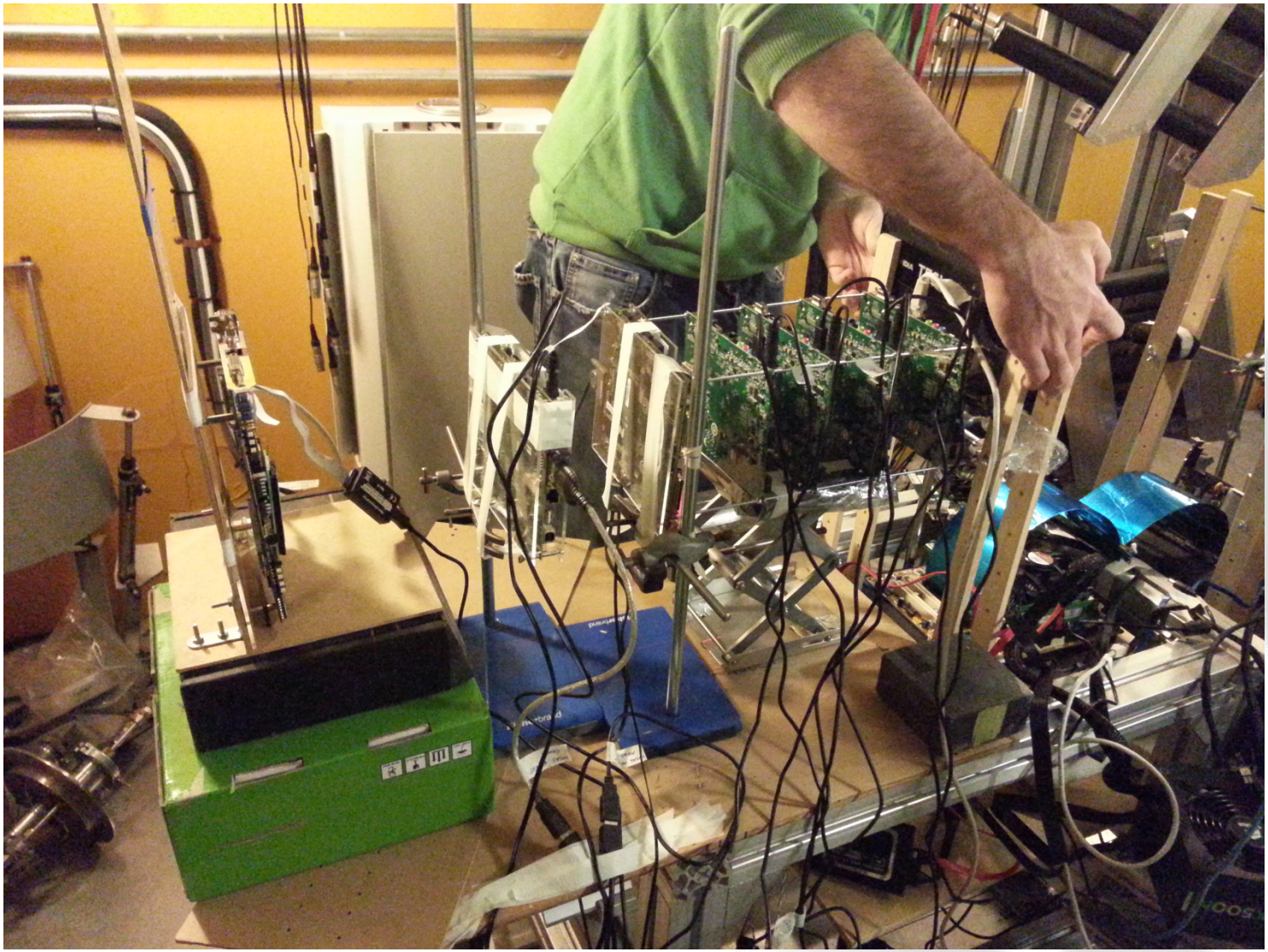
Radiation Test Facilities





Irradiation of Chips Electronics

Experimental Setup



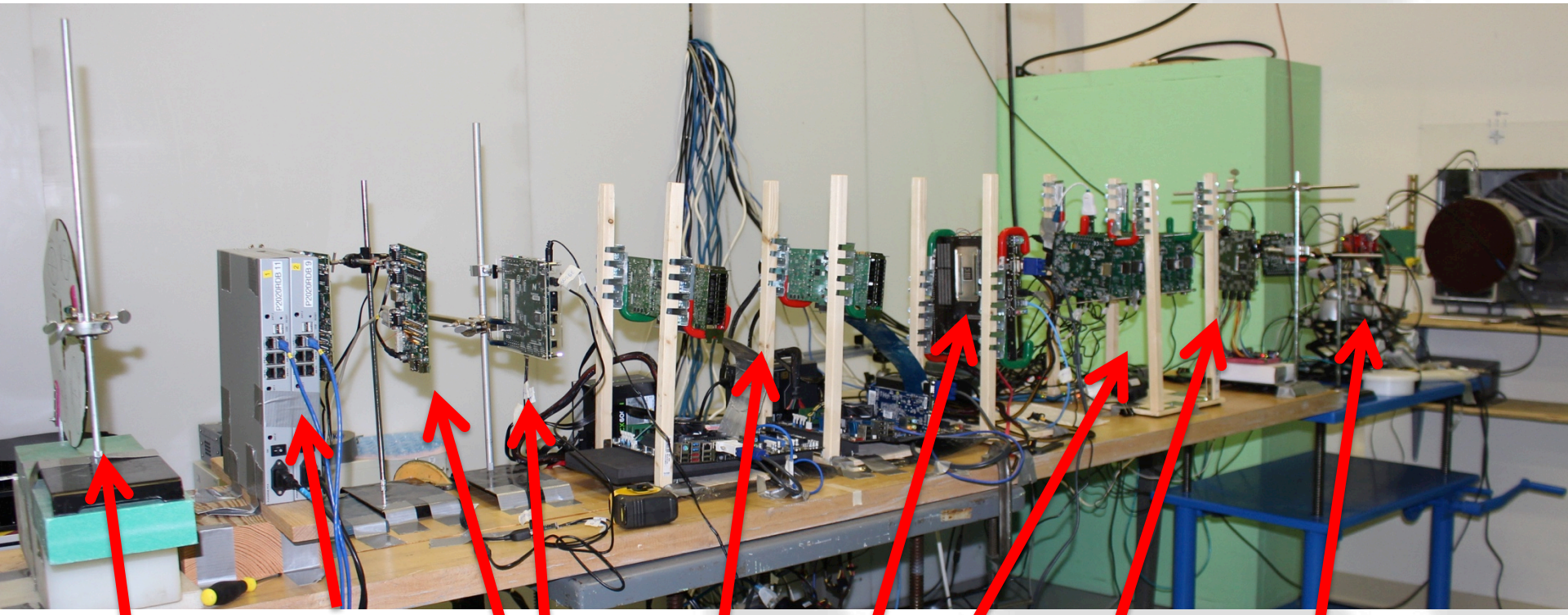
Radiation Test are NOT for dummies

W

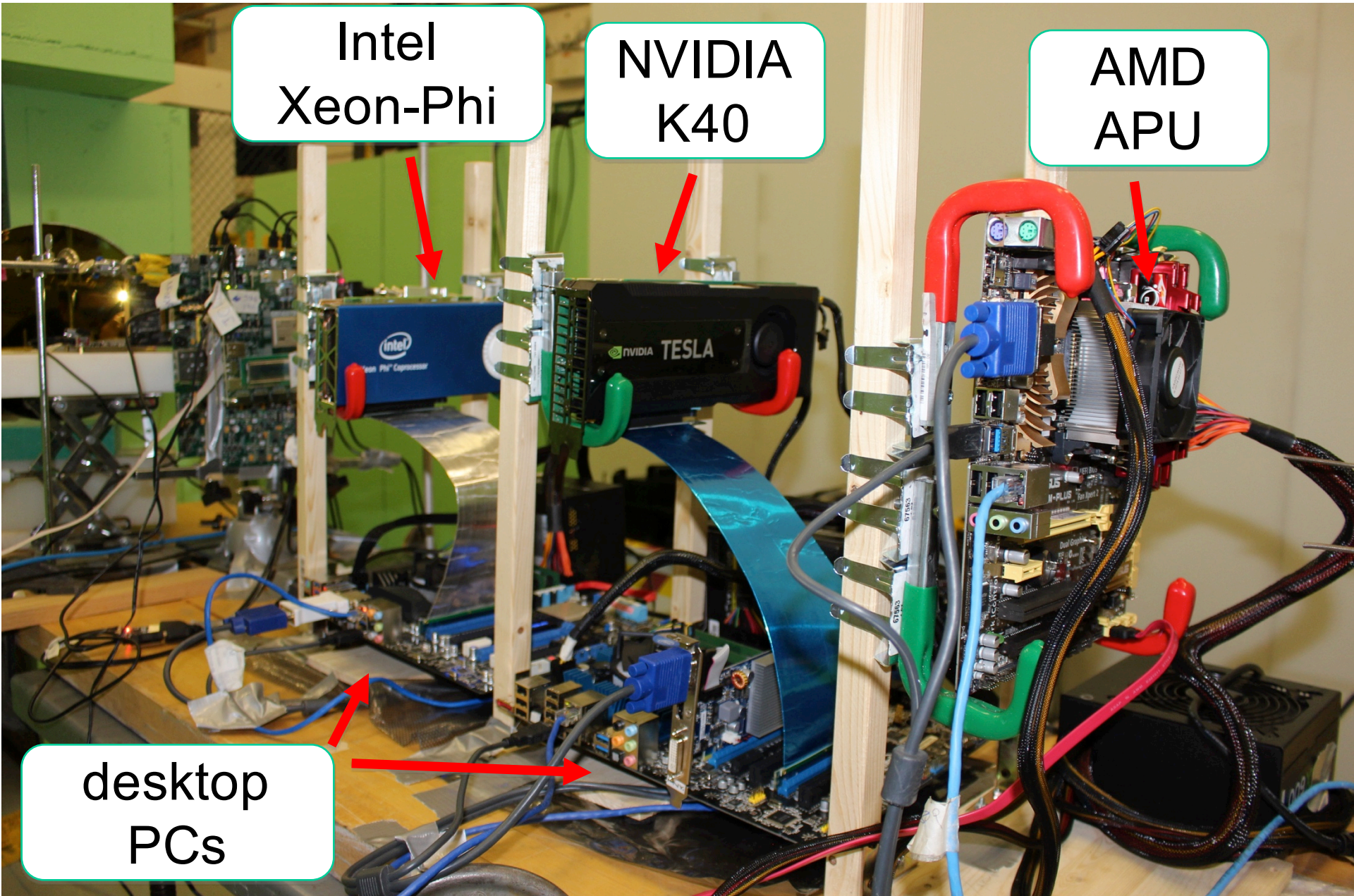
-
-
-
-
-
-
-
-
-

list?

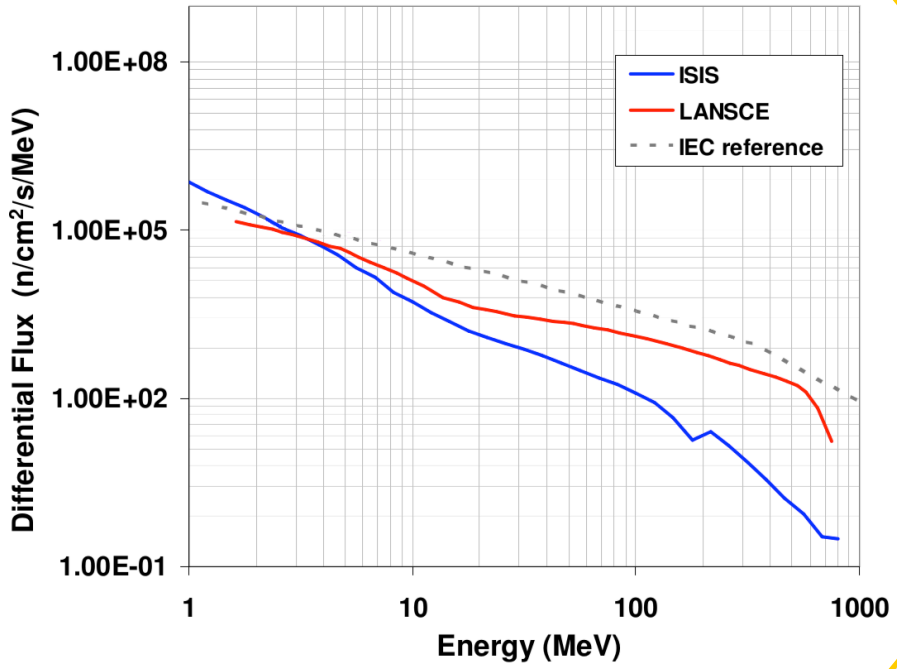
GPU Radiation Test Setup



GPU Radiation Test Setup



Neutrons Spectrum

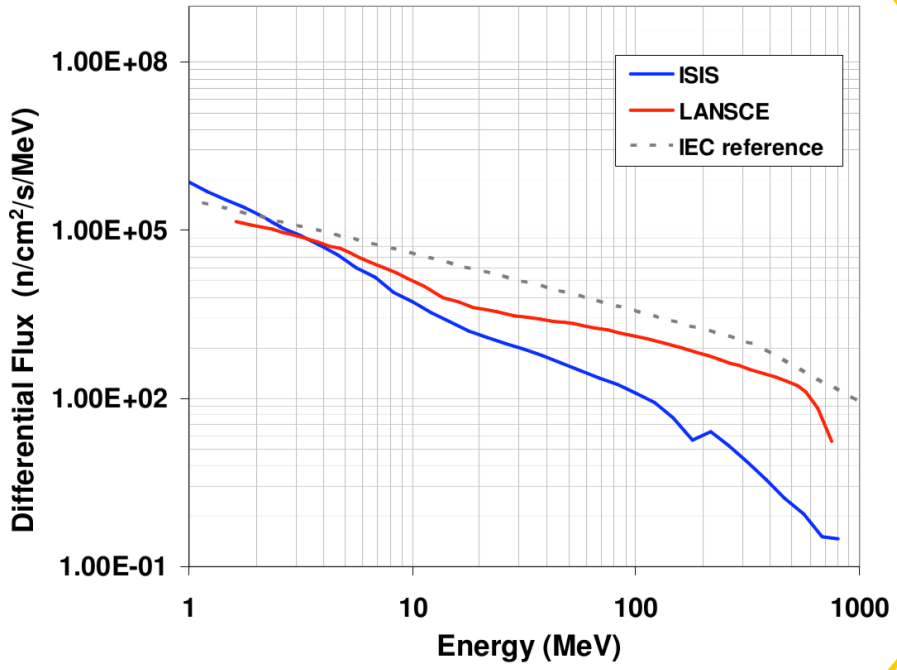


@LANSCE 1.8×10^6 n/(cm^2 h)

@NYC 13 n/(cm^2 h)

We test each architecture for 800h, simulating 9.2×10^8 h of natural radiation (~ **91,000 years**)

Neutrons Spectrum



@LANSCE 1.8×10^6 n/(cm^2 h)

@NYC 13 n/(cm^2 h)

We test each architecture for 800h, simulating 9.2×10^8 h of natural radiation (~ **91,000 years**)

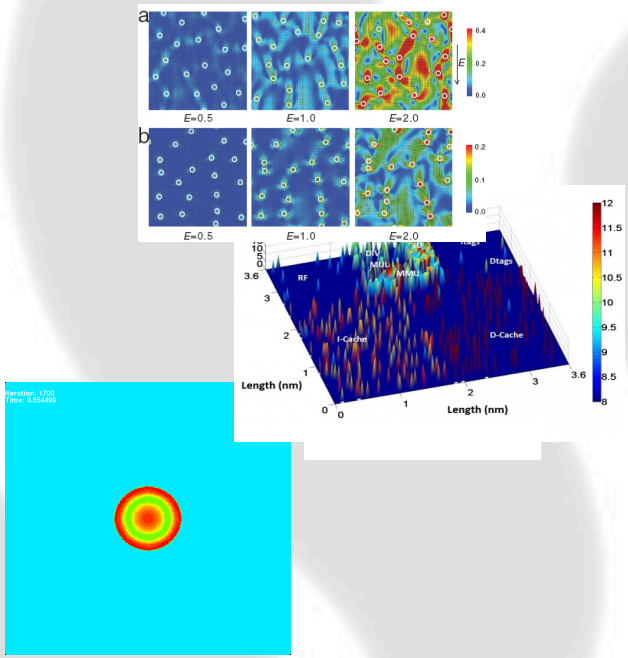
All the collected SDCs are publicly available:
https://github.com/UFRGS-CAROL/HPCA2017-log-data

Selected Algorithms

We select a set of benchmarks that:

- stimulate different resources
- are representative of HPC applications
- minimize error masking (high AVF)

- **DGEMM:** matrix multiplication
- **lavaMD:** particles interactions
- **Hotspot:** heat simulation
- **Needleman–Wunsch:** Biology
- **CLAMR:** DOE's workload
- **Quick- Merge- Radix-Sort**
- **Matrix Transpose:** Memory
- **Gaussian**

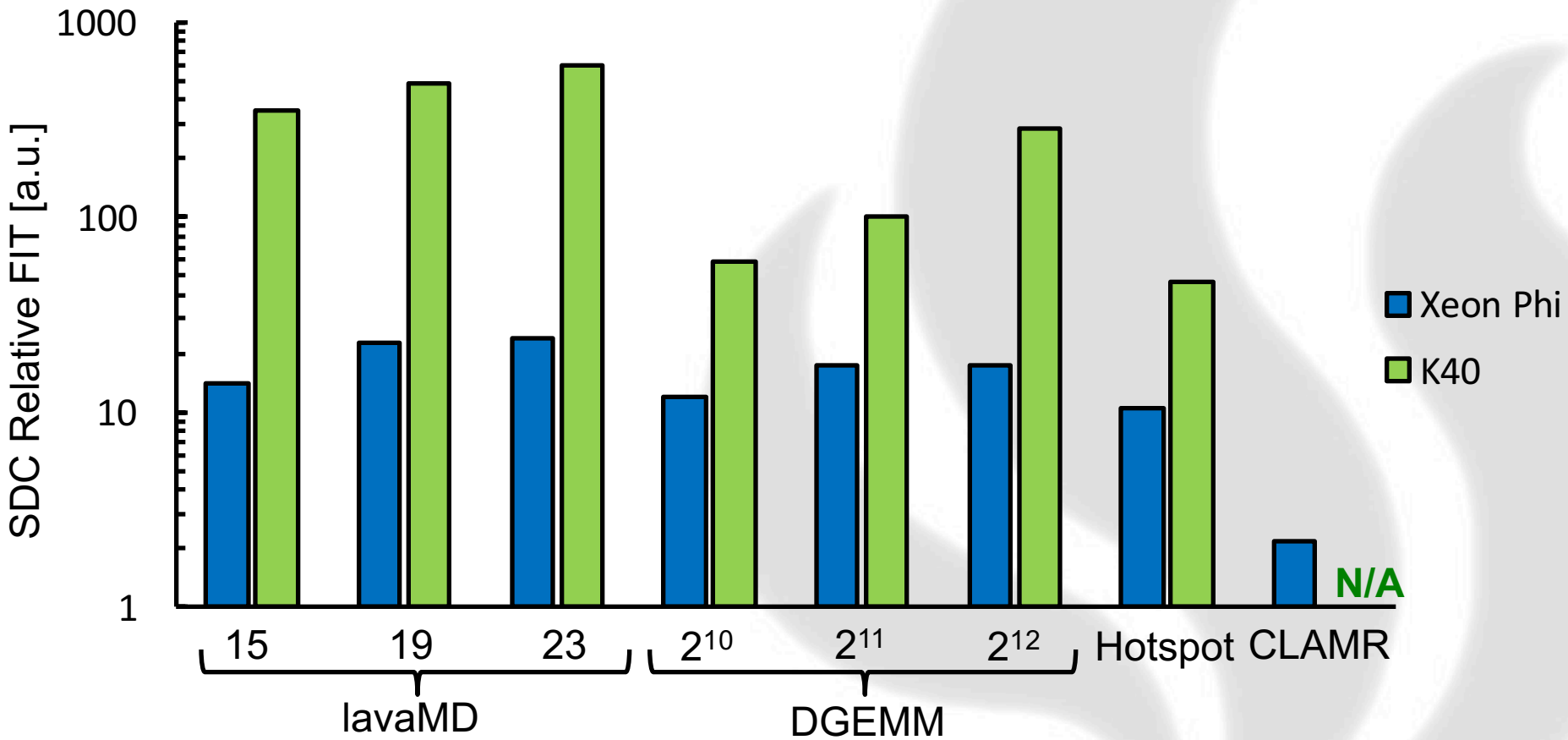


Xeon Phi vs K40 SDC rate

Xeon Phi error rate seems lower than Kepler, but:

