

A RESILIENT INTERFACE FOR APPROXIMATE DATA ACCESS

João Fabrício Filho^{1,2}

Isaías B. Felzmann¹

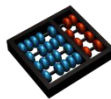
Rodolfo Azevedo¹

Lucas F. Wanner¹

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



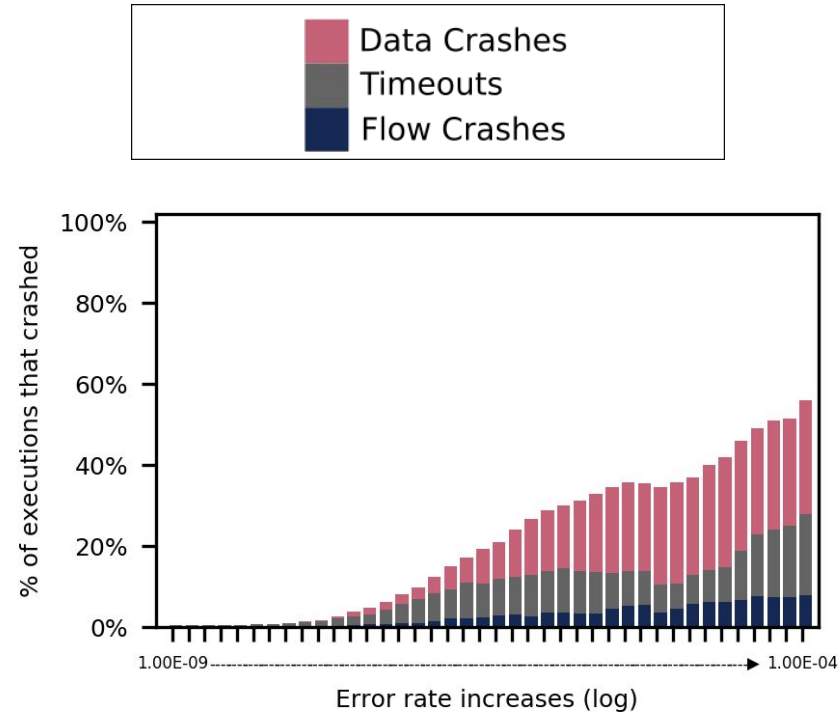
