A RESILIENT INTERFACE FOR APPROXIMATE DATA ACCESS

João Fabrício Filho[1,2]

Isaías B. Felzmann[1]

Rodolfo Azevedo [1]

Lucas F. Wanner [1]

[1] University of Campinas

[2] Federal University of Technology - Paraná

isaias.felzmann@ic.unicamp.br

Trading power

- Problem: We want to save power!
- Solution 1: Make hardware smaller…
  - Physics says “not anymore”.
- Solution 2: Trade power for Performance…
  - Large portions of hardware kept off - *Dark Silicon*
- Solution 3: Trade power for Quality…
  - Not every application need a perfect result
  - **Approximate Computing**

Memory approximation

- SRAM - Voltage Scaling
  - Reduces noise margins on read/write operations
  - Exposes data to errors
  - Error rate increases for lower voltage levels
    - Exponentially!

(Wang & Calhoun, TVLSI'2011)

- Alternatives:
  - DRAM Refresh rate
  - Precision scaling

Classifying Execution Crashes

Data Crash

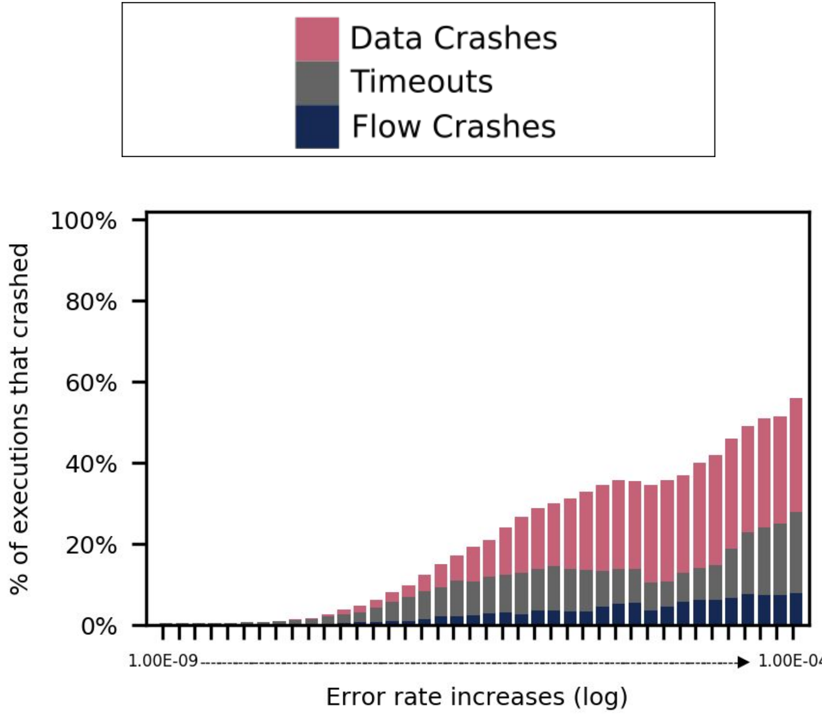
- Illegal memory access while fetching data

Control Crash

- Illegal memory access while fetching instruction

Timeout

- Application fails to converge



AxRAM: Preventing crashes

- Lightweight implementation
  - Avoid checkpoint & rollback
  - Avoid recovery software routines
- Find upper bounds for error rate
  - And lower bounds for energy
- Minimal user intervention for control
  - Less code to maintain
  - No expert knowledge required