-Xeon Phi is built in 3D Trigate, Kepler in planar CMOS

-Xeon Phi and K40 have different throughput



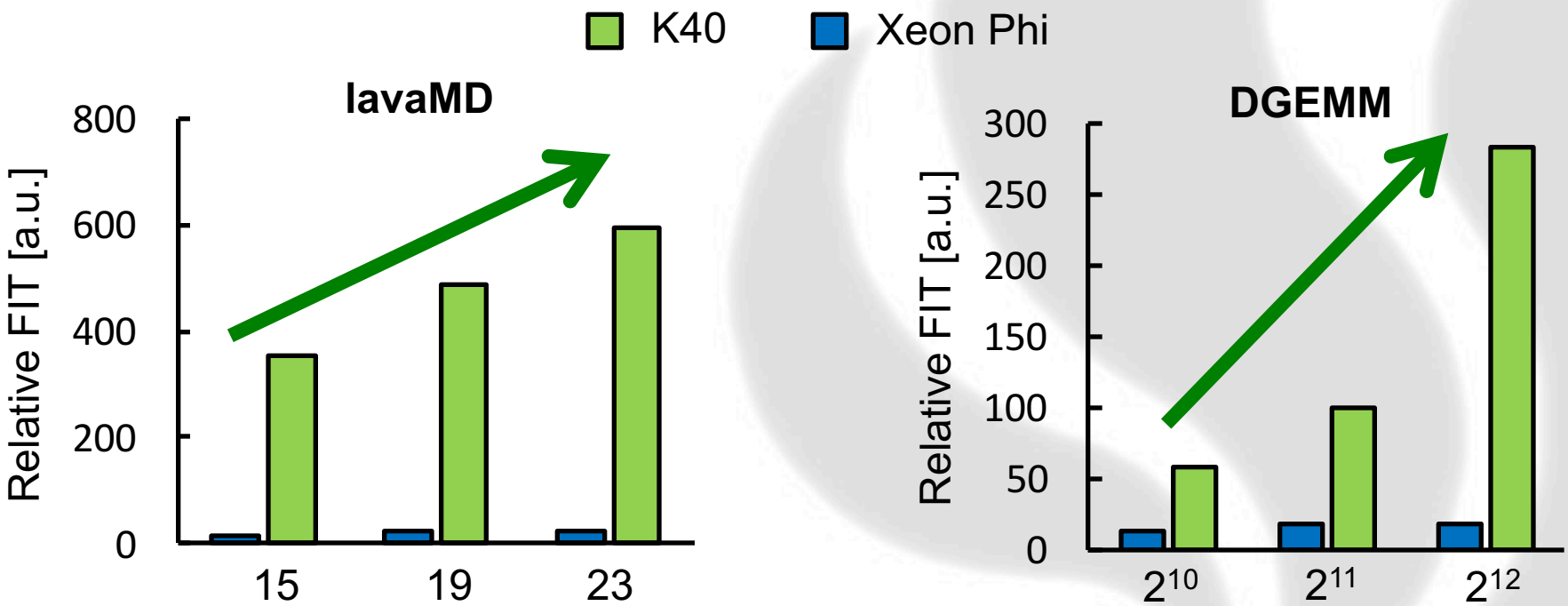
Parallelism Management Reliability

~95% processor resources used with smallest input

Increasing the input size we increase the #threads:

-Xeon-Phi error rate remains constant (<20% variation)

-K40 SDC error rate increases with input size



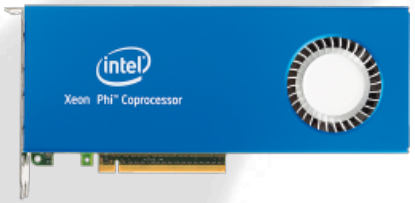
Parallelism Management Reliability

K40

FIT increases with input size: **HW scheduler is prone to be corrupted!**

data of 2048 active threads is maintained in the register file

Xeon-Phi



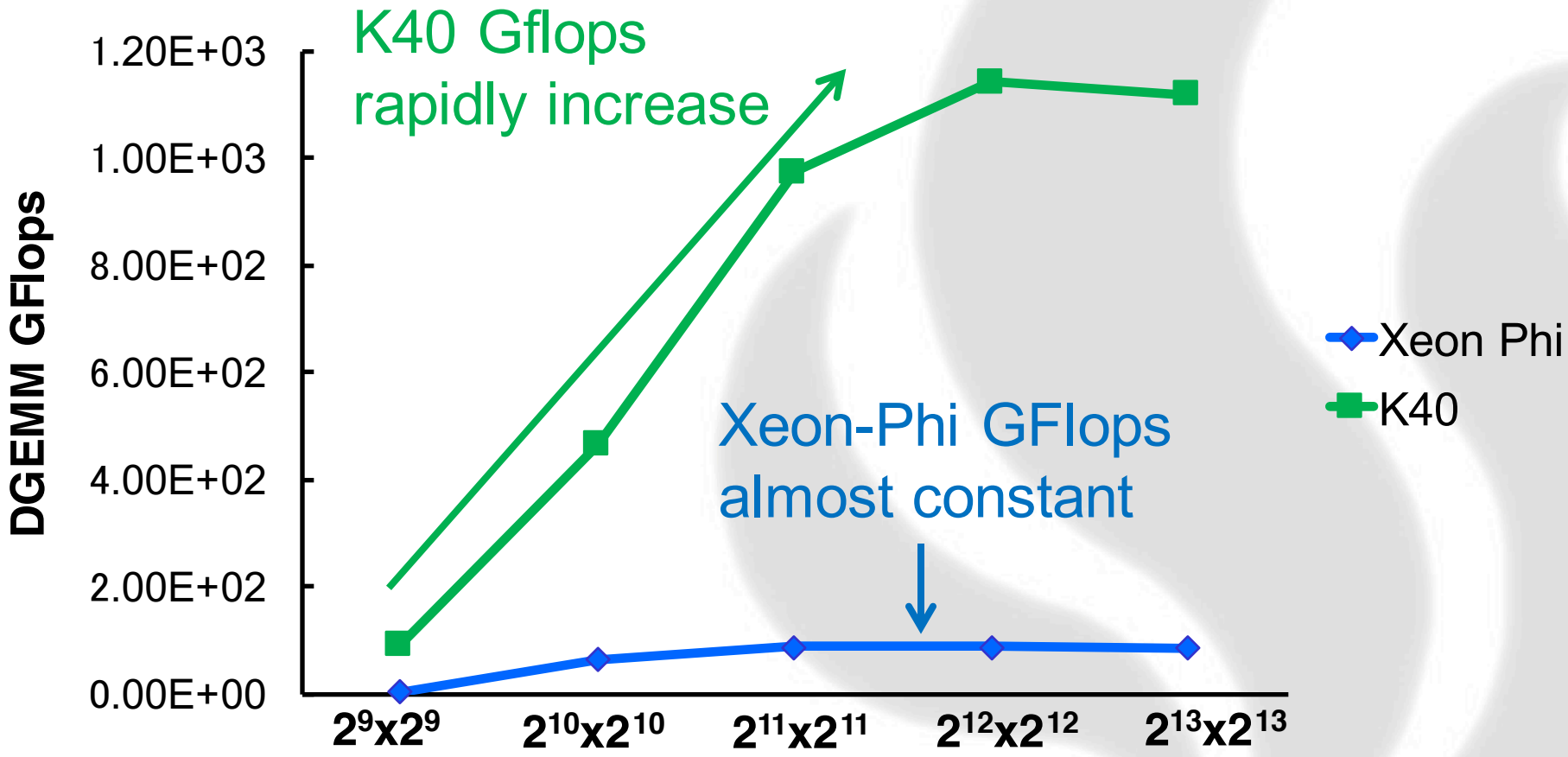
constant FIT rate: **embedded OS is OK!**

only 4 threads/core are maintained. Other threads data in the main memory (not exposed)

Parallelism Management Reliability

K40 throughput increases with input size.

Reliability vs Performances trade-off should be considered



Mean Workload Between Failures

↑ Parallel threads

↑ Error rate ☹

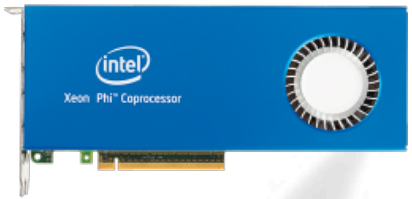
↑ Throughput ☺

Mean Workload Between Failures

↑ Parallel threads

↑ Error rate

↑ Throughput



= Error rate

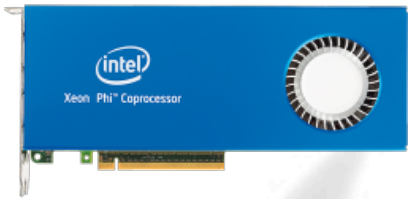
= Throughput

Mean Workload Between Failures

↑ Parallel threads

↑ Error rate ☹

↑ Throughput ☺



= Error rate ☺

= Throughput 😐

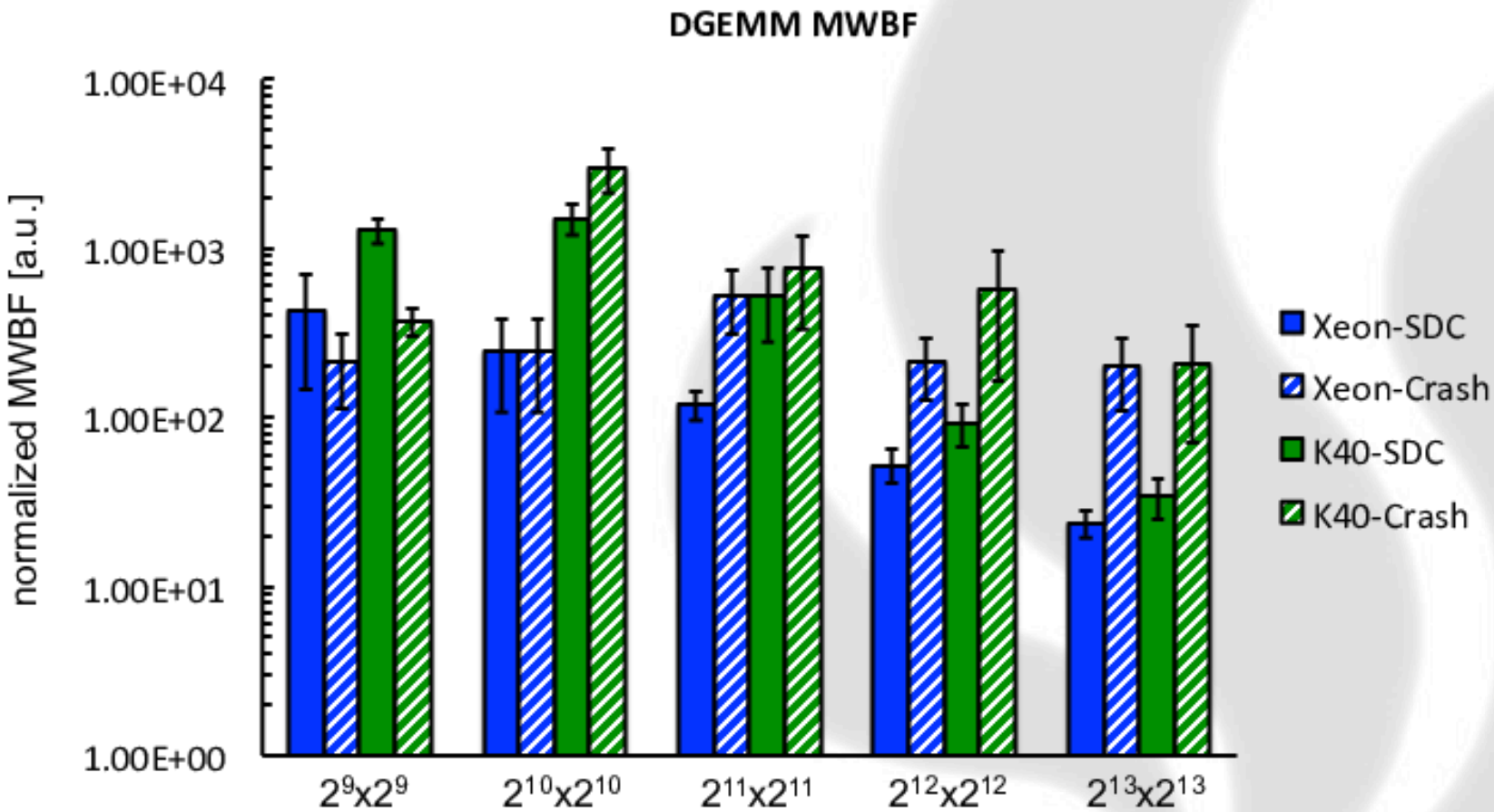
Which architecture produces a higher amount of data before experiencing a failure? Is there a sweet spot?

Mean Workload Between Failures

DGEMM MWBF

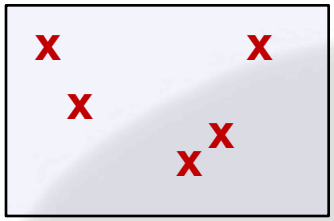
Xeon-Phi MWBF decreases significantly with input size.

Even if more prone to be corrupted, Kepler produces more correct data (if parallelism is exploited)