Control Crash

- Illegal memory access while fetching instruction



Timeout

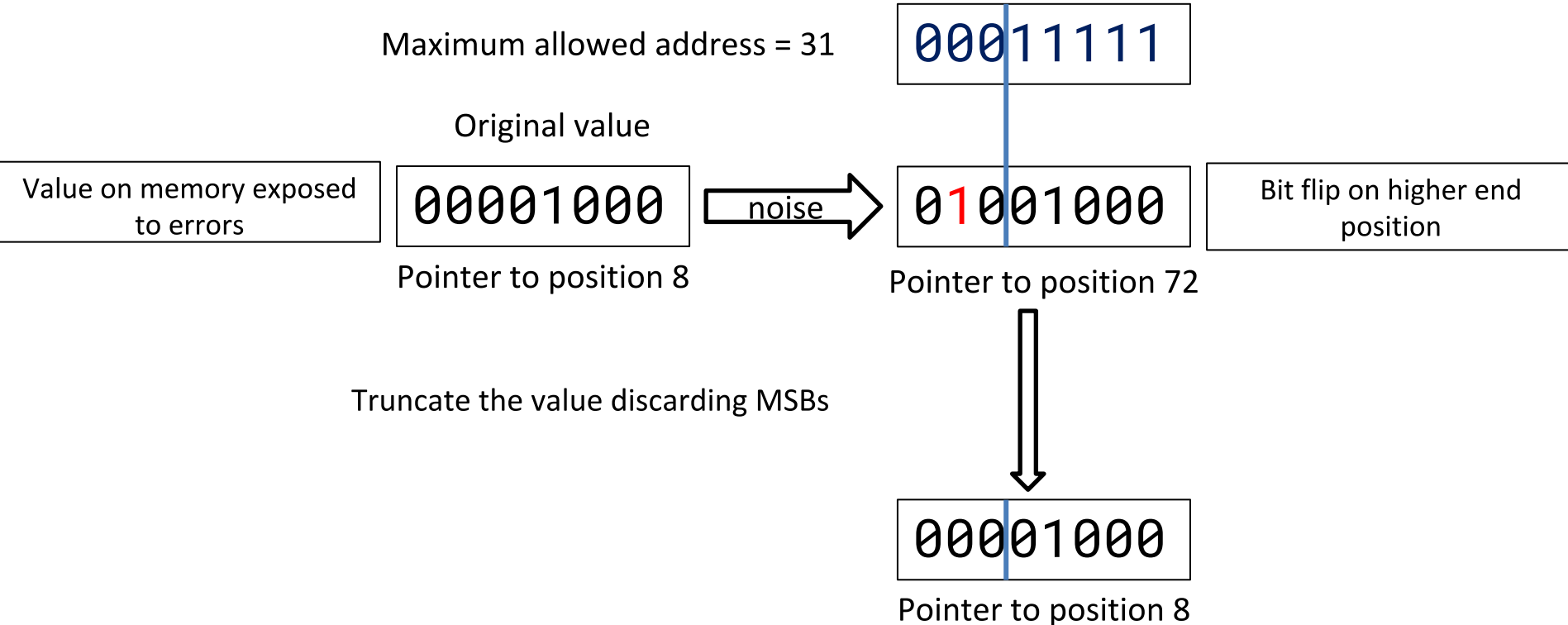
- Application fails to converge



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



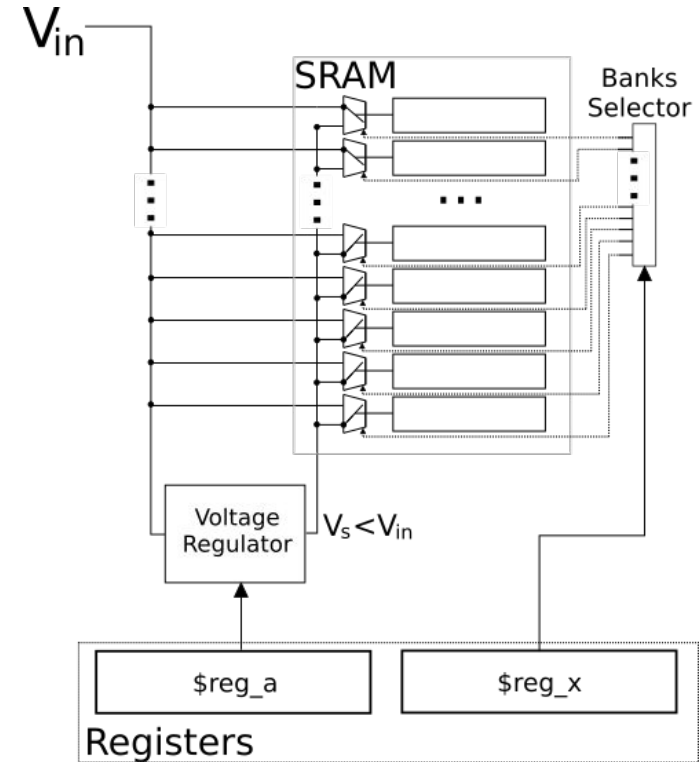
Preventing Control Crashes:

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

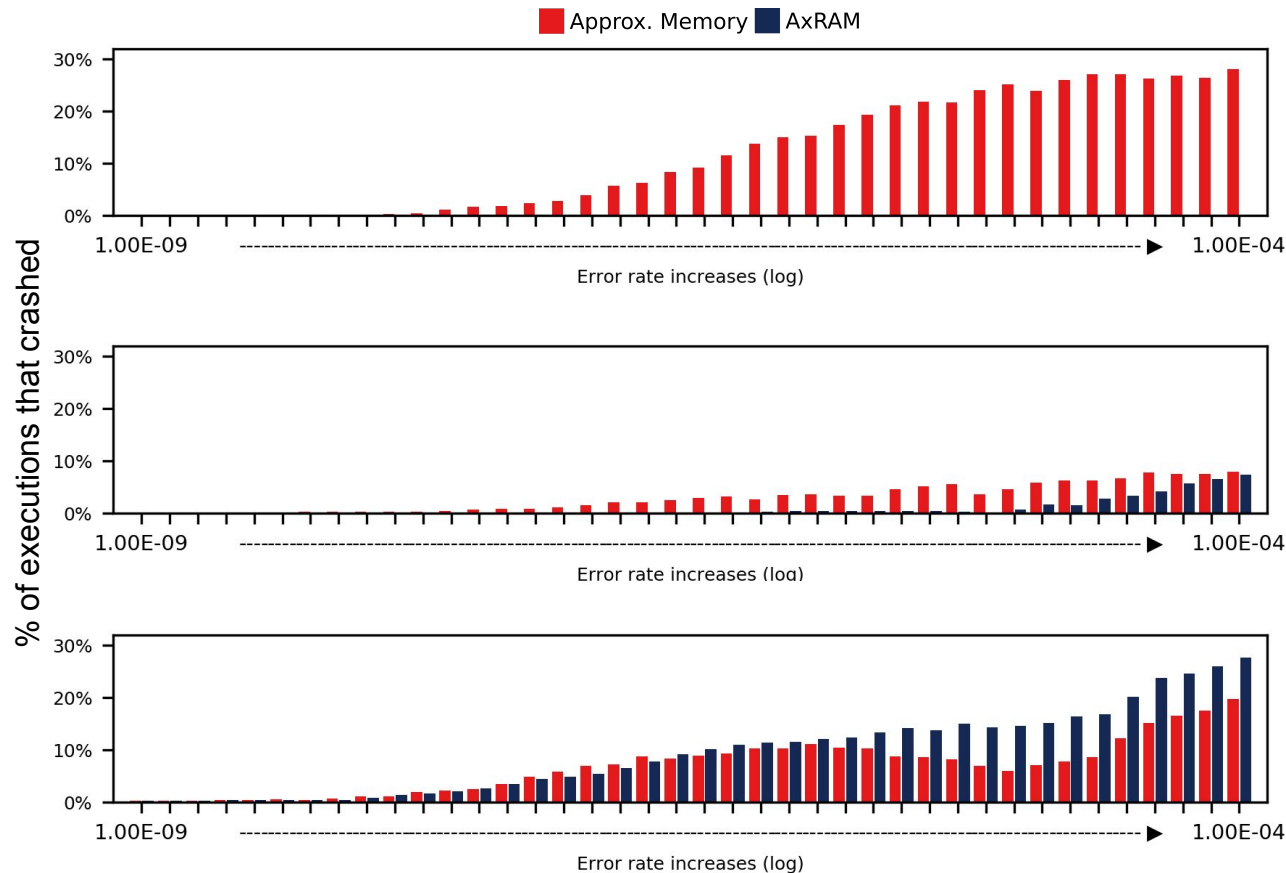
- Error rates from 10^{-9} to 10^{-4}
- Errors are probabilistic
- All results compared to unprotected scenario