Correcting Data Crashes

Maximum allowed address = 31

00011111

Original value

Value on memory exposed to errors

00001000

noise

01001000

Bit flip on higher end position

Pointer to position 8

Pointer to position 72

Truncate the value discarding MSBs

00001000

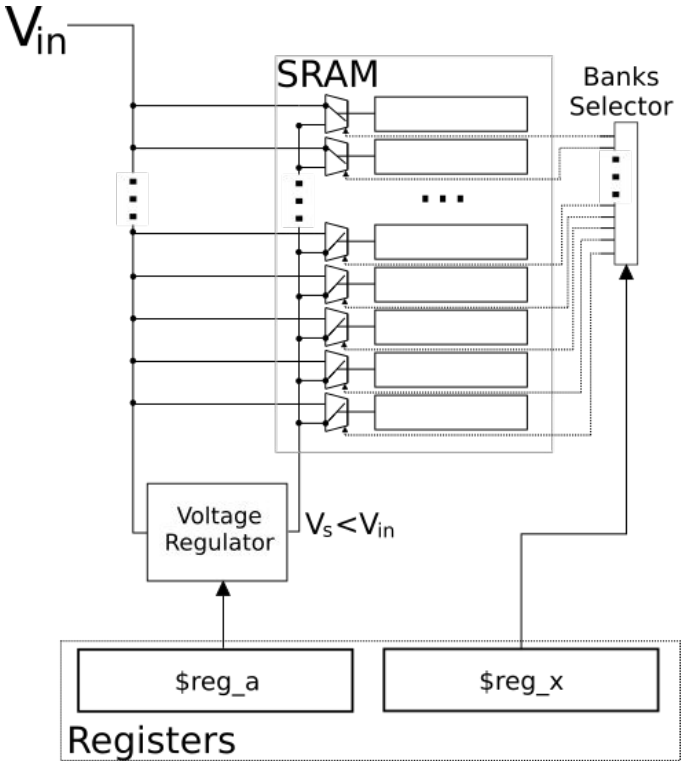
Pointer to position 8

Preventing Control Crashes: Stack protection

- Stores some control pointers
  - E.g. function return addresses
- Also stores other critical data
  - Local variables, loop control indexes
- Stack addresses are identifiable without user intervention

How to protect?

- Architectural model
  - Voltage selector for each memory bank
- Voltage regulator to control approximate state
- Memory-mapped control registers



Experiments

Memory-bound

- 2mm
- bunzip
- bzip
- dijkstra
- floyd-warshall
- qsort

CPU-bound

- nbody
- mandelbrot
- spectralnorm

Signal processing

- jpeg
- fft
- reg_detect

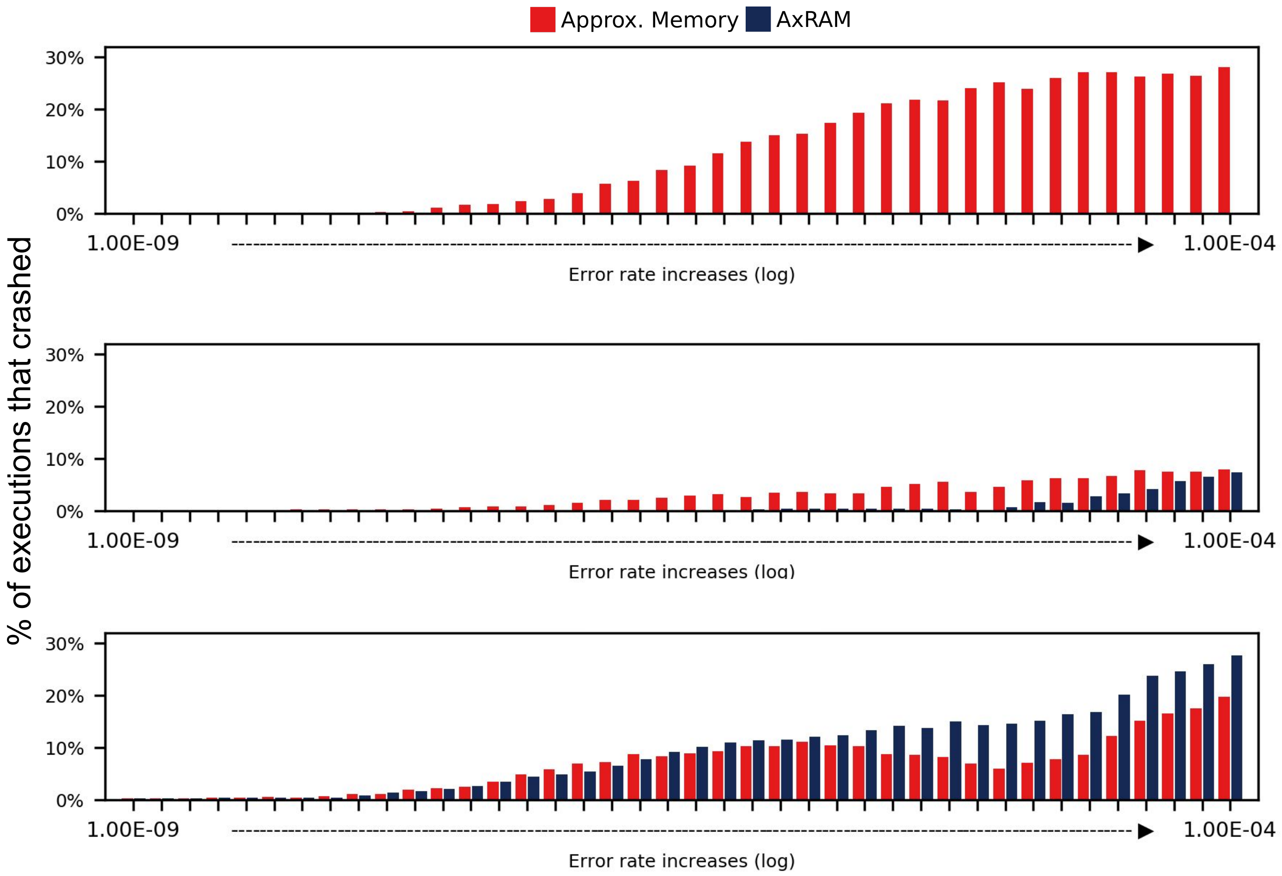
- Error rates from 10^{-9} to 10^{-4}
- Errors are probabilistic
- All results compared to unprotected scenario