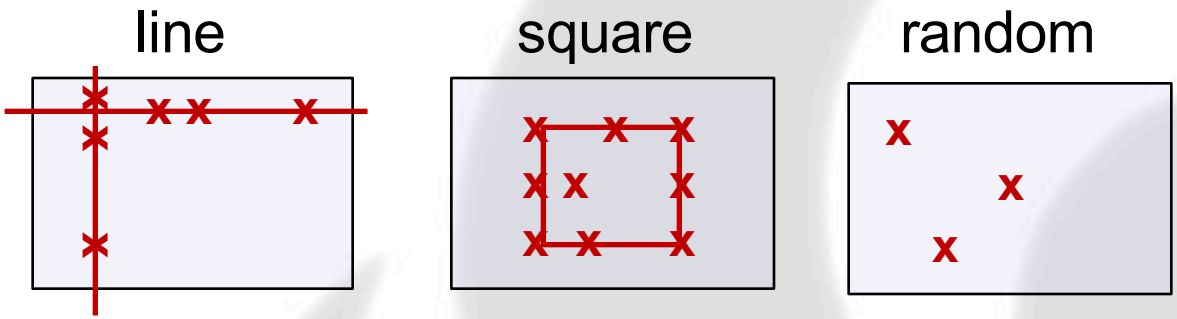


Quantify and Qualify SDCs

Number of incorrect elements

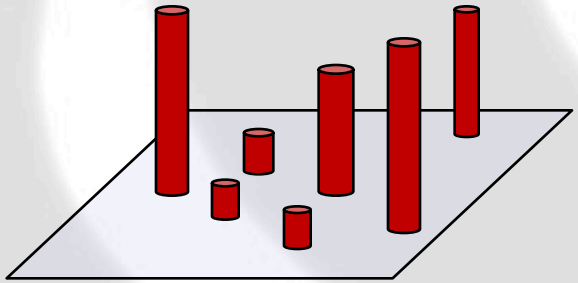


Spatial Locality



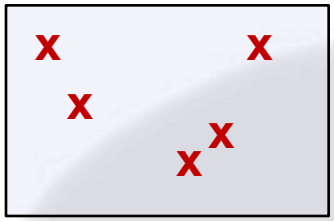
Relative Error

how different the error is from the expected value

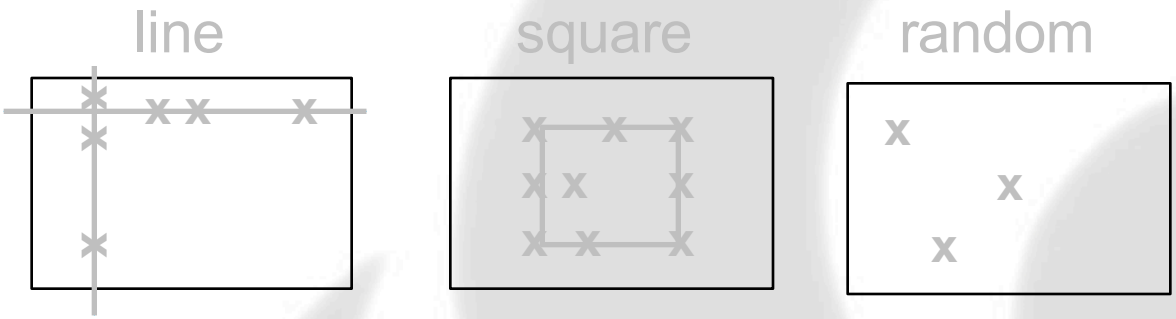


Quantify and Qualify SDCs

Number of incorrect elements

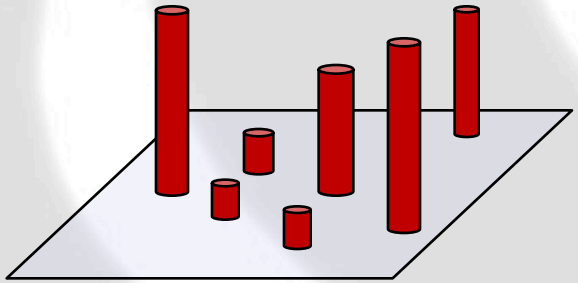


Spatial Locality

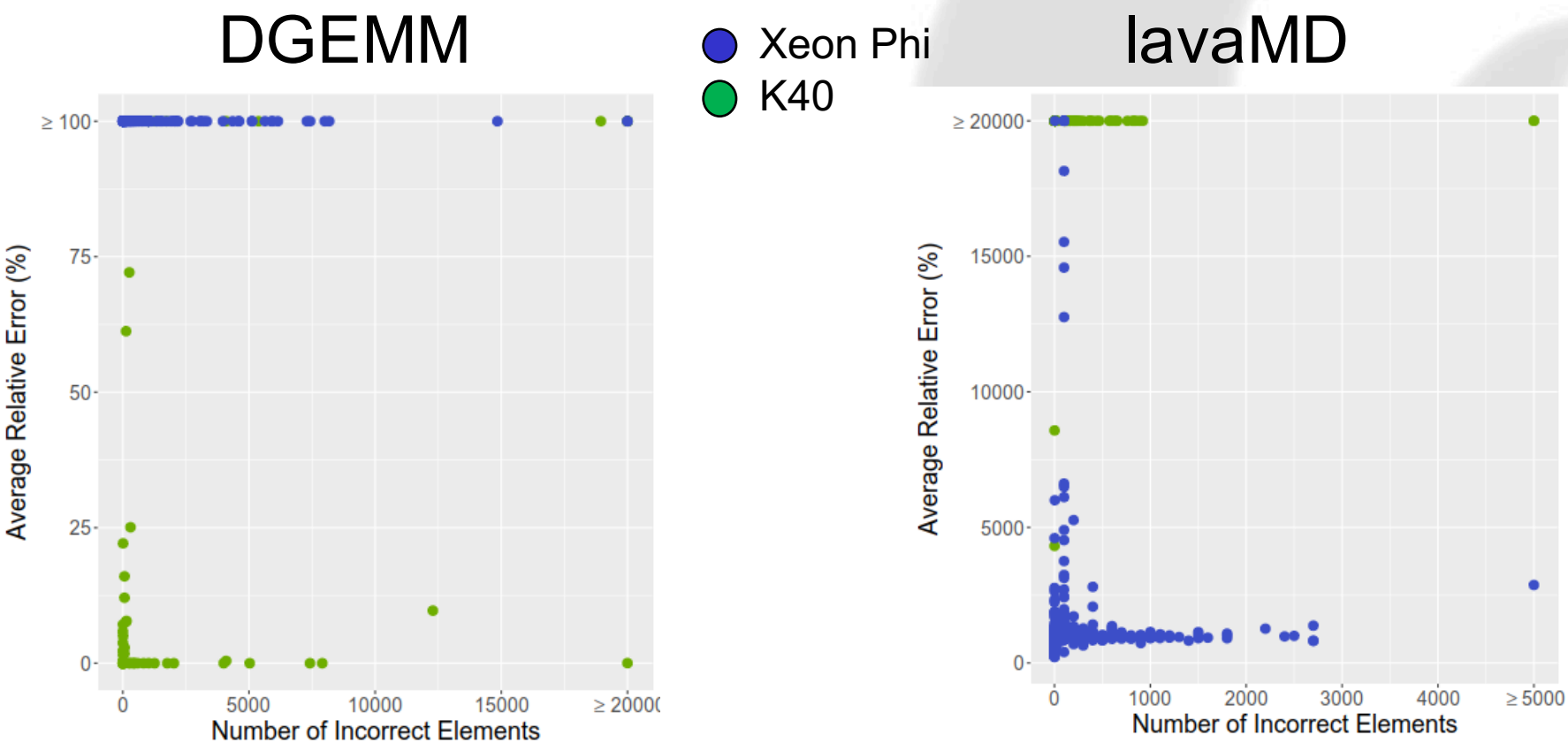


Relative Error

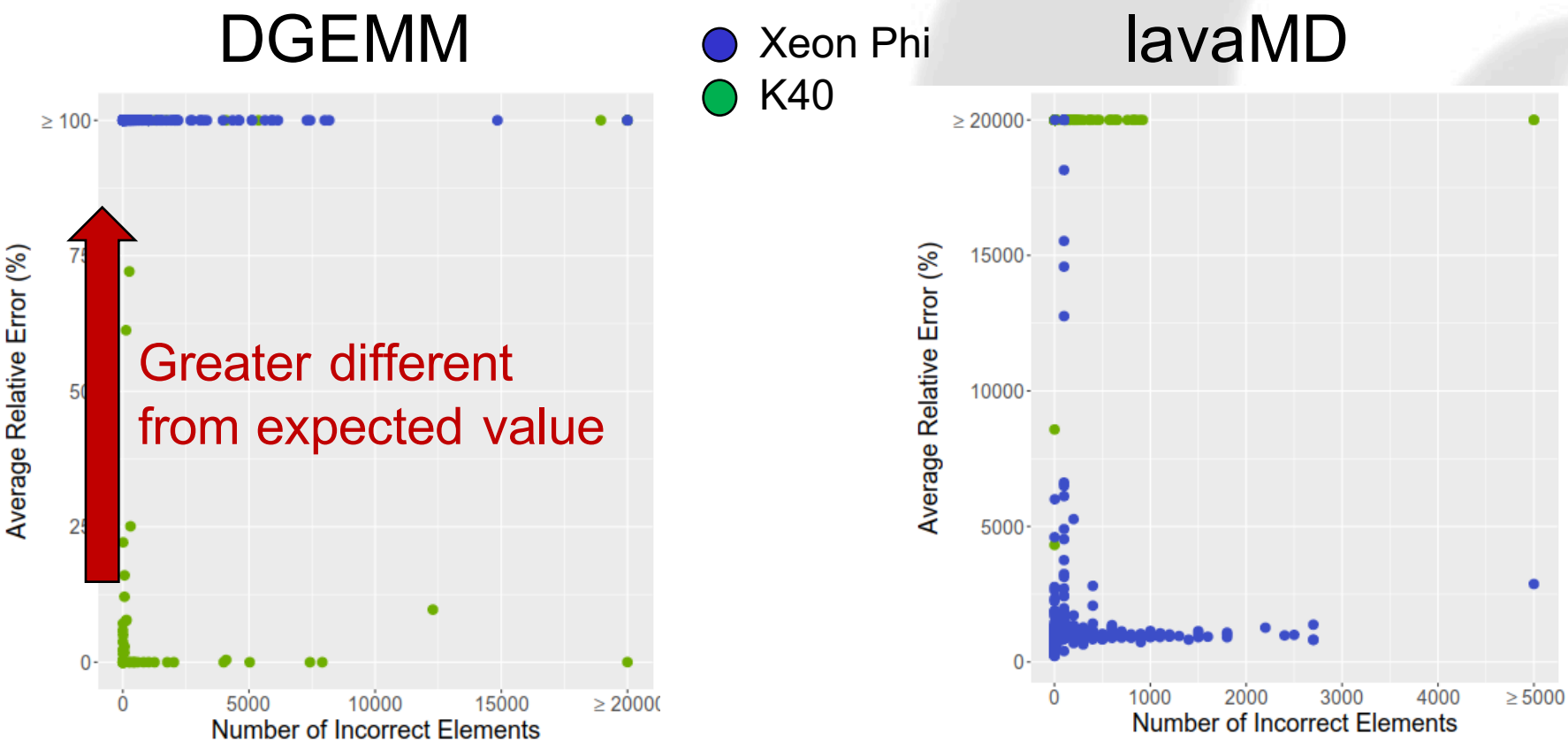
how different the error is from the expected value



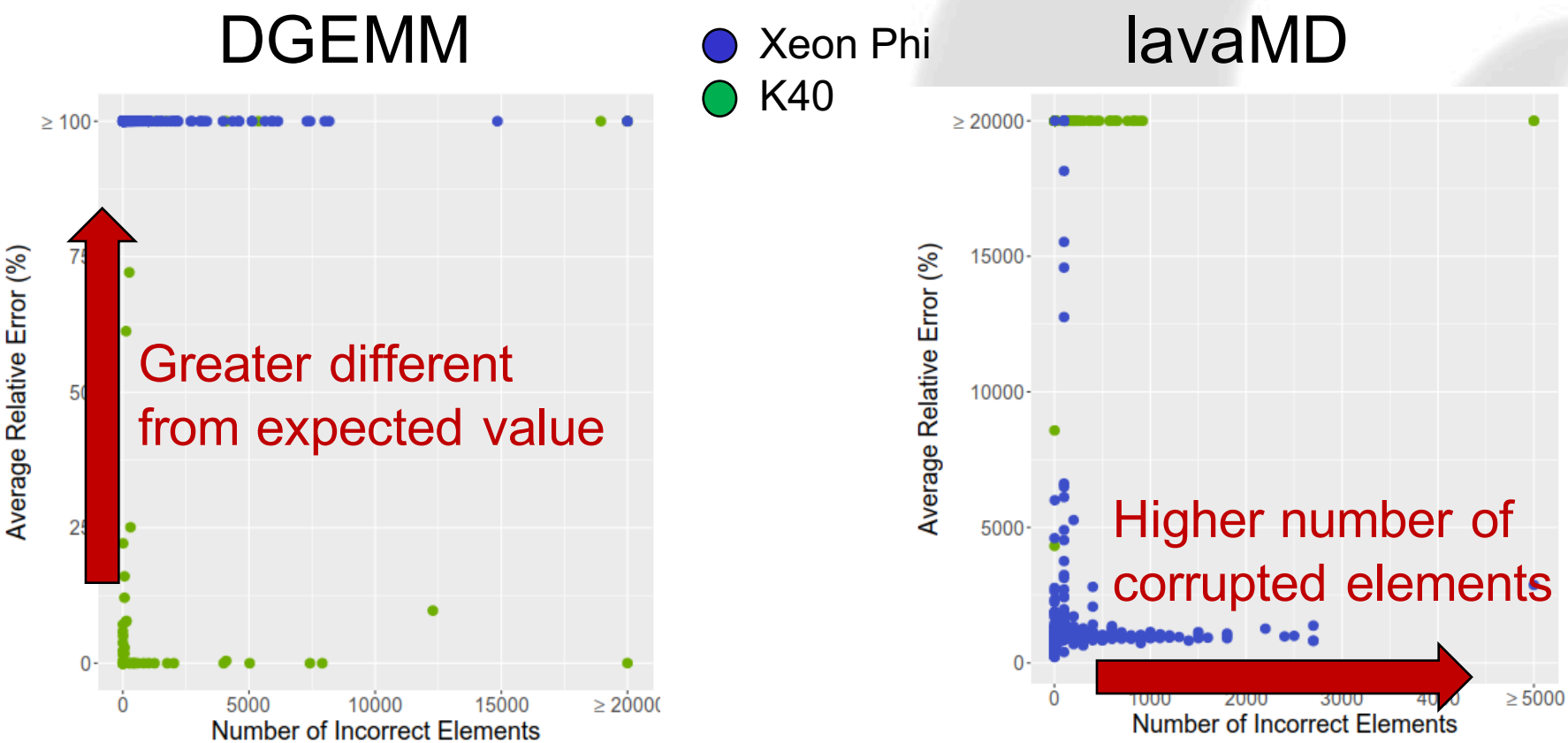
Number of Incorrect Elements vs Relative Error



Number of Incorrect Elements vs Relative Error

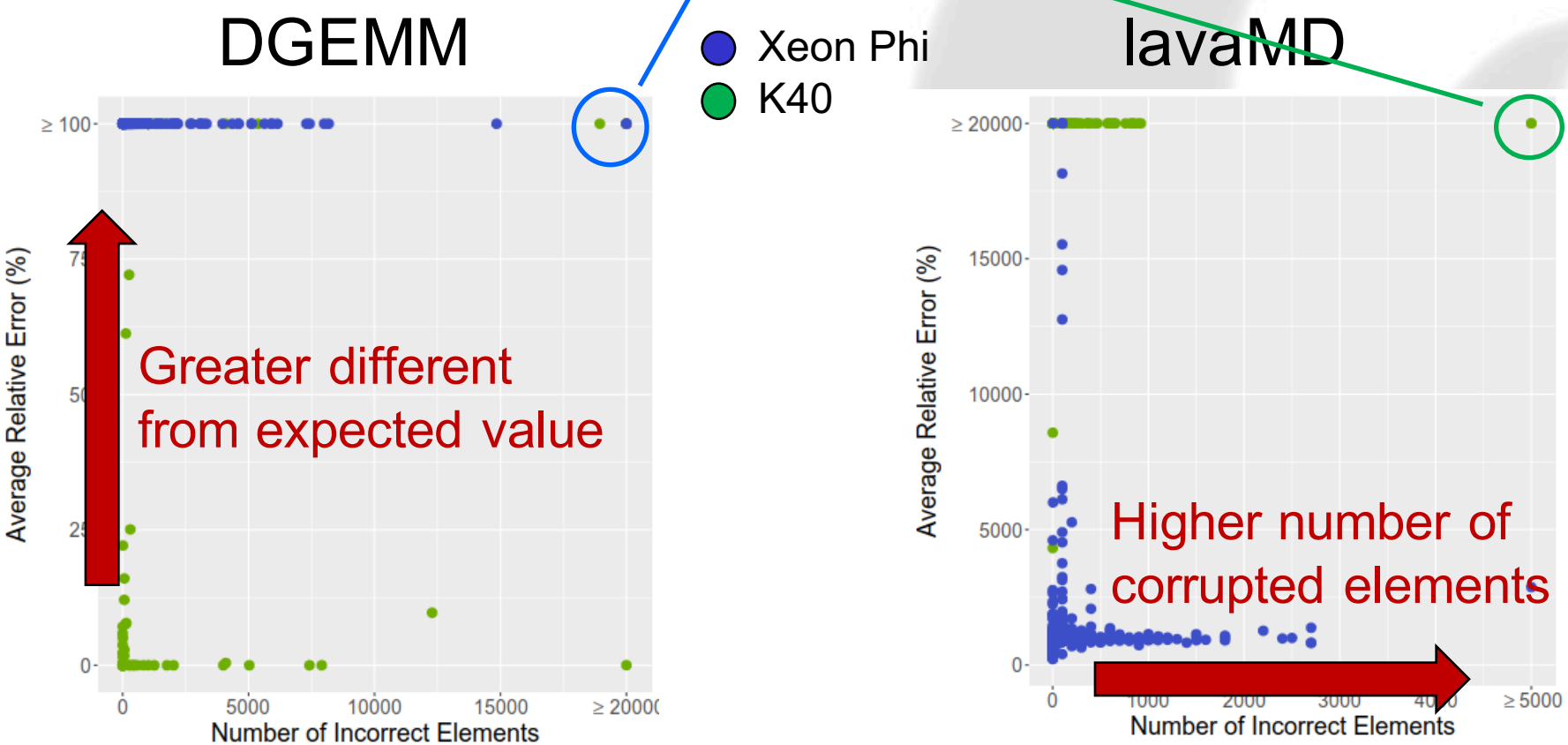


Number of Incorrect Elements vs Relative Error



Number of Incorrect Elements vs Relative Error

BAD: high number of corrupted elements, which are very different from the expected output

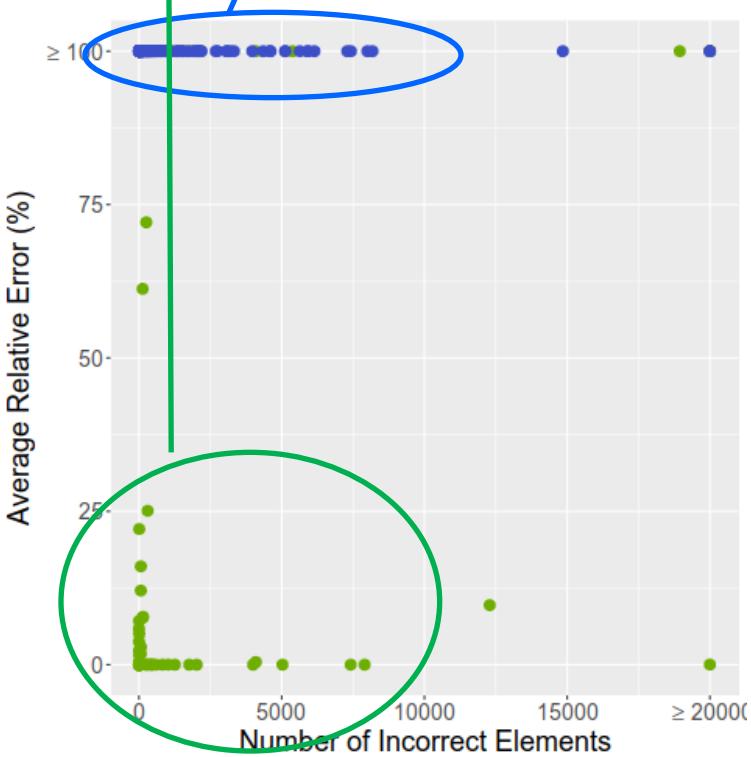


Number of Incorrect Elements vs Relative Error

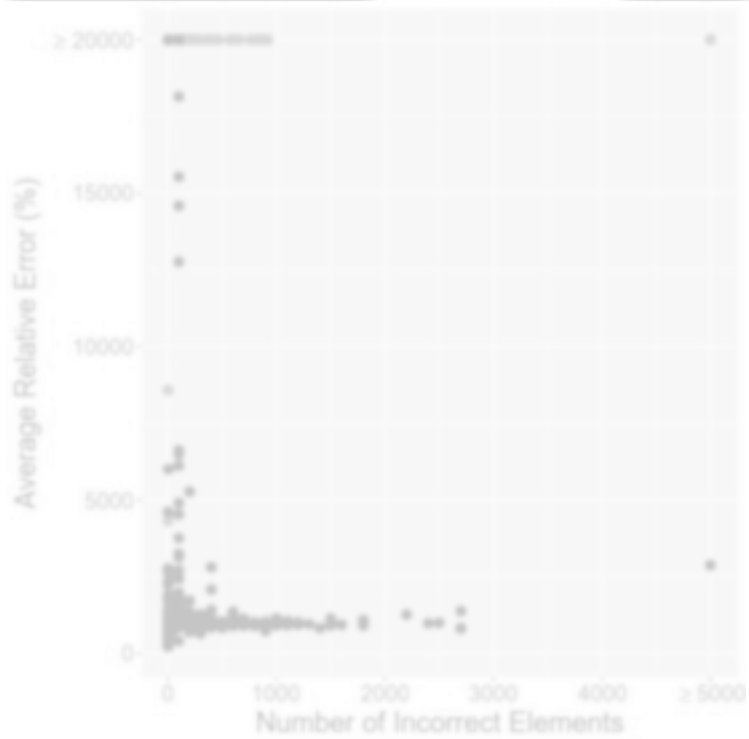
K40 few corrupted elements, value similar to expected one

Xeon Phi: a lot of corrupted elements, which are very different from expected value

DGEMM

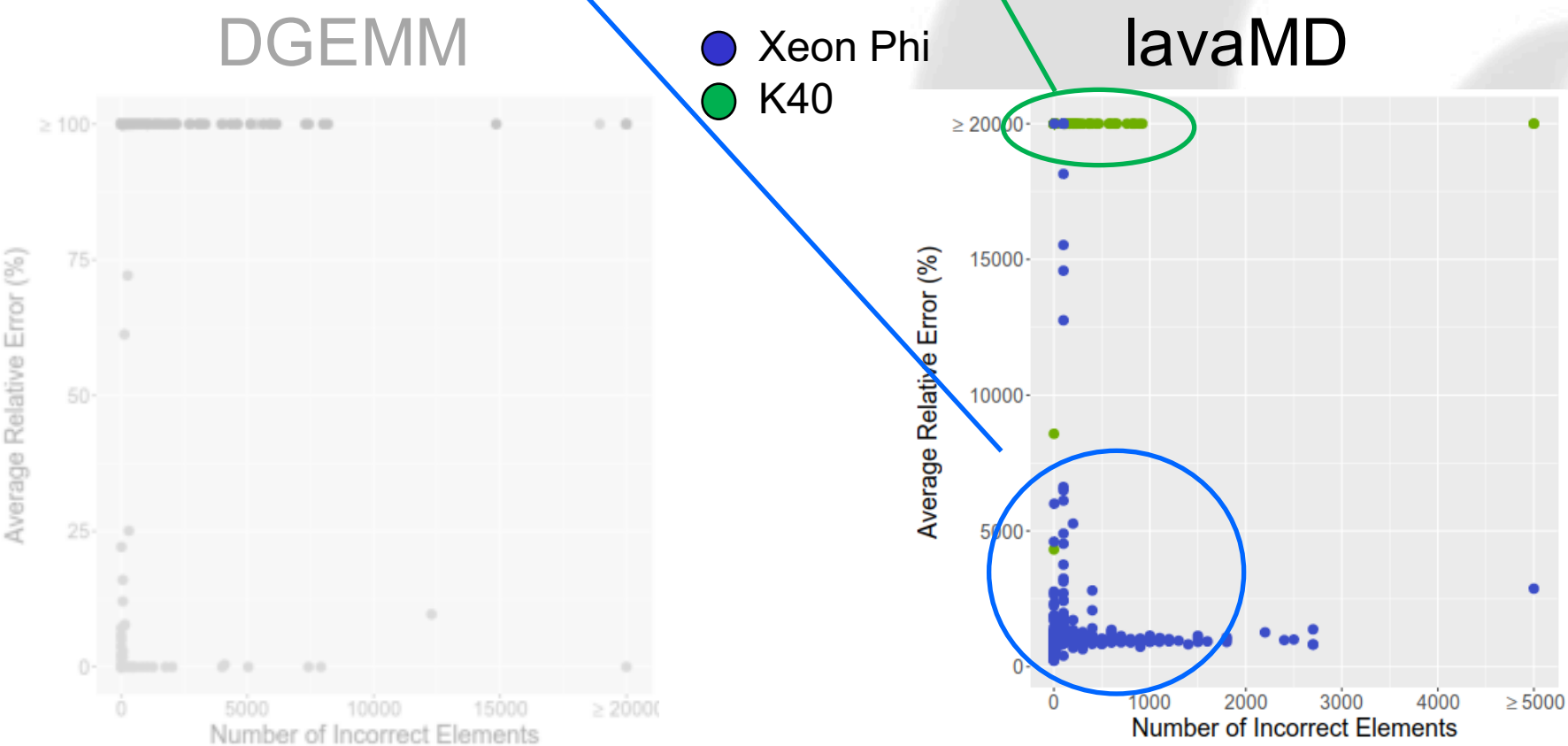


lavaMD



Number of Incorrect Elements vs Relative Error

Both K40 and Xeon Phi have few corrupted elements.
K40 corruption are very different from the expected one

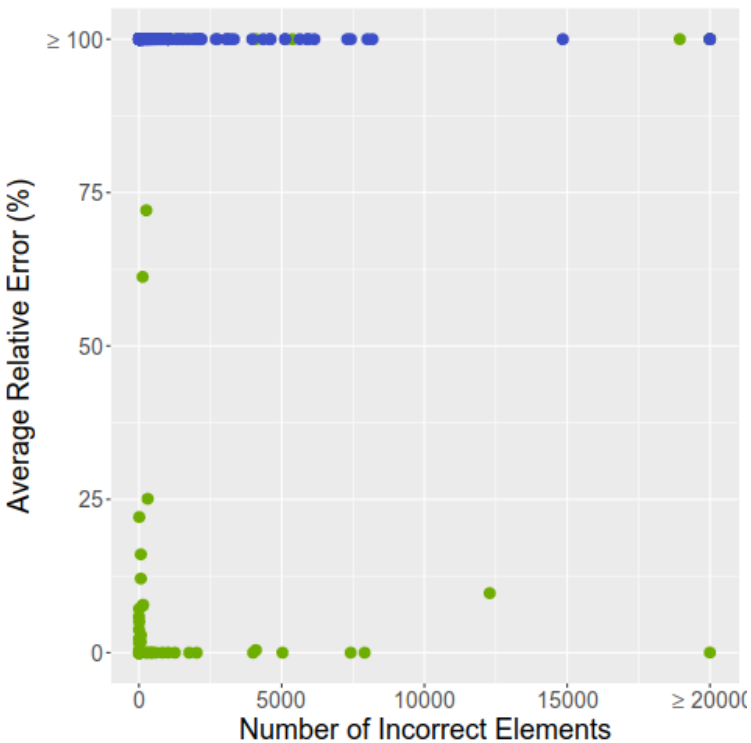


Number of Incorrect Elements vs Relative Error

Purely arithmetic operations are more reliable (and faster) on the K40 (GPUs have shorten and faster pipelines).