Execution Crashes

Data crashes

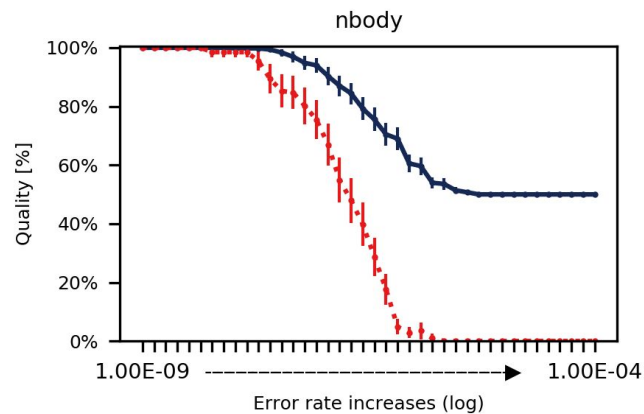
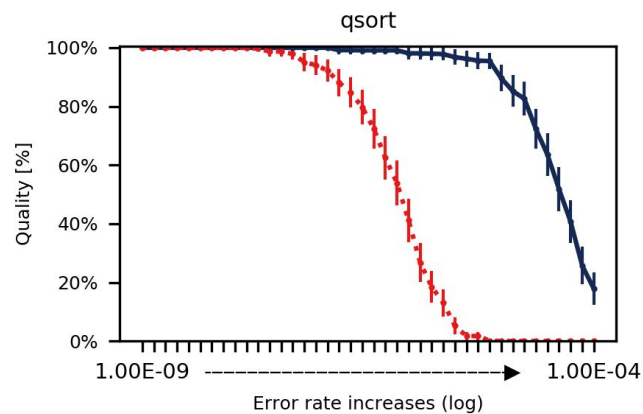
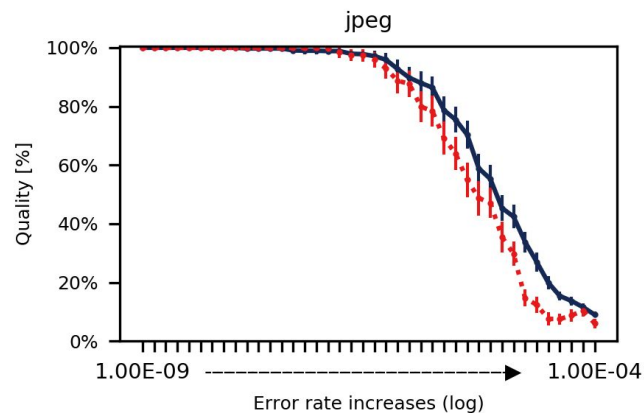
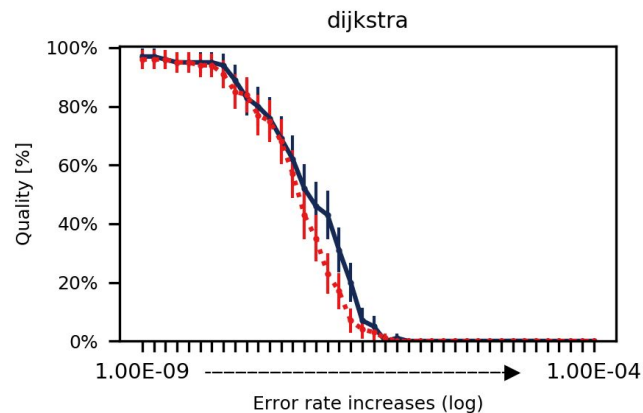
Flow crashes

Timeouts



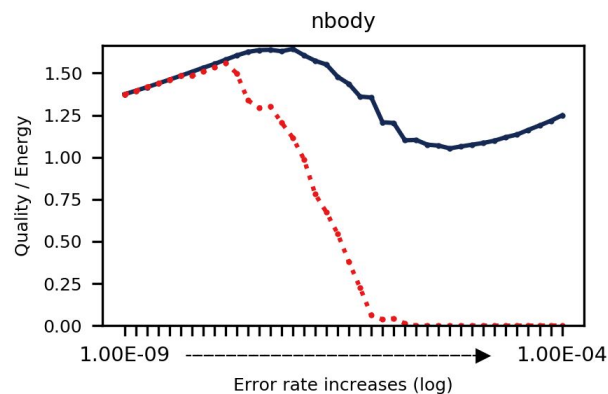
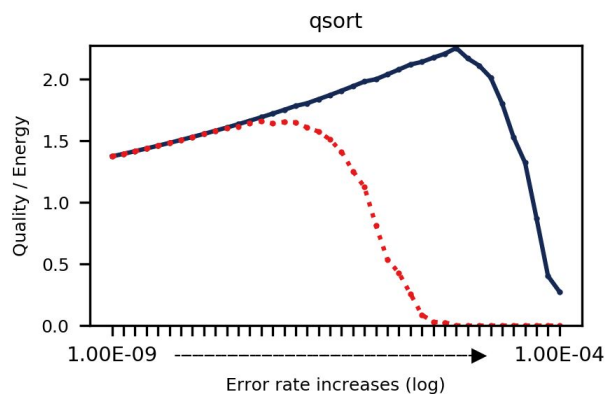
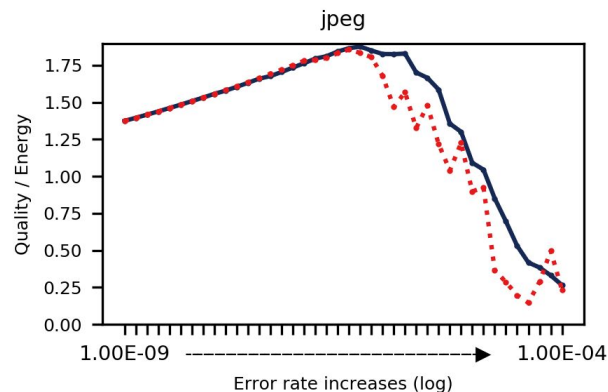
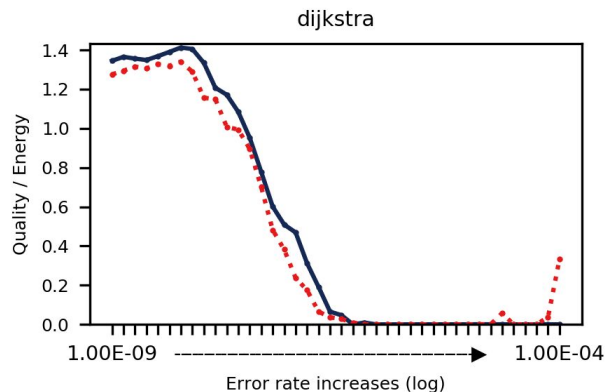
Quality