Execution Crashes

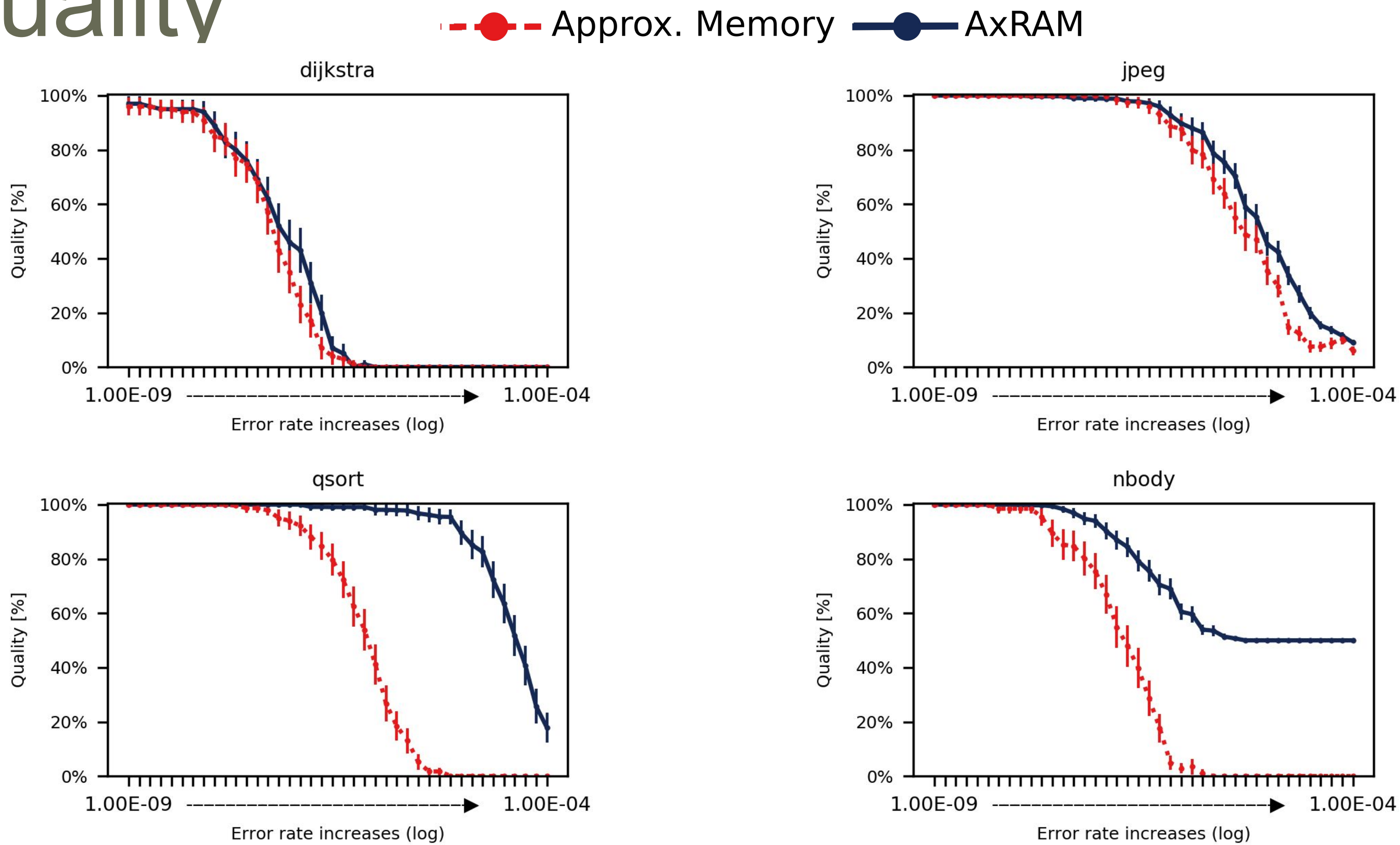
Data crashes

Flow crashes

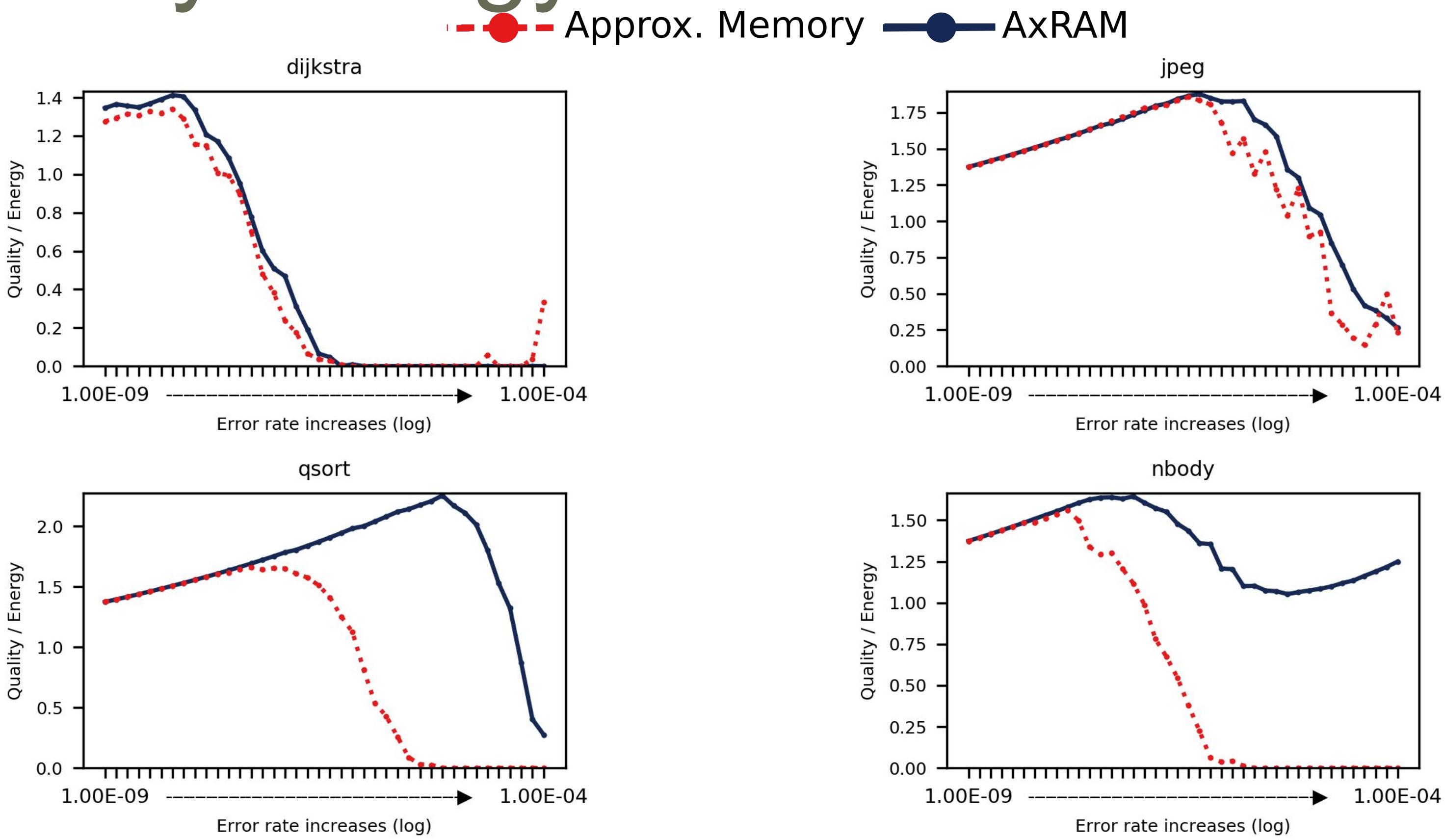
Timeouts



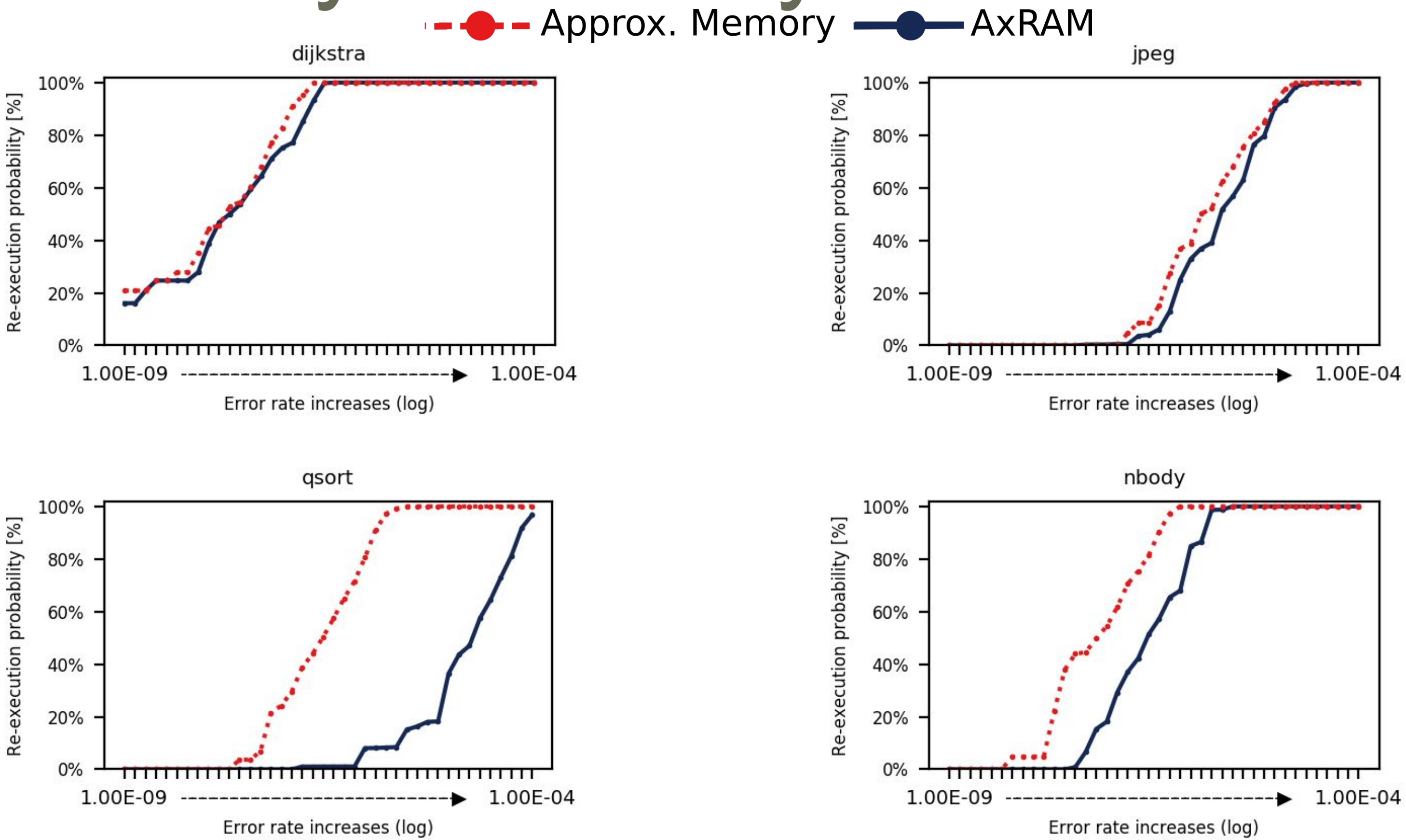
Quality