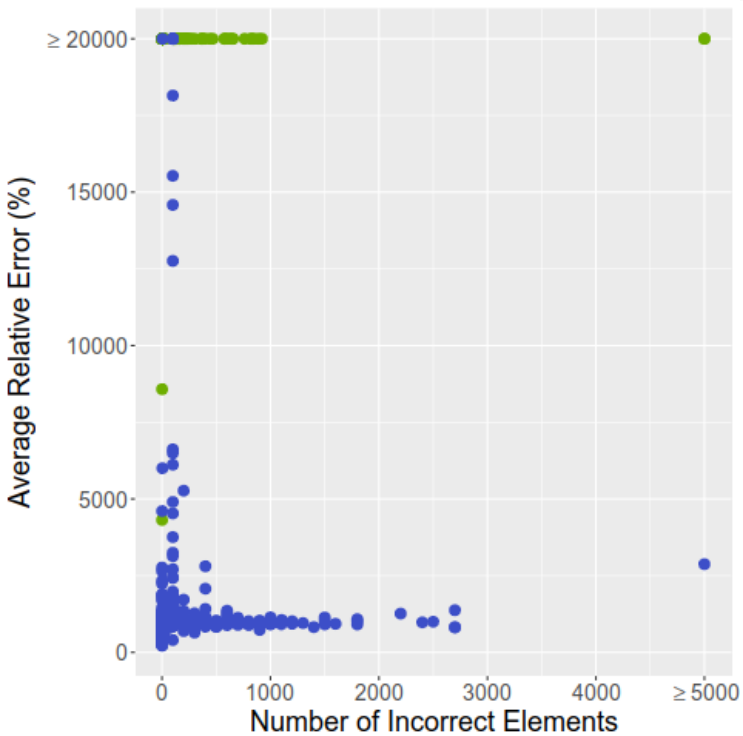
Xeon Phi is more reliable for Finite Different Methods (lavaMD), which are based on transcendental functions (exp).

DGEMM



Xeon Phi
K40

lavaMD



Outline

The origins of the issue:

- Radiation Effects Essentials
- Error Criticality in HPC

Understand the issue:

- Experimental Procedure
- K40 vs Xeon Phi

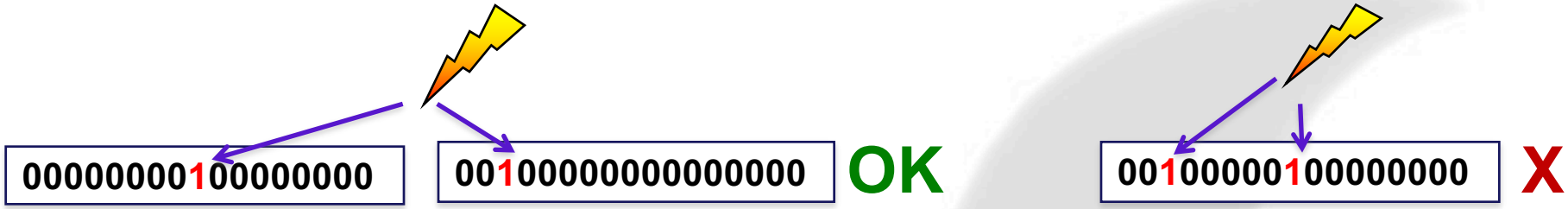
Toward the solution of the issue:

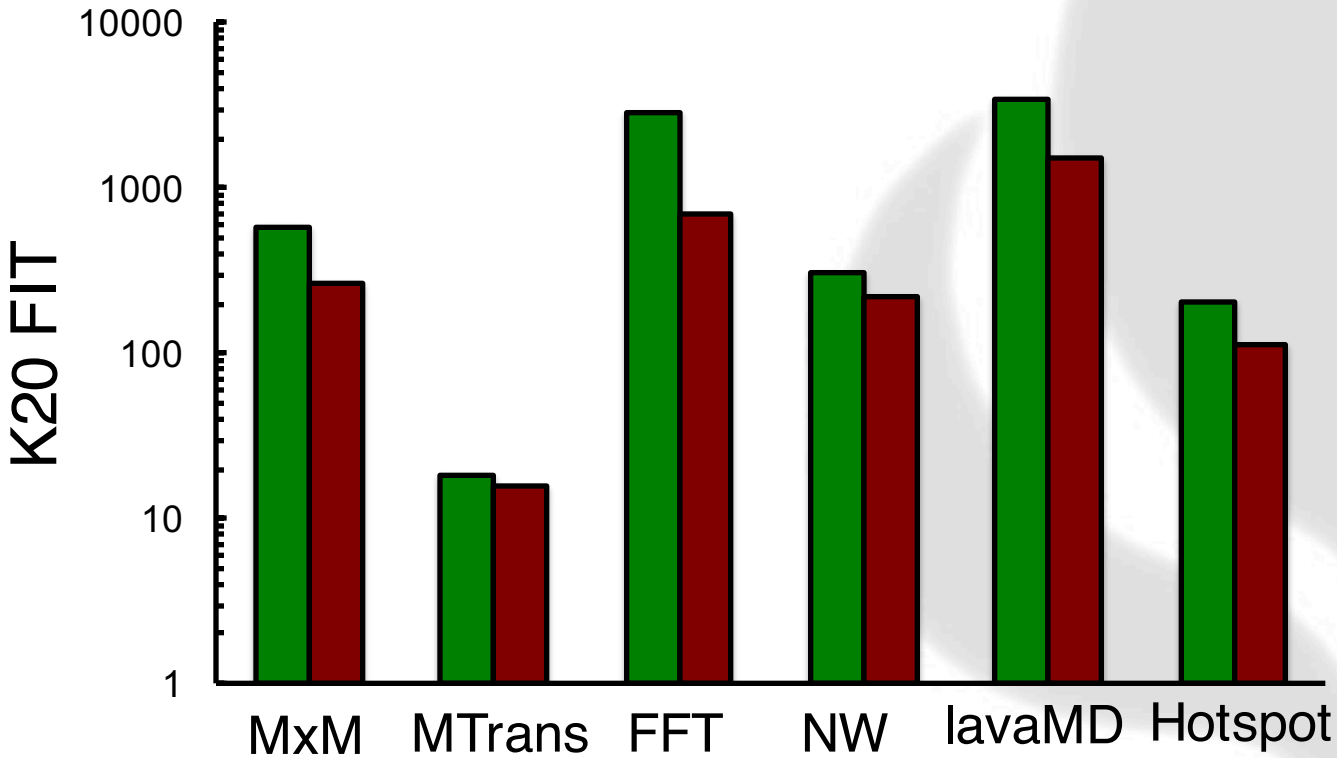
- **ECC – ABFT – Duplication**
- **Selective Hardening**

What's the Plan?

Experimental Results (ECC OFF)

Single Error Correction Double Error Detection ECC.





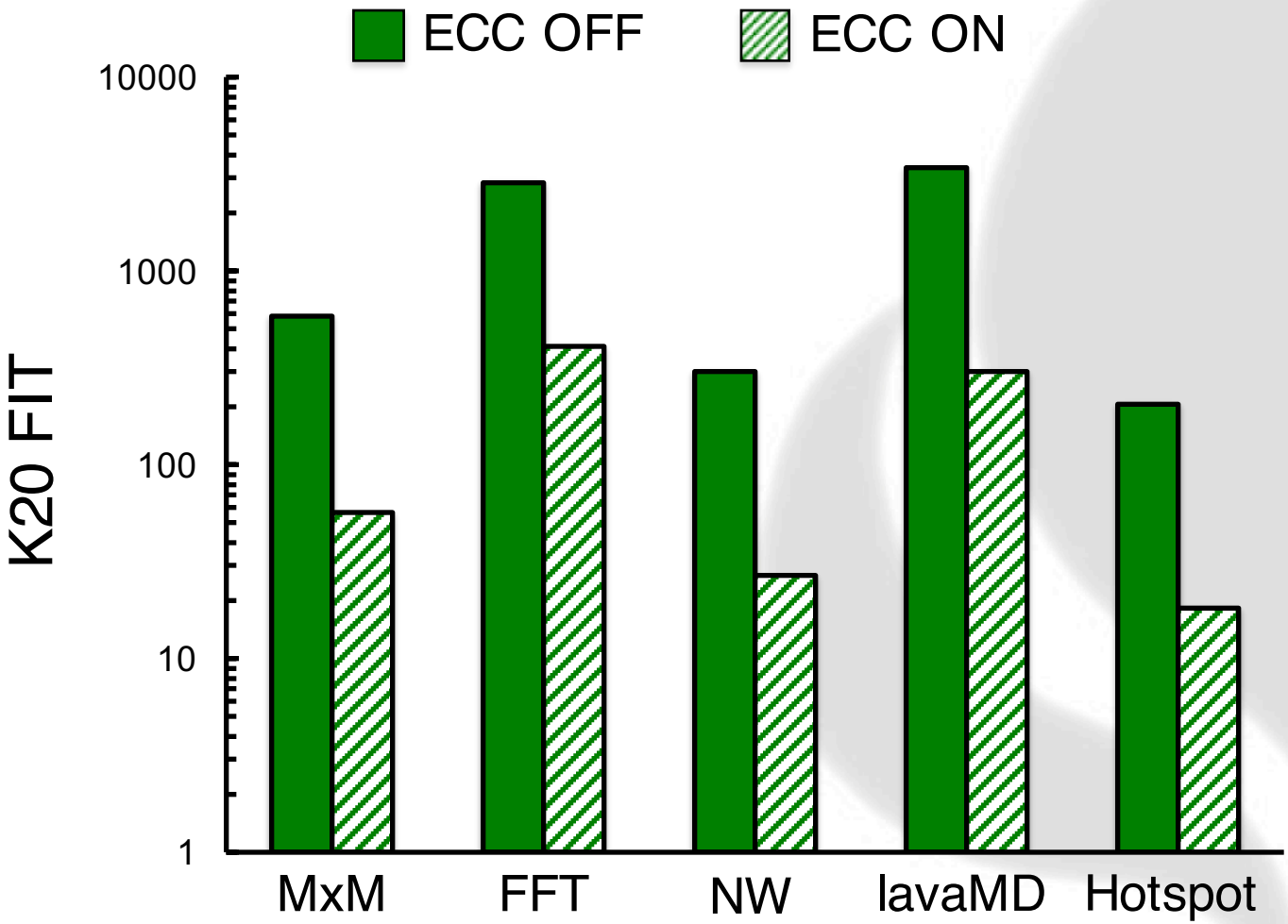
Crashes

SDC

data from Oliveira et al. Trans. Comp. 2016

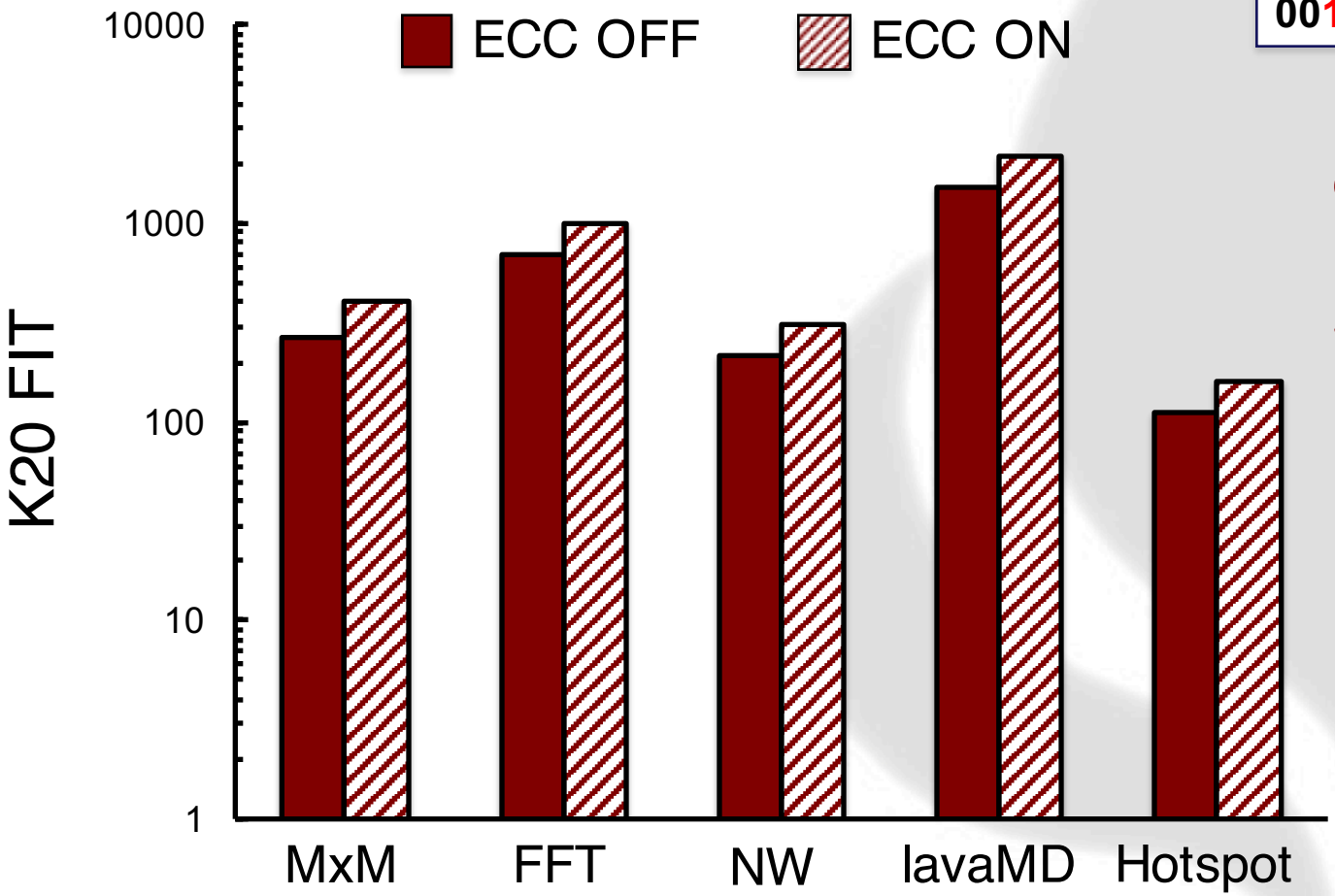
ECC ON - SDC

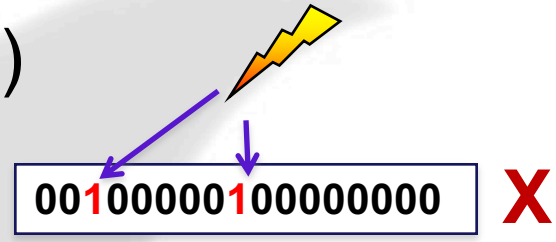
ECC reduces the **SDC FIT** of ~1 order of magnitude (there is almost no code dependence)



ECC ON - Crash

ECC increases the **Crash FIT** of about 50%
(there is almost no code dependence)



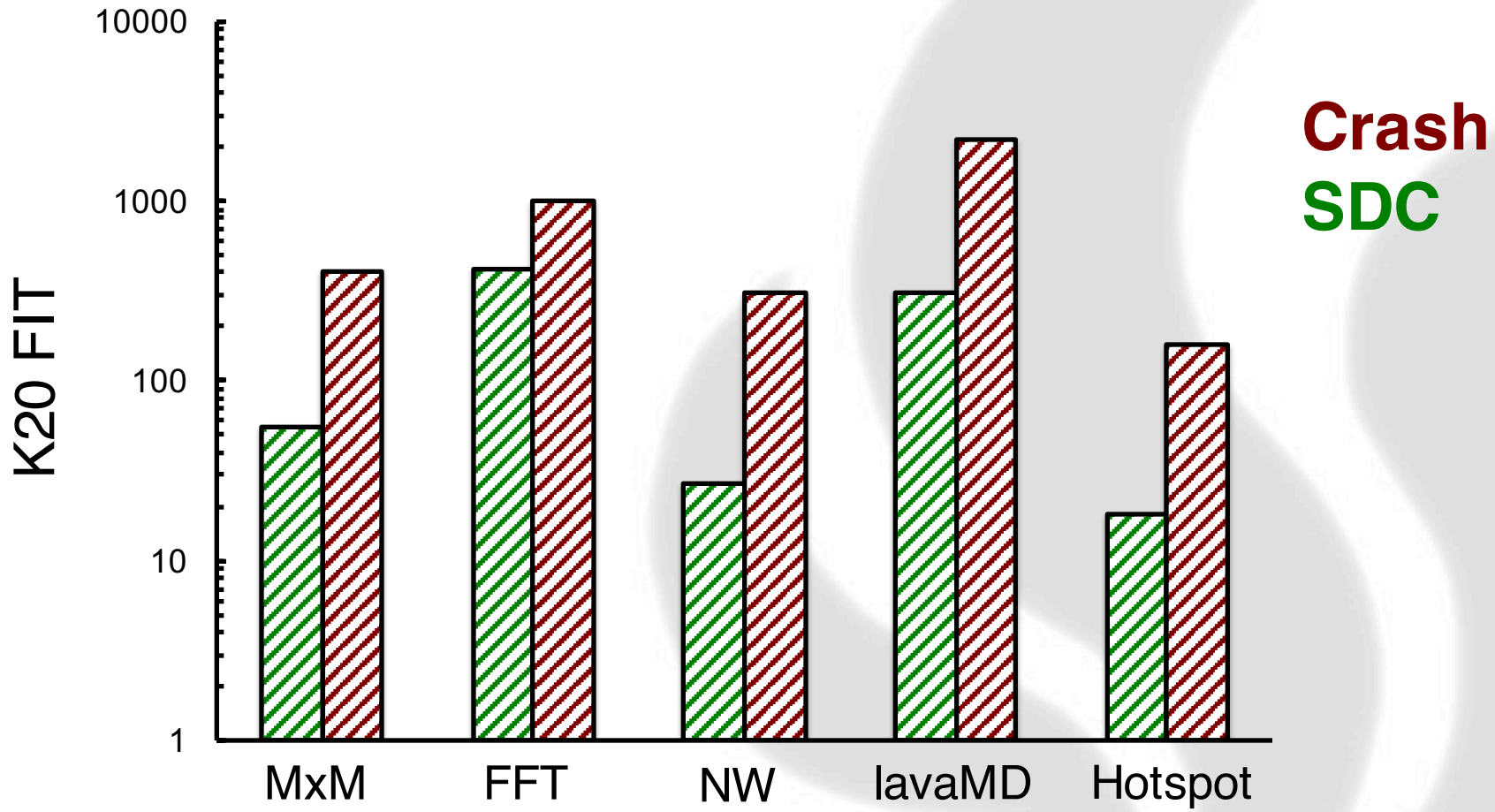


Double Bit Errors cause a crash

scheduler is not protected

ECC ON – SDC vs Crashes

When the ECC is ON **Crashes** are more likely to occur than **SDCs (this is GOOD for HPC centers!)**

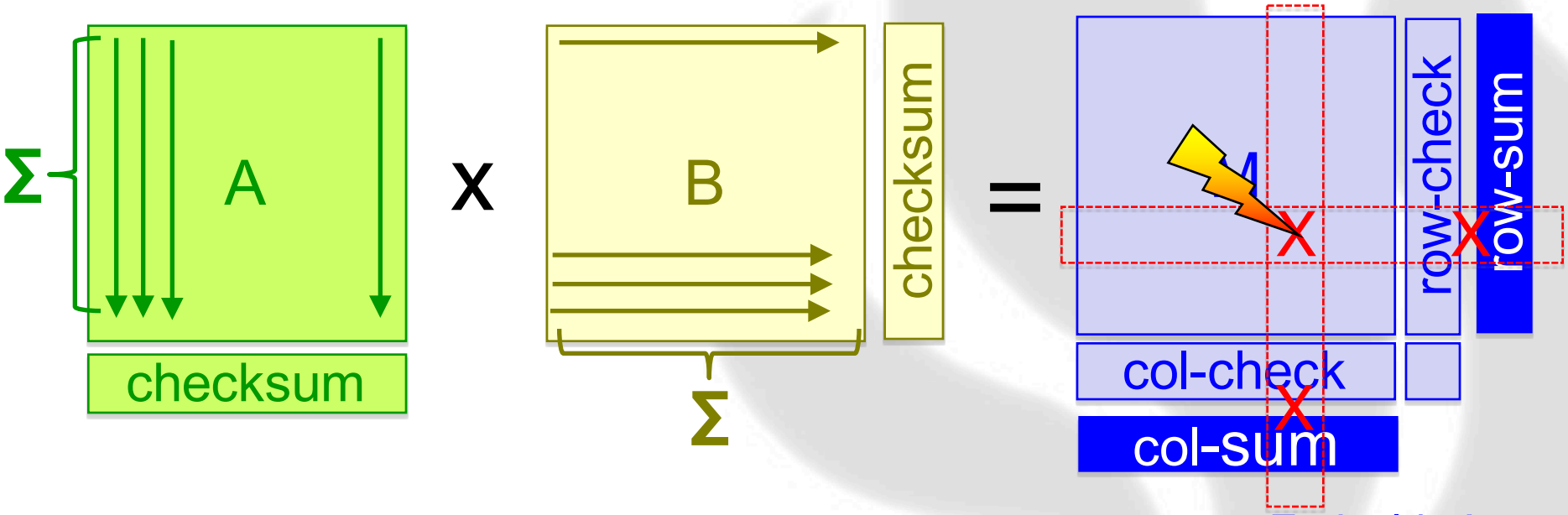


Algorithm Based Fault Tolerance

ABFT: technique designed specifically for an algorithm.