---●--- Approx. Memory —●— AxRAM



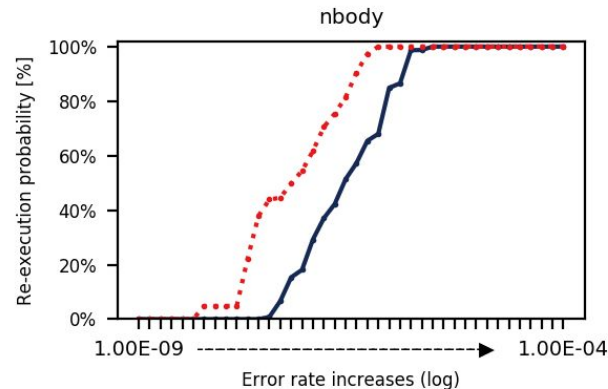
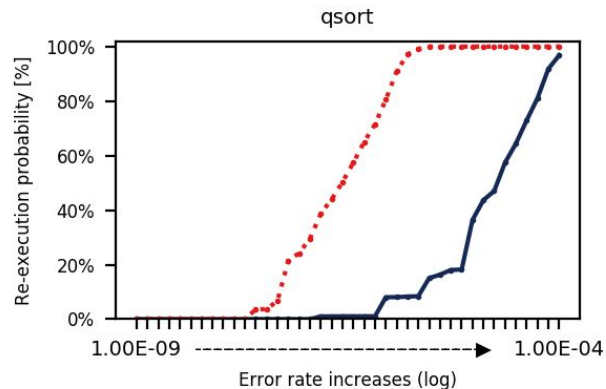
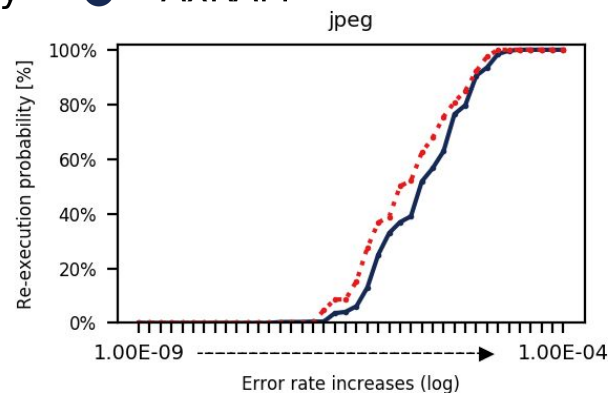
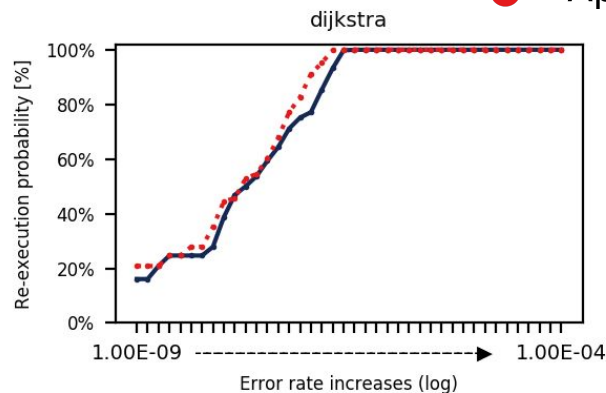
Quality/Energy

---●--- Approx. Memory —●— AxRAM