Approx. Memory
AxRAM
dijkstra
jpeg
qsort
nbody
100%
80%
60%
40%
20%
0%
Quality [%]
1.00E-09
1.00E-04
Error rate increases (log)

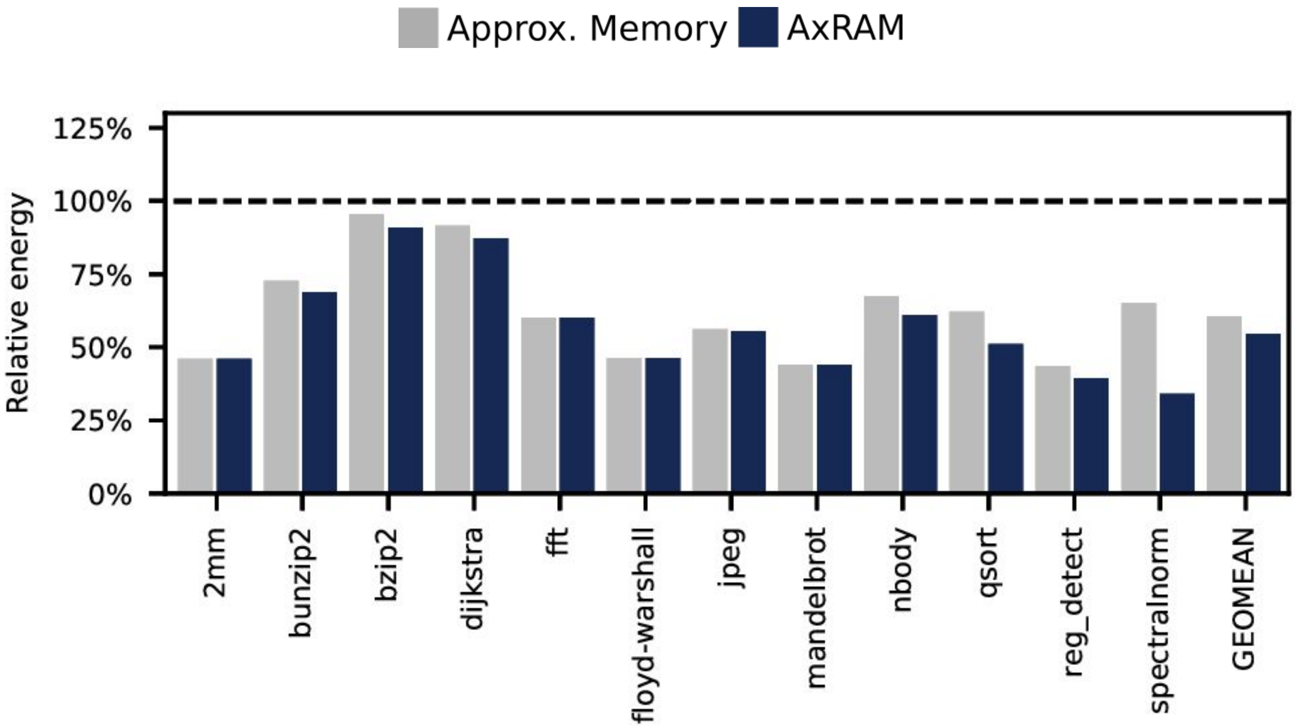
Quality/Energy



Probability of Quality < 80%



Relative Energy, Quality > 80%



Final Remarks

- Most quality depreciation results from crashes
- Applications tolerate higher error rates when crashes are mitigated
- AxRAM access protection prevents application crashes
  - Higher energy savings
  - Even higher if compared to traditional SW techniques

Thank You!

varchc.github.io/sbesc/

isaias.felzmann@ic.unicamp.br