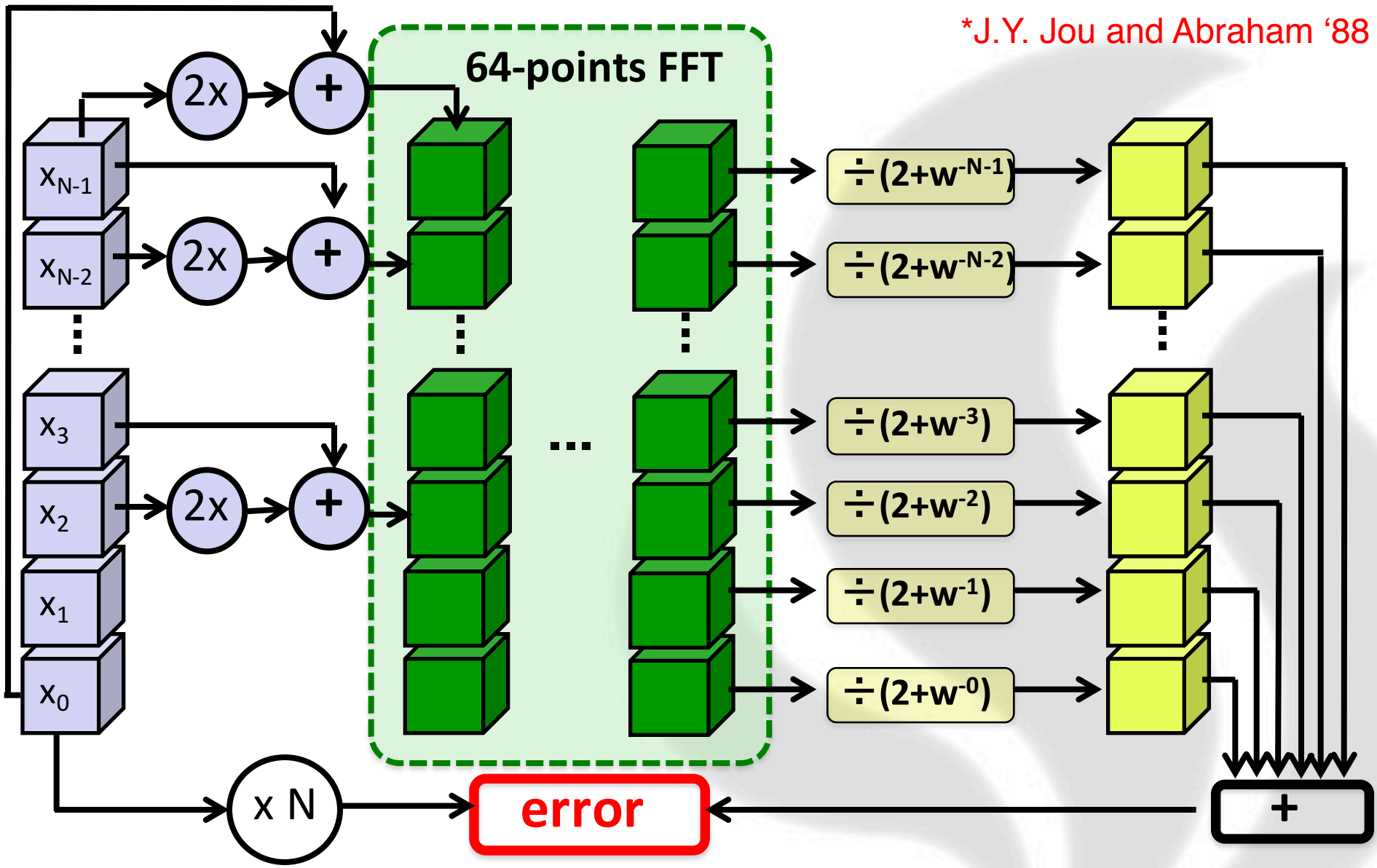
ABFT requires: **input coding**, **algorithm modification**, and **output decoding** with error detection/correction

Huang and Abraham '84
Rech et al., TNS '13



Freivalds '79

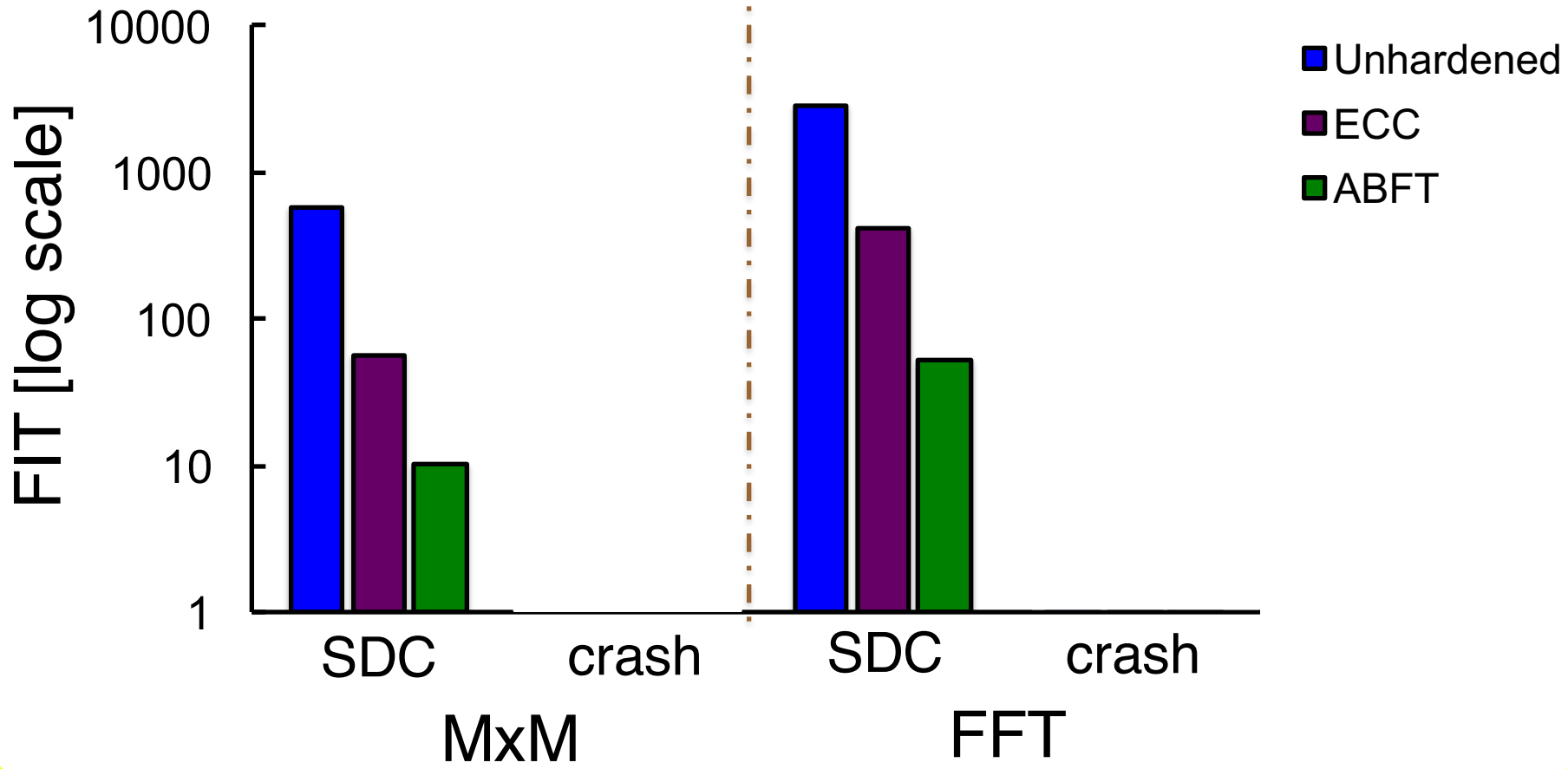
FFT Hardening Idea*



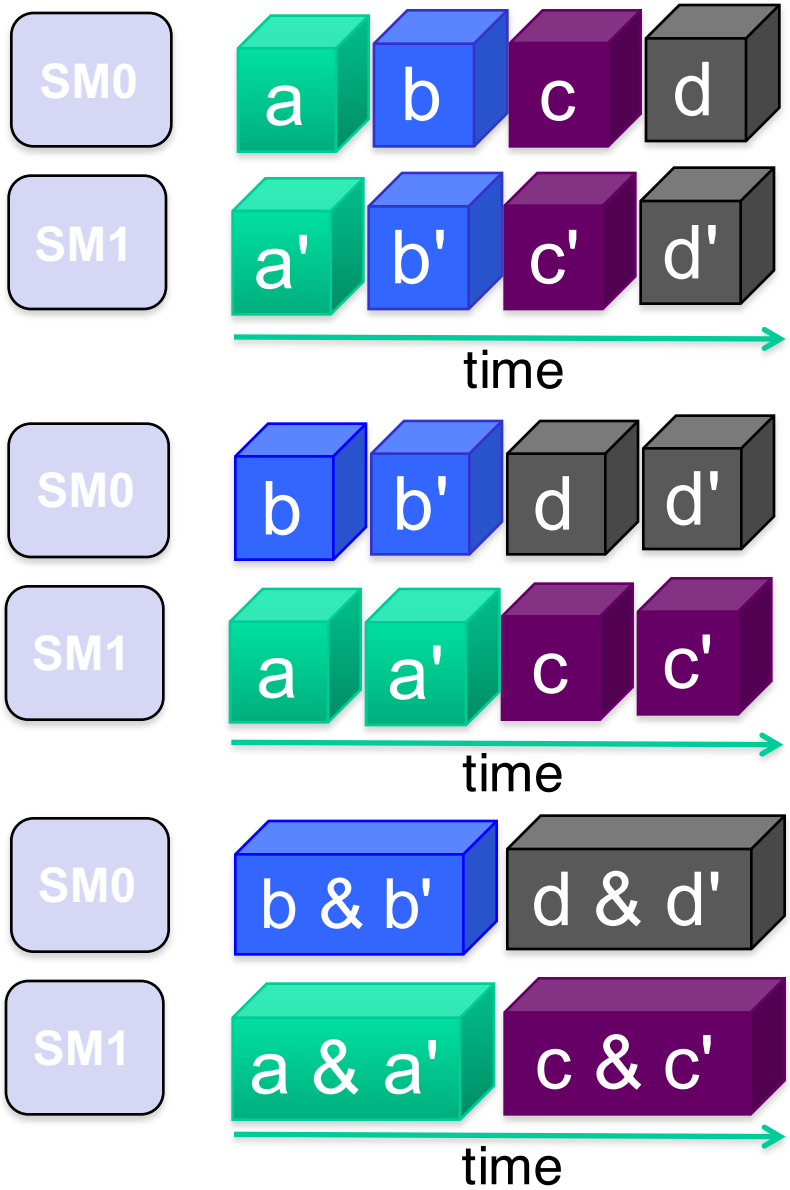
ECC vs ABFT

ECC reduces FIT of ~10 times, ABFT of ~56 times!

ECC increases Crashes of 50% ABFT of 10%!



Duplication With Comparison



Spatial: block i and i+N are duplicated

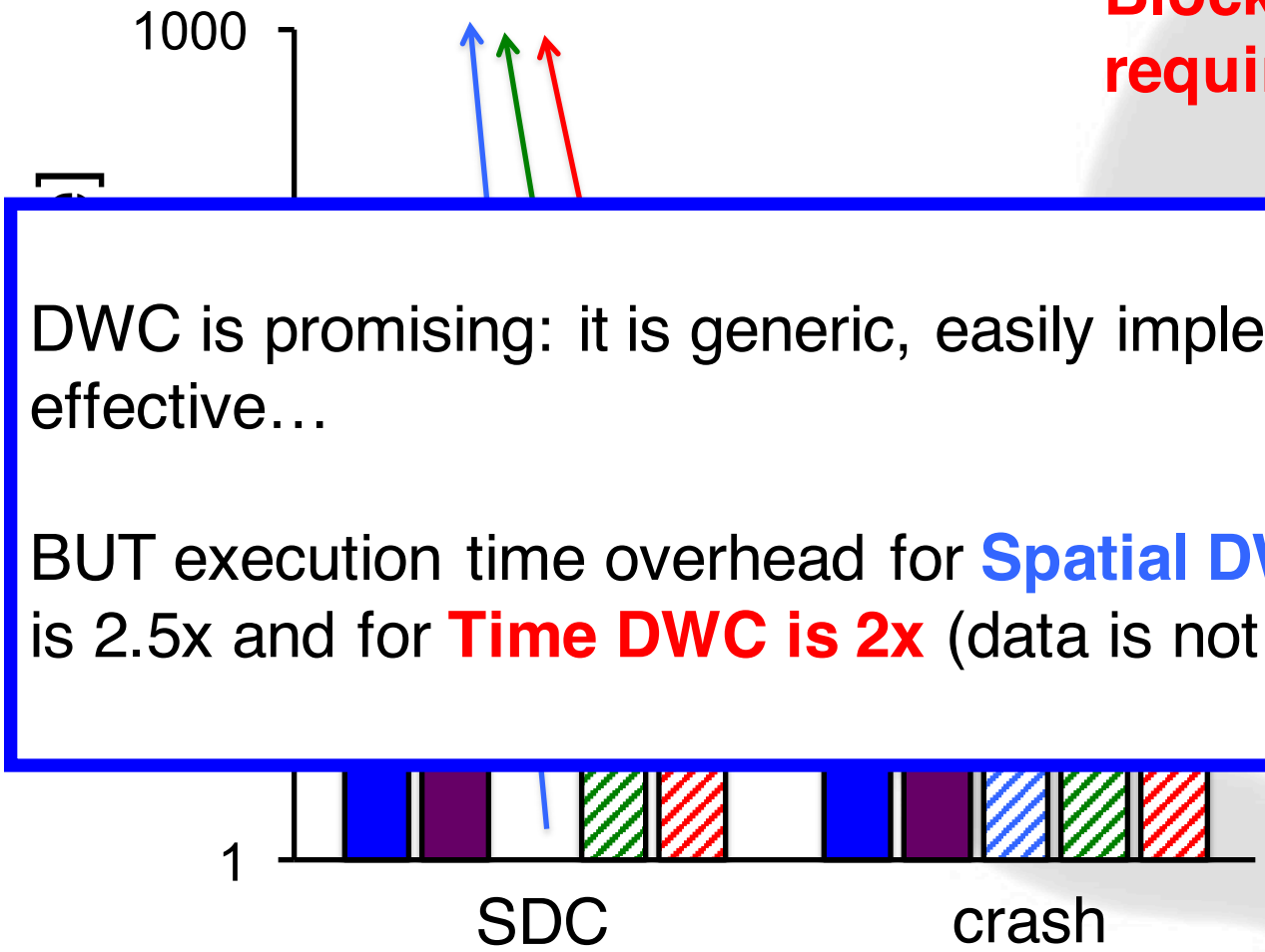
E-O Spatial: block i and i+1 are duplicated

Time: a thread executes twice the operations

Hotspot - DWC results*

Spatial DWC detects all SDC
Spatial E-O detects 80% of SDC
Time DWC detects 90% of SDC

Only Time DWC reduces Crashes (no additional Blocks scheduling required)



DWC is promising: it is generic, easily implemented, and effective…

BUT execution time overhead for **Spatial DWC** and **Spatial E-O** is 2.5x and for **Time DWC is 2x** (data is not copied)

*details on Oliveira et al. Trans. Nucl. Sci., 2014

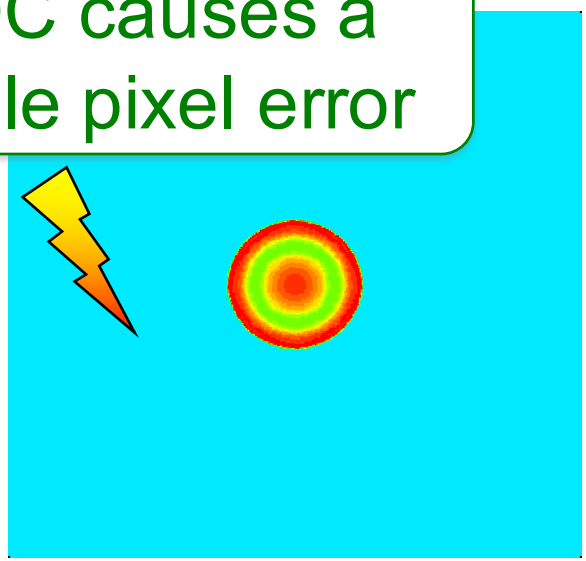
What's next? Selective Hardening!

Duplicate only what REALLY matters

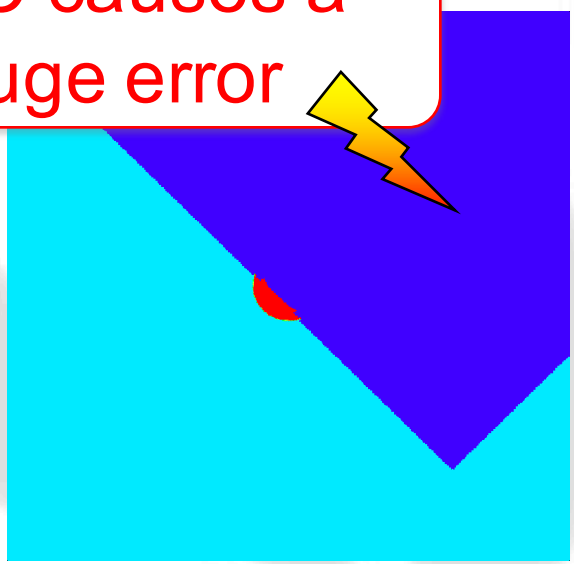
analyze SDC criticality: are there "acceptable" SDCs?

example: CLAMR (DOE workload) experimental result

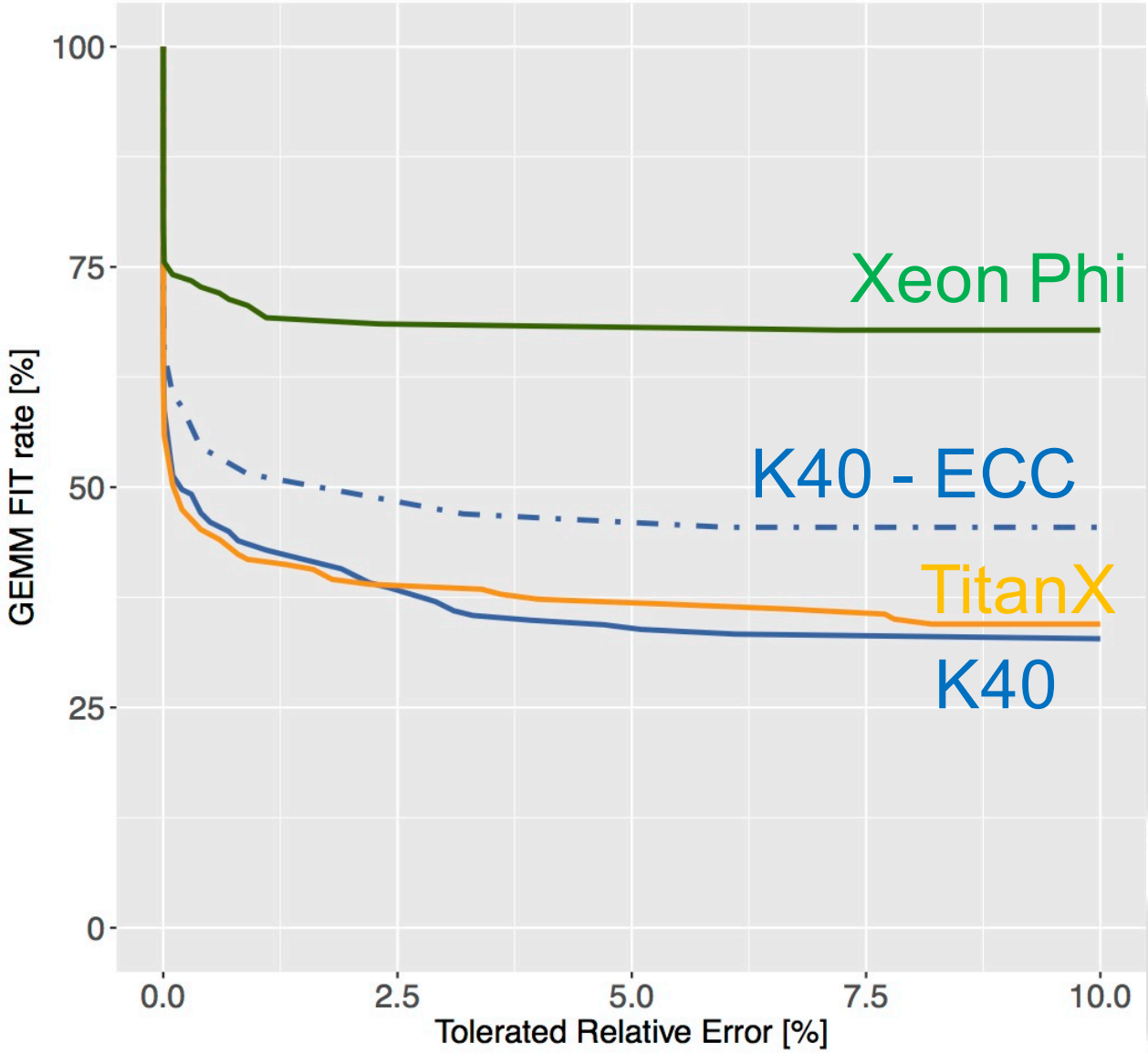
SDC causes a single pixel error



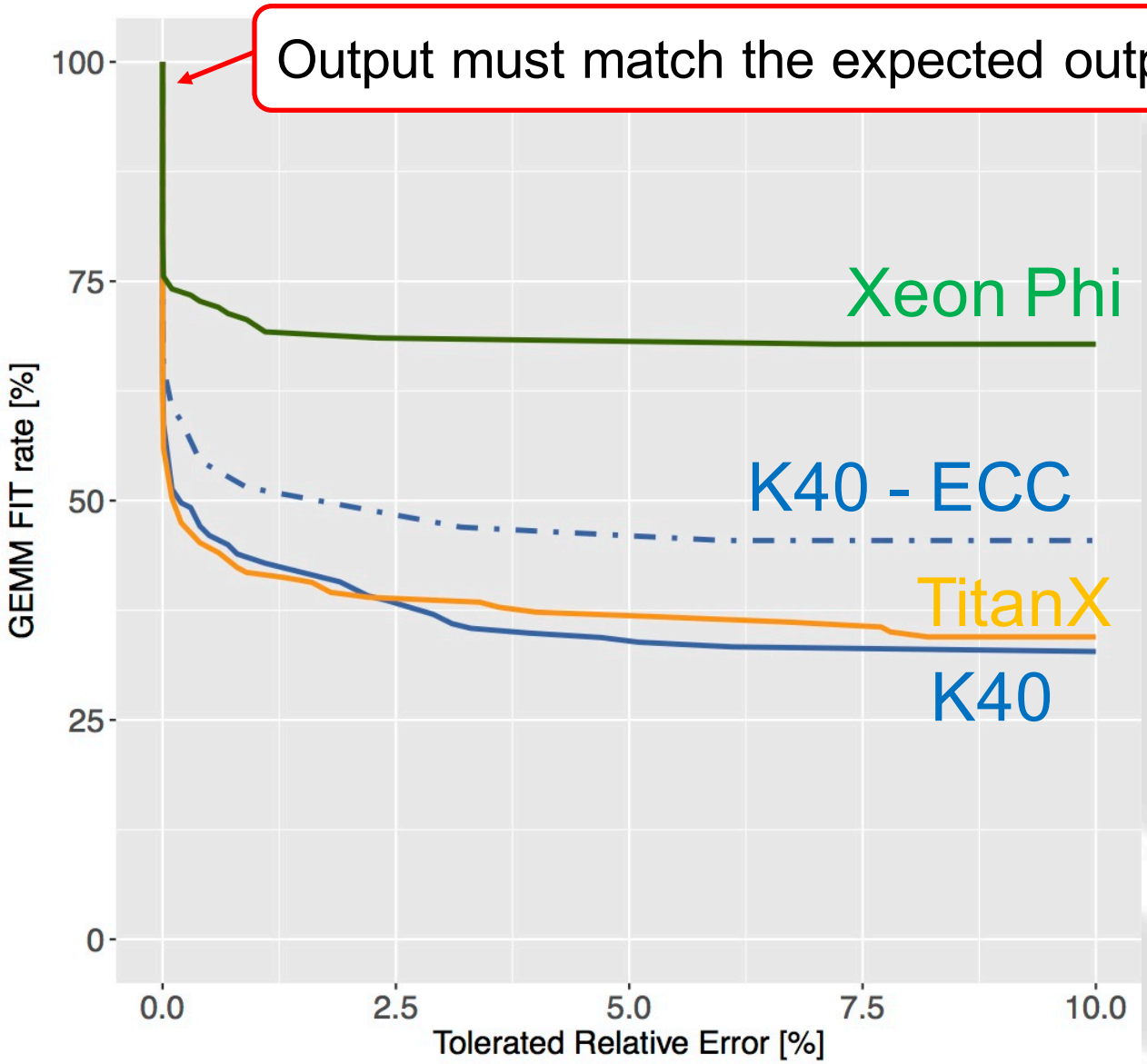
SDC causes a huge error



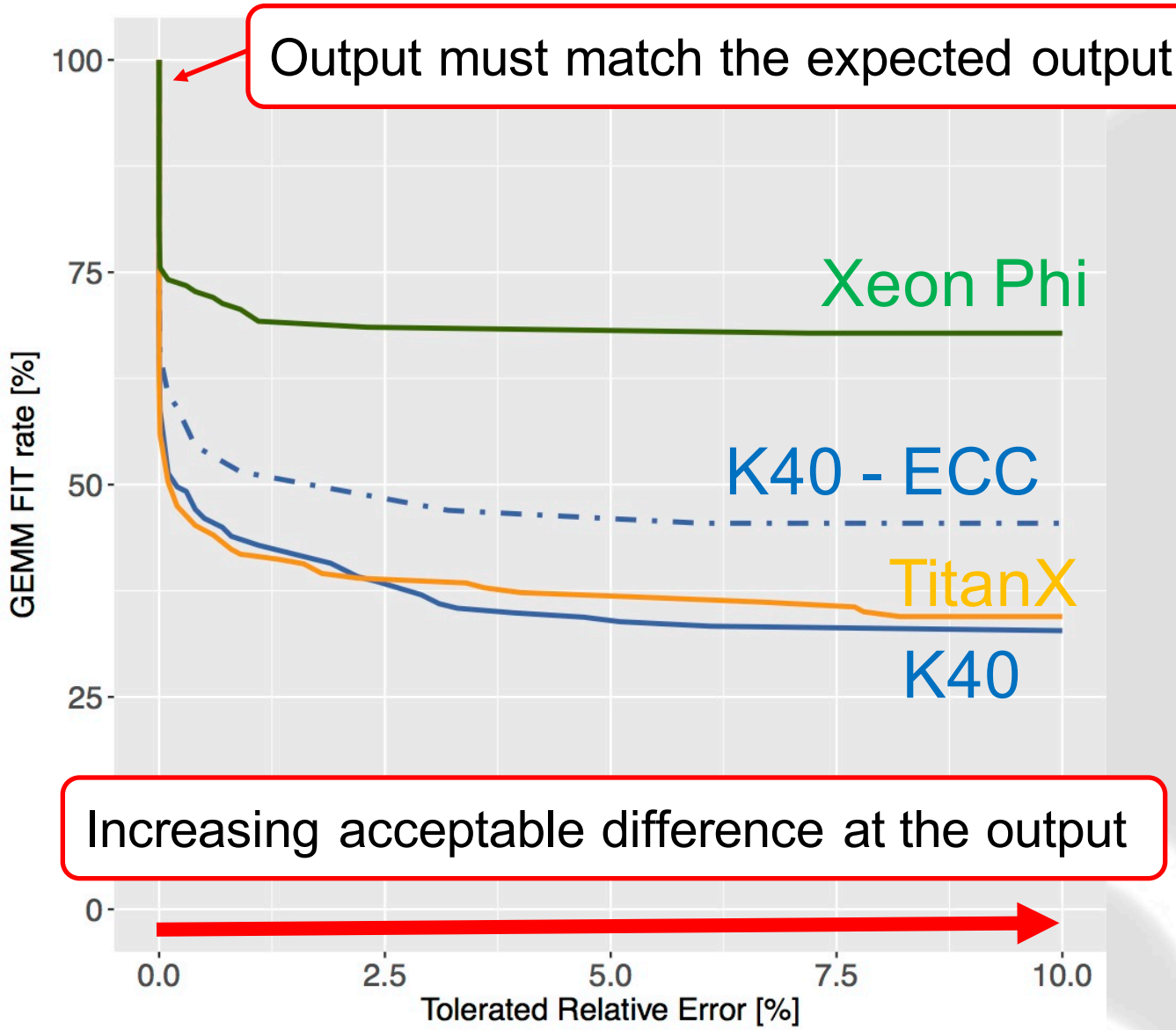
Tolerable SDCs



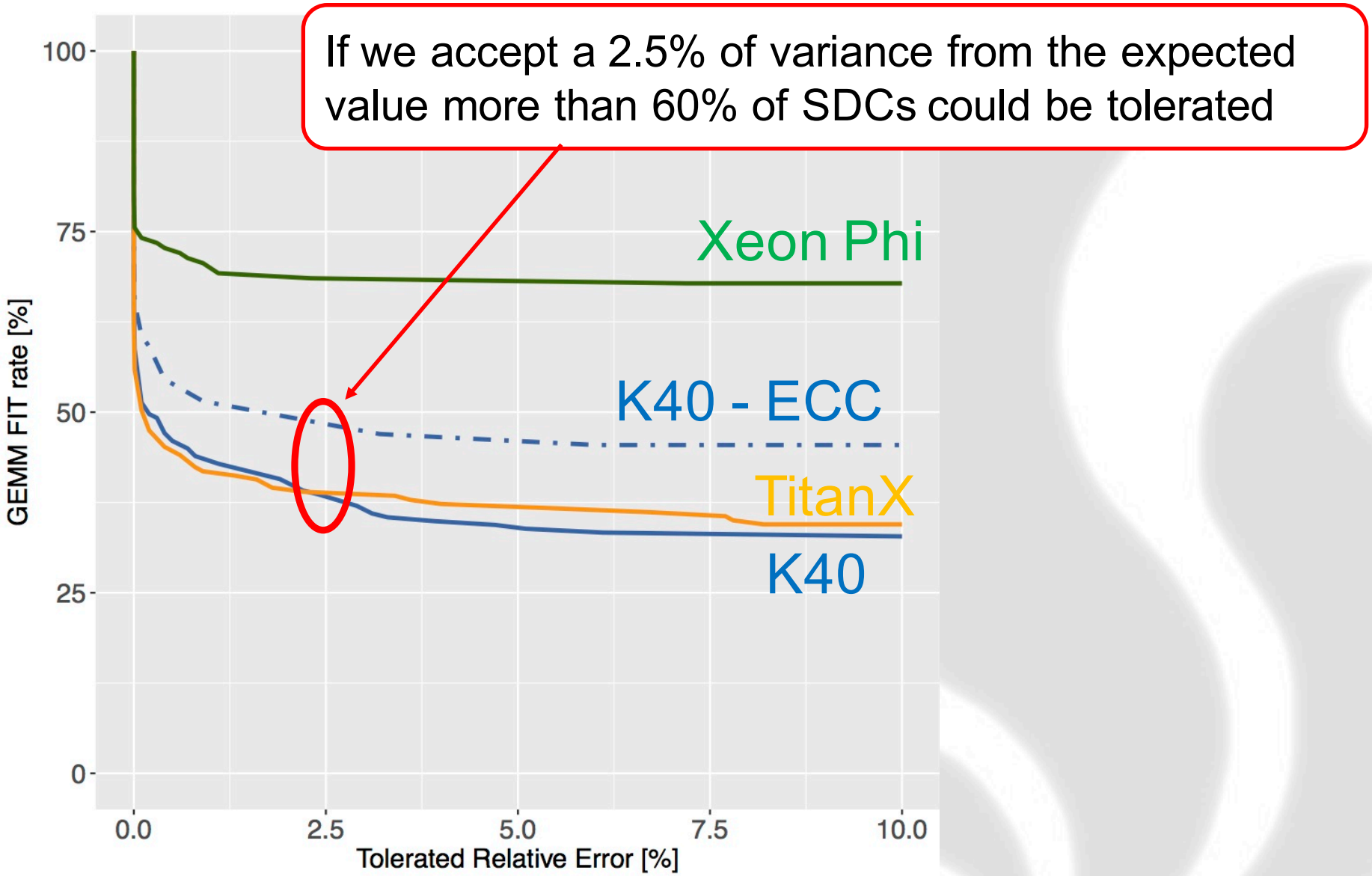
Tolerable SDCs



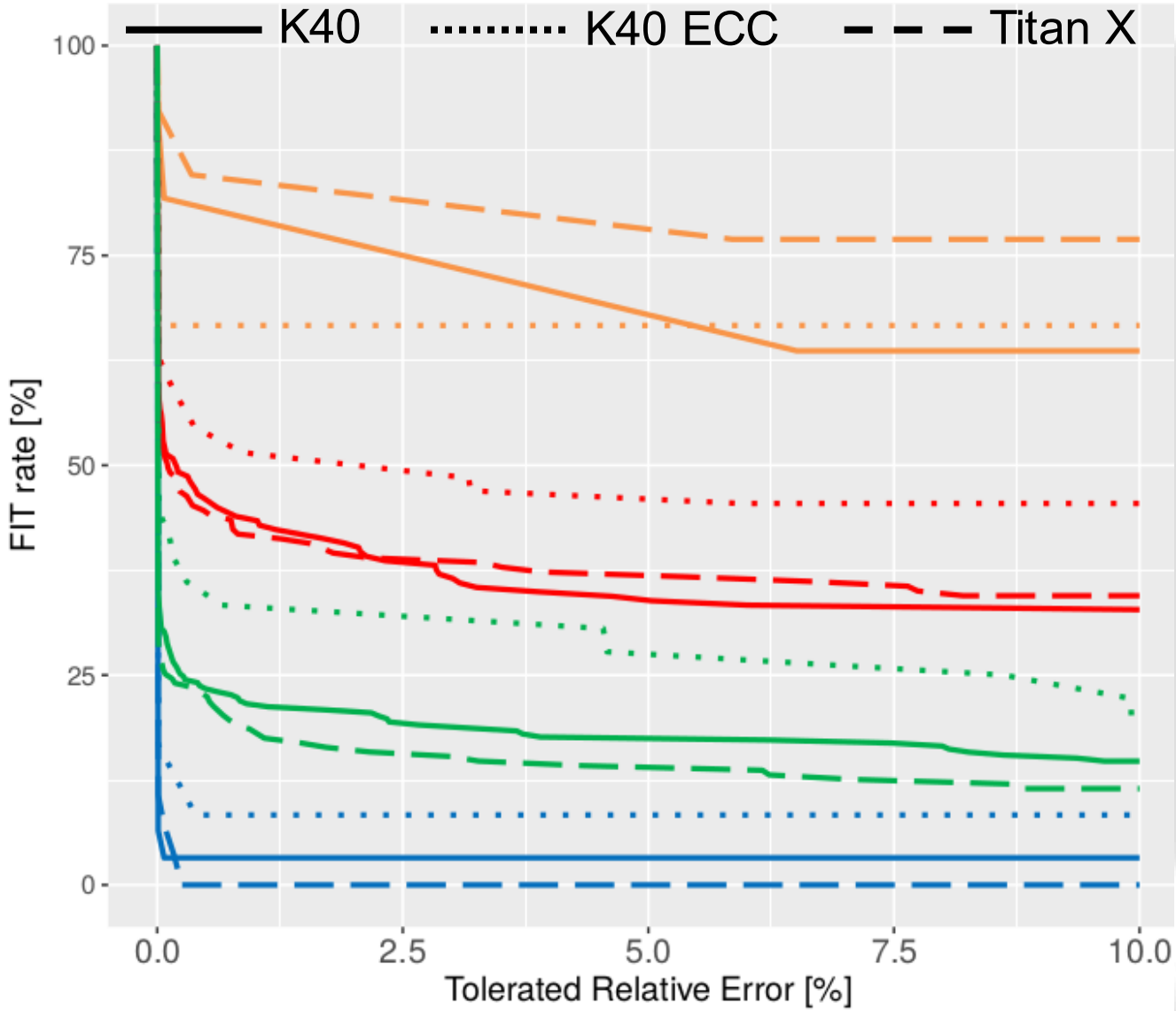
Tolerable SDCs



Tolerable SDCs



Tolerable SDCs



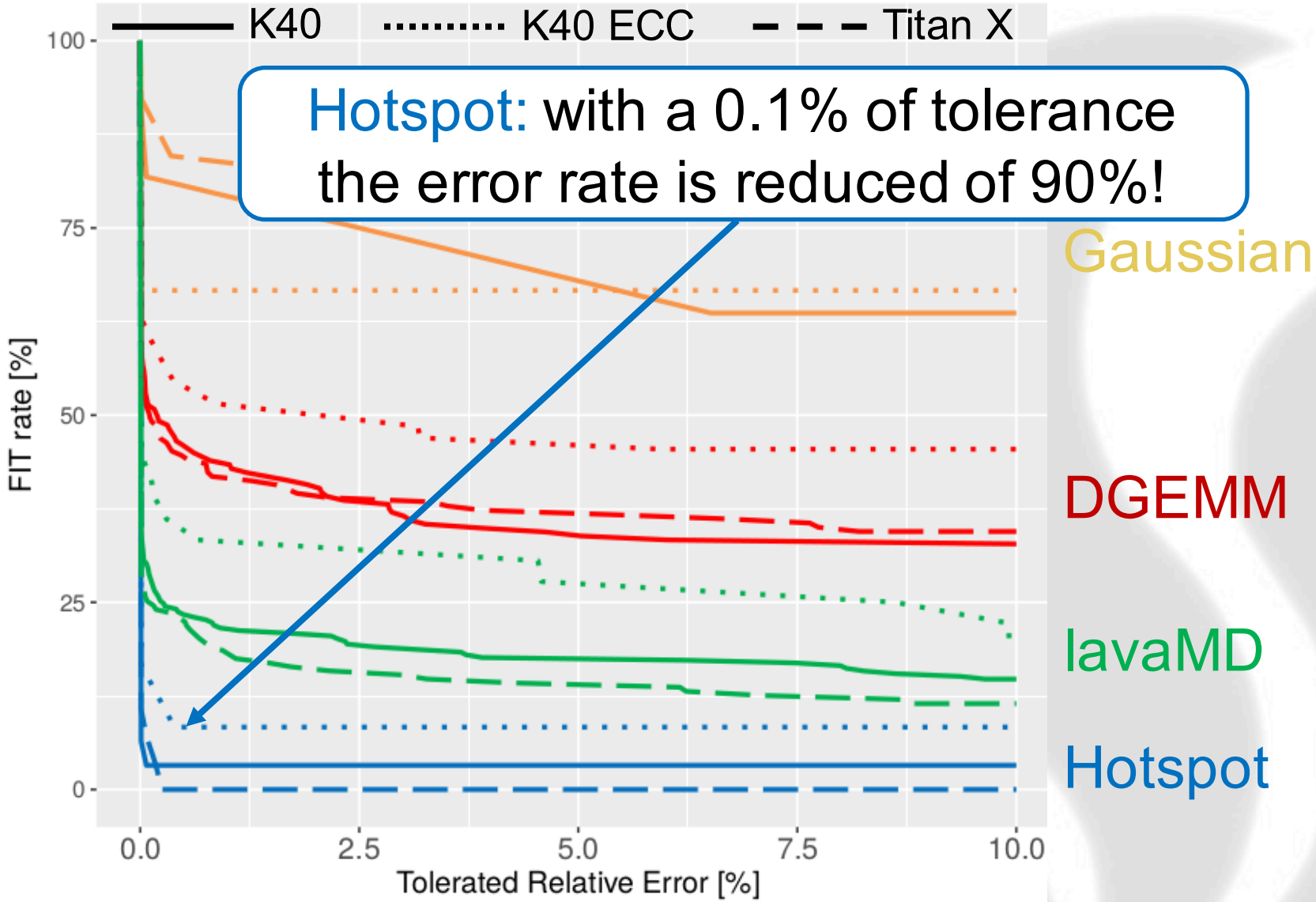
Gaussian

DGEMM

lavaMD

Hotspot

Tolerable SDCs



What's next? Selective Hardening!

Duplicate only what REALLY matters

1. analyze SDC criticality: are there "acceptable" SDCs?

2. detect SW-HW causes for critical SDCs
   - -code analysis
   - -fault-injection
   - (NVIDIA SASSIFI and UFRGS CAROL-FI)

What's next? Selective Hardening!

Duplicate only what REALLY matters

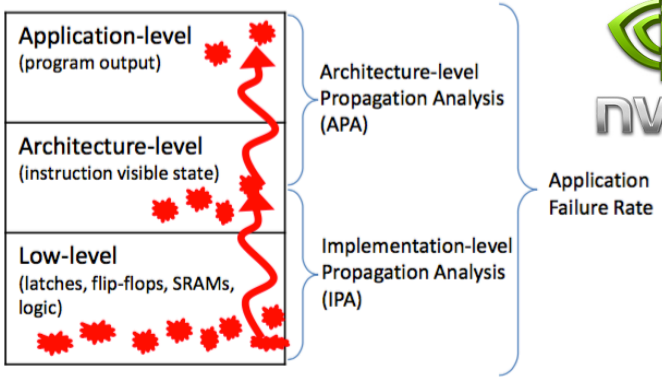
1. analyze SDC criticality: are there "acceptable" SDCs?

2. detect SW-HW causes for critical SDCs
   - -code analysis
   - -fault-injection
   - (NVIDIA SASSIFI and UFRGS CAROL-FI)

3. harden selected portions of the code

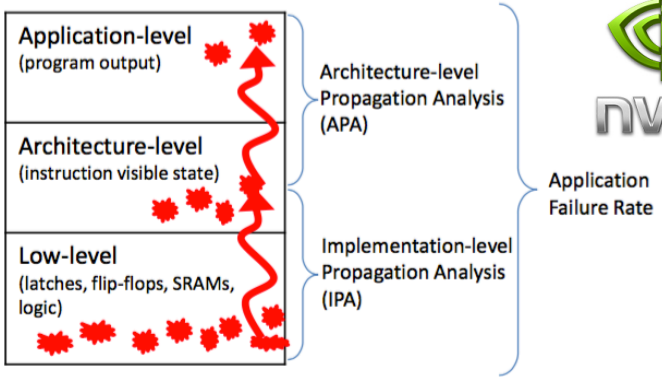
4. evaluate enhanced reliability and performances

SASSI-FI and CAROL-FI



SASSI-FI: NVIDIA architectural-level fault-injector

SASSI-FI and CAROL-FI



SASSI-FI: NVIDIA architectural-level fault-injector

CAROL-FI: UFRGS high-Level Fault Injector for Xeon-Phi and any X86-base processor

Modify content of memory currently allocated.

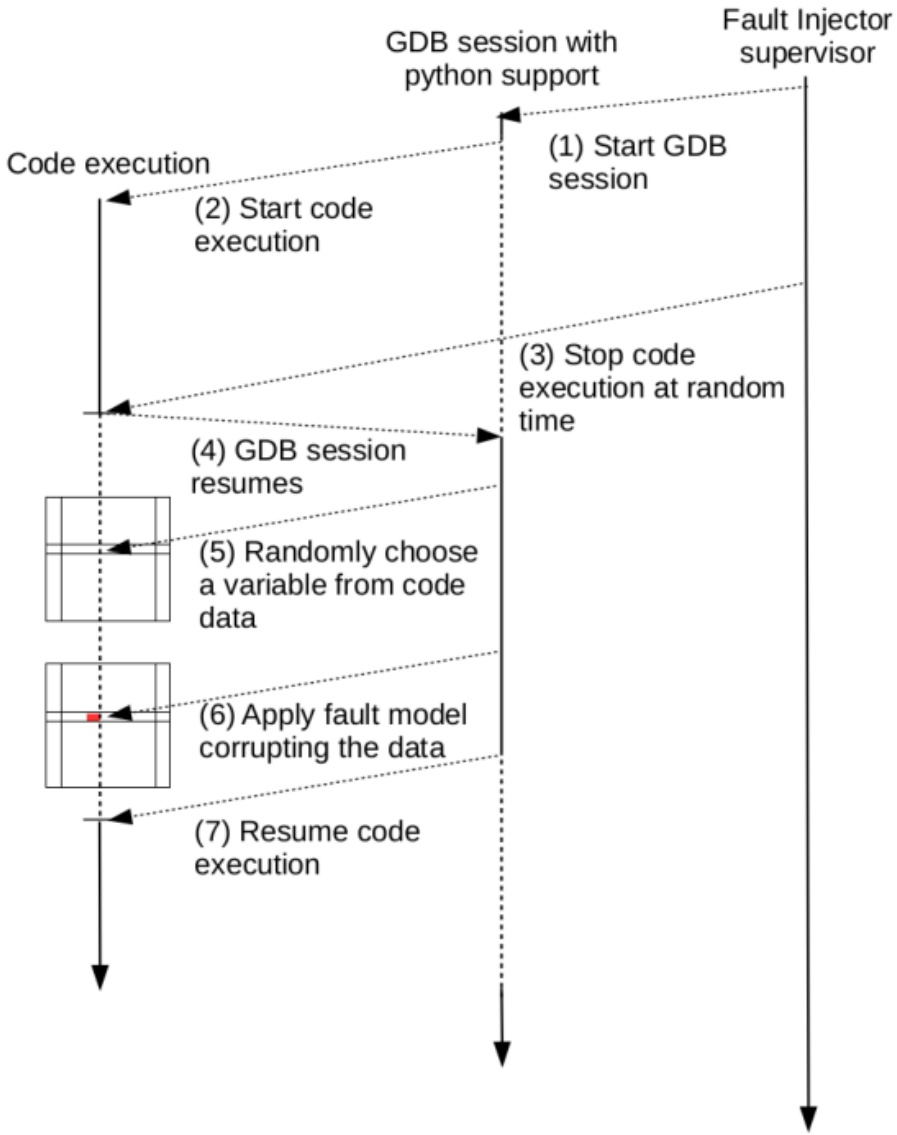
Fault Injector requirements:

–GDB with python support

–OS Interruption signals

–Compile the source code in debug mode

CAROL-FI



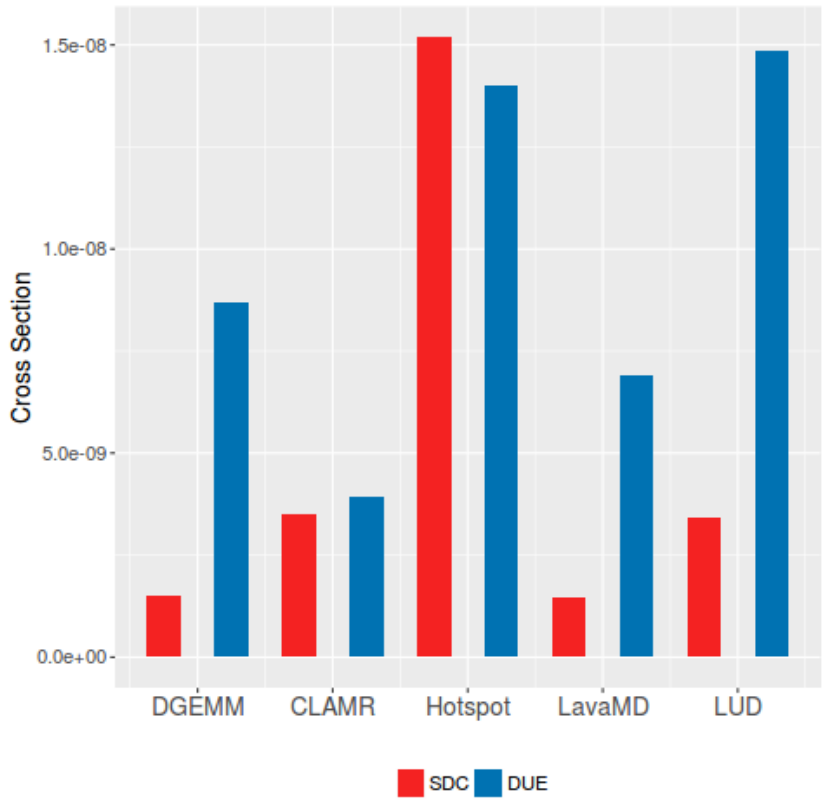
Fault model can be adapted
We only inject single bit-flip

Overhead ~5x

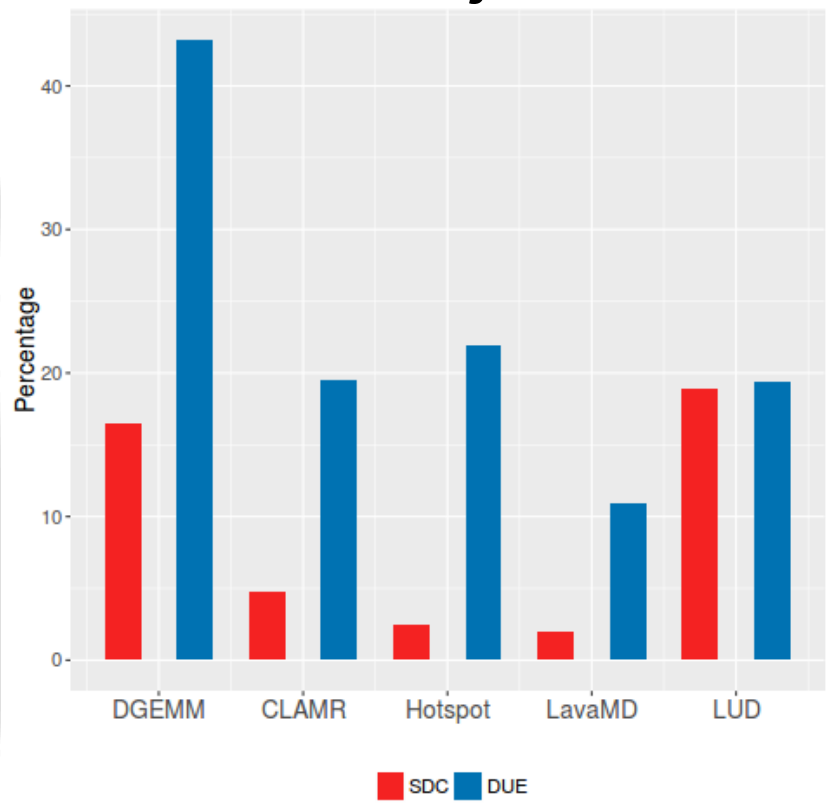
Radiation Data vs CAROL-FI

Radiation and FI give very different information.

Radiation

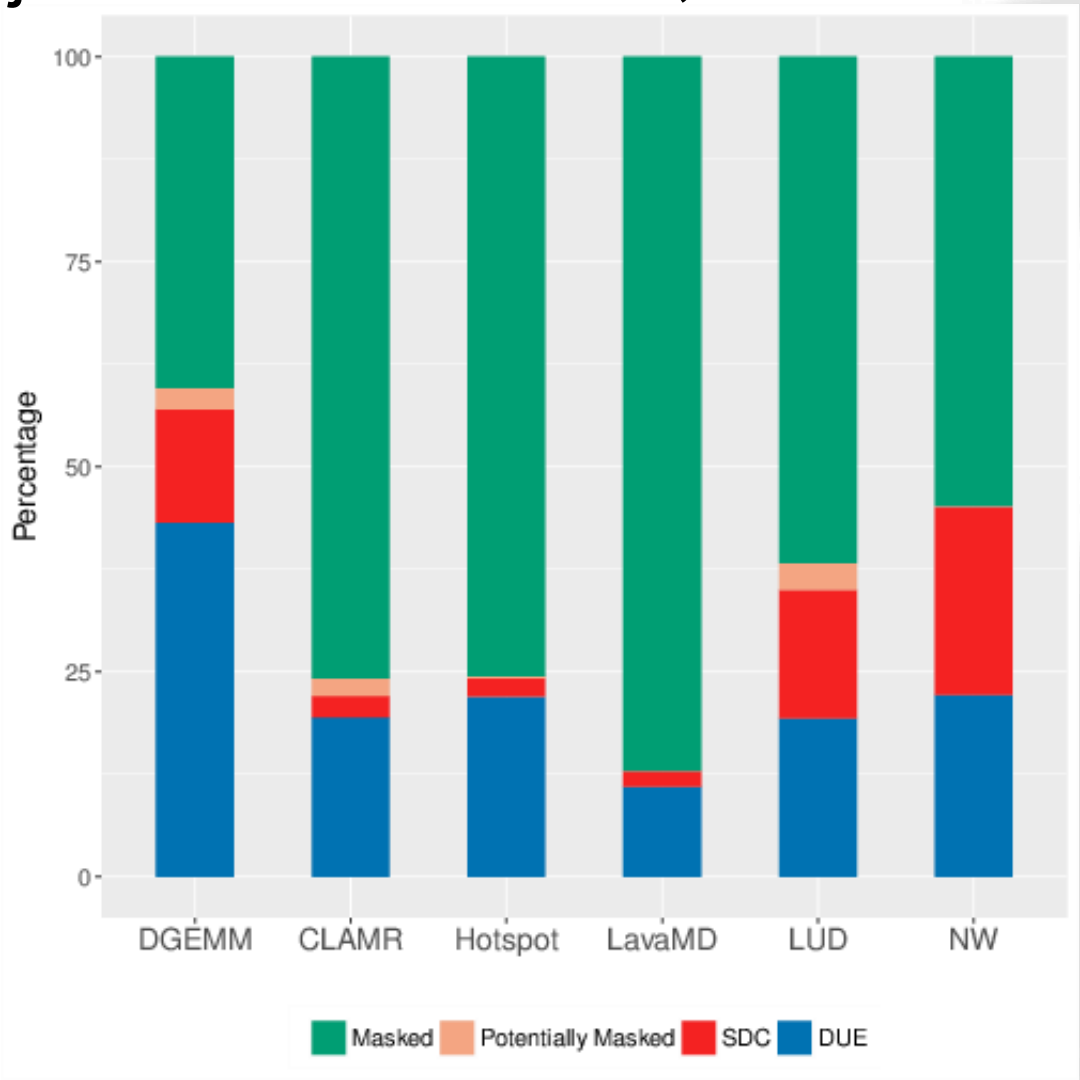


Fault-injection

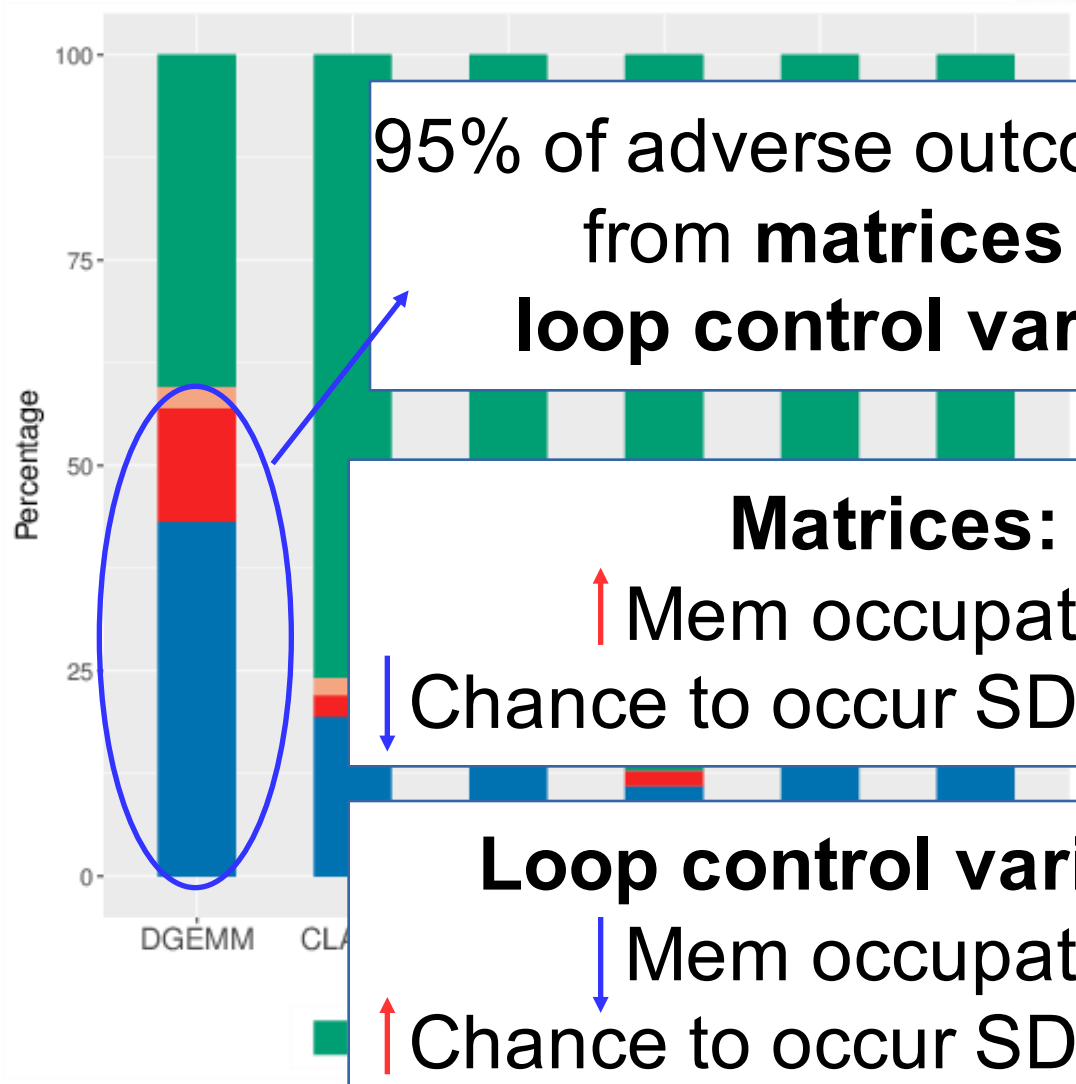


CAROL-FI Results

We have injected more than 67,000 faults



Results - DGEMM



95% of adverse outcomes come from **matrices** and **loop control variables**

Matrices:

↑ Mem occupation

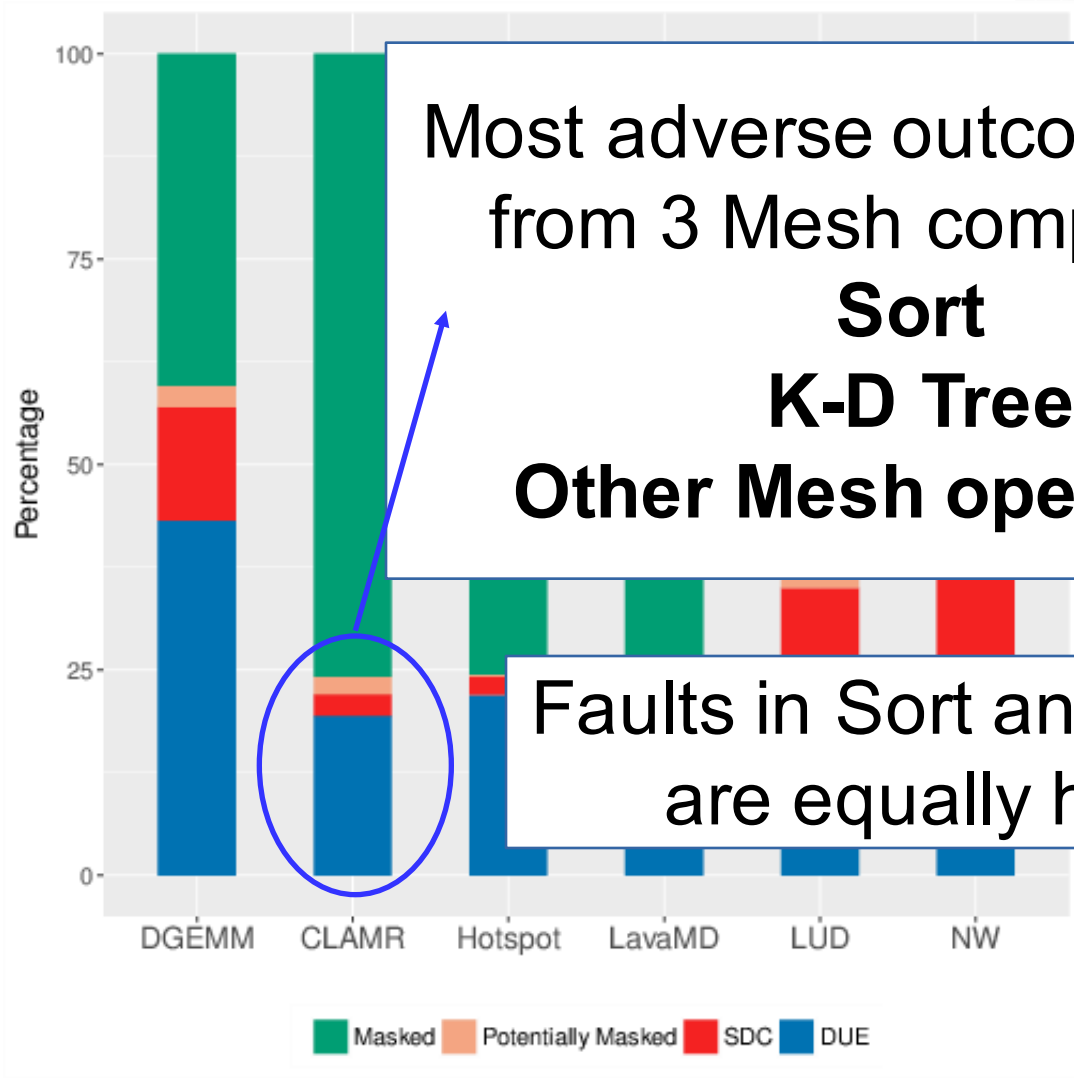
↓ Chance to occur SDC or DUE

Loop control variables:

↓ Mem occupation

↑ Chance to occur SDC or DUE

Results - CLAMR



Most adverse outcomes come from 3 Mesh components

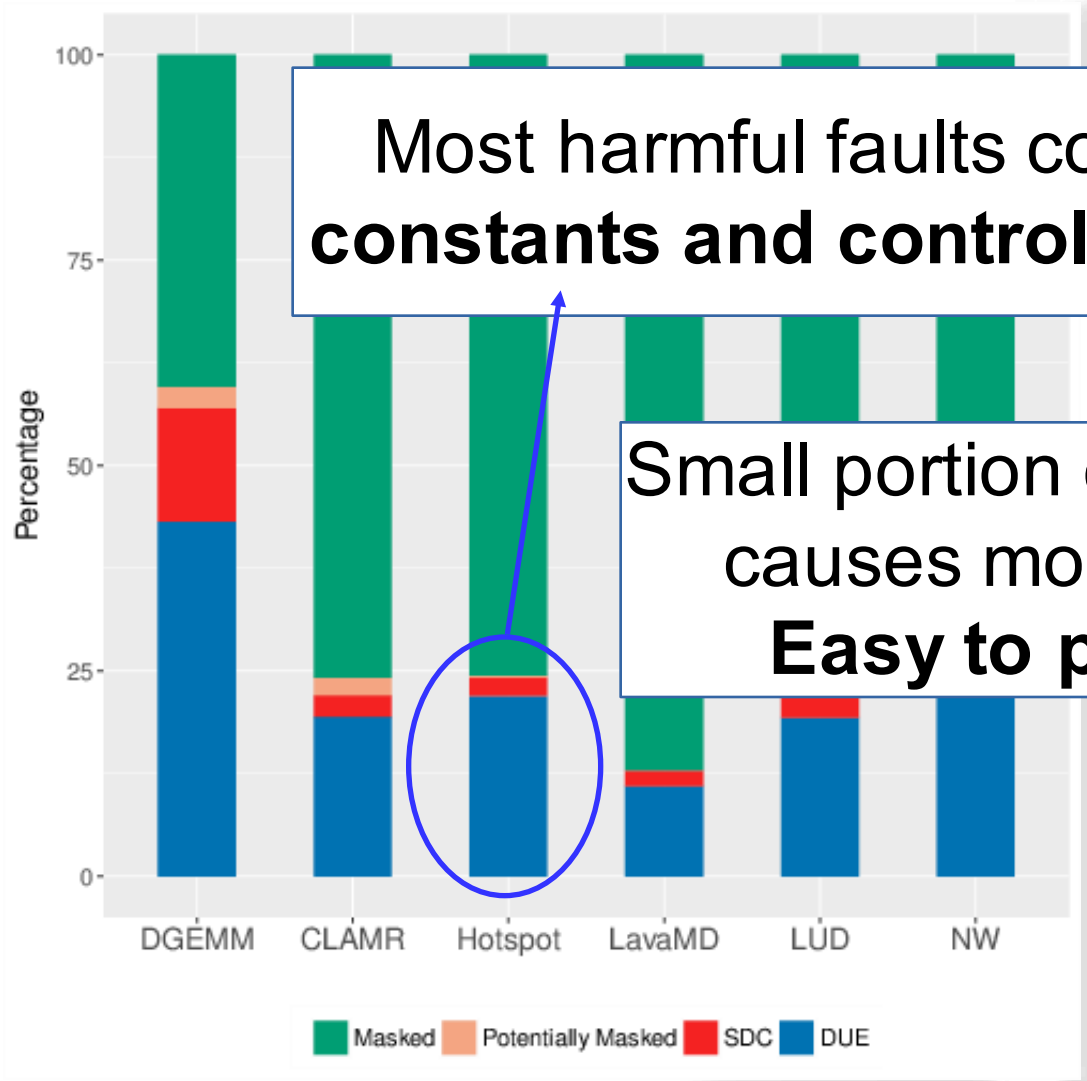
Sort

K-D Tree

Other Mesh operations

Faults in Sort and K-D Tree are equally harmful

Results - Hotspot



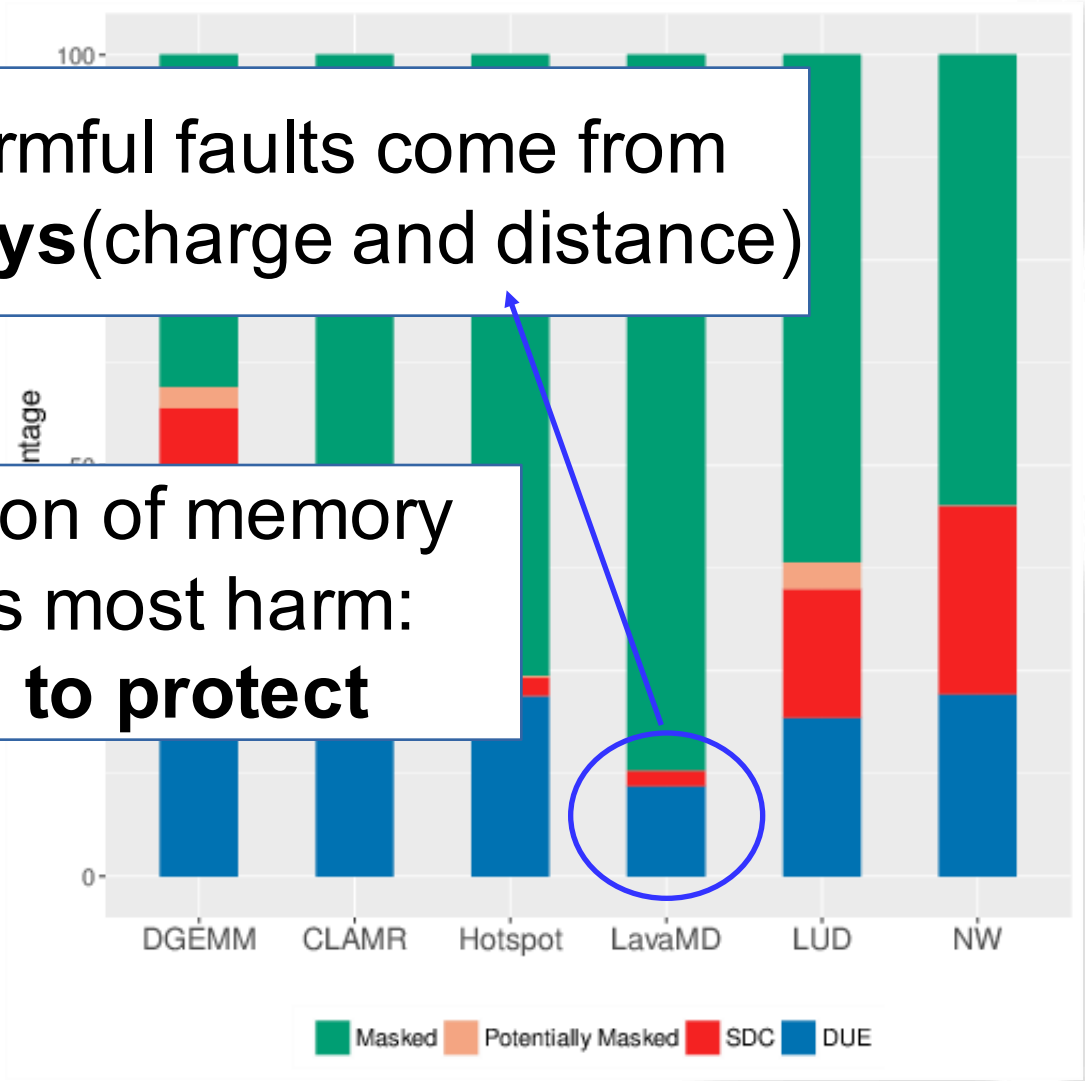
Most harmful faults come from **constants and control variables**

Small portion of memory causes most harm: **Easy to protect**

Results - LavaMD

Most harmful faults come from **Input arrays**(charge and distance)

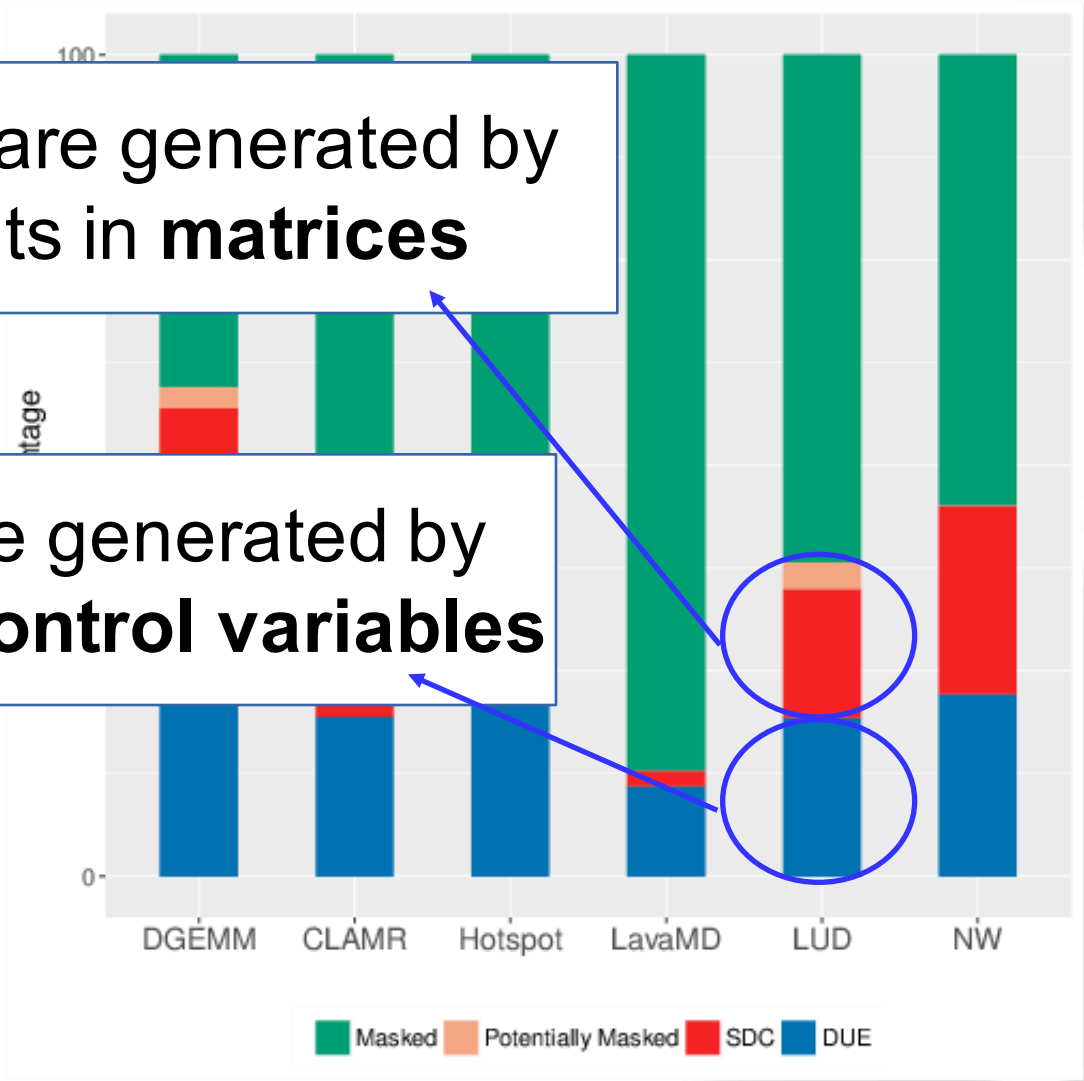
Big portion of memory causes most harm: **Hard to protect**



Results - LUD

SDCs are generated by faults in **matrices**

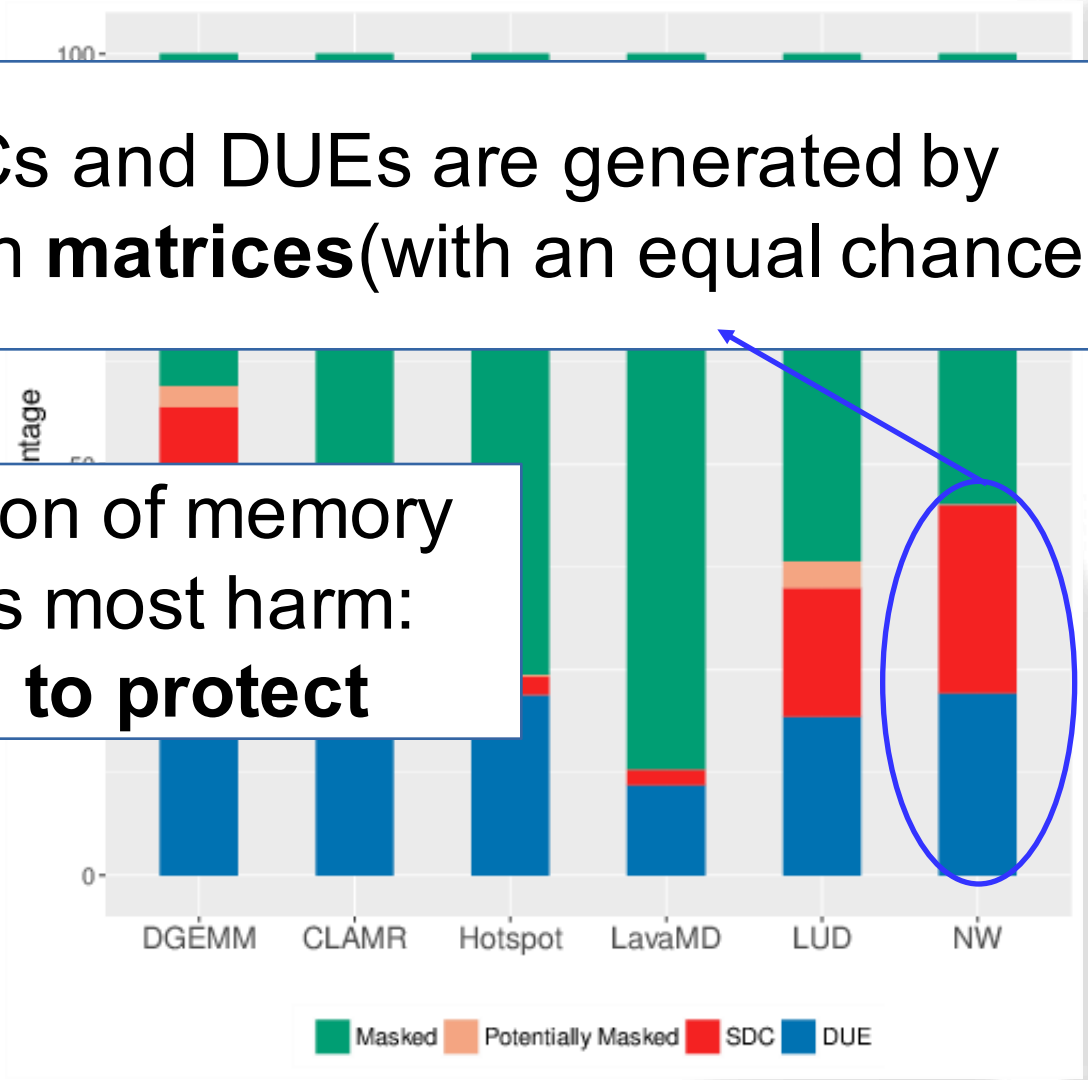
DUEs are generated by faults in **control variables**



Results - NW

SDCs and DUEs are generated by faults in **matrices**(with an equal chance)

Big portion of memory causes most harm: **Hard to protect**



Results

CAROL-FI insights:

- Selective hardening will be effective for DGEMM and Hotspot (small portion of memory causes harm)
- Selective hardening may not be effective for LavaMD and NW (big portion of memory causes harm)
- CLAMR: specific operations should be hardened (Sort and K-D Tree)

What's The Plan?

Exascale = 55x Titan. Can we afford a 55x error rate? Probably not.

What's The Plan?

Exascale = 55x Titan. Can we afford a 55x error rate? Probably not.

- We can show how SDC appears at the output, to ease detection

- Understand SDC criticality. Not all errors significantly affect output: there are "acceptable" SDC

What's The Plan?

Exascale = 55x Titan. Can we afford a 55x error rate? Probably not.

- We can show how SDC appears at the output, to ease detection

- Understand SDC criticality. Not all errors significantly affect output: there are "acceptable" SDC

- Fault-injection to better understand error propagation

SASSIFI: NVIDIA architectural-level fault-injector
CAROL-FI: UFRGS fault-injector for Xeon Phi and X86

What's The Plan?

Exascale = 55x Titan. Can we afford a 55x error rate? Probably not.

- We can show how SDC appears at the output, to ease detection

- Understand SDC criticality. Not all errors significantly affect output: there are "acceptable" SDC

- Fault-injection to better understand error propagation

SASSIFI: NVIDIA architectural-level fault-injector
CAROL-FI: UFRGS fault-injector for Xeon Phi and X86

- Propose selective-hardening solutions (duplicate only what matters, what REALLY matters)

Acknowledgments

Caio Lunardi
Caroline Aguiar
Daniel Oliveira
Fernando Santos
Laercio Pilla
Vinicius Frattin
Philippe Navaux
Luigi Carro

Nathan DeBardeleben
Sean Blanchard
Heather Quinn
Thomas Fairbanks
Steve Wender

Timothy Tsai
Siva Hari
Steve Keckler

Chris Frost

Matteo Sonza Reorda
Luca Sterpone

David Kaeli
NUCAR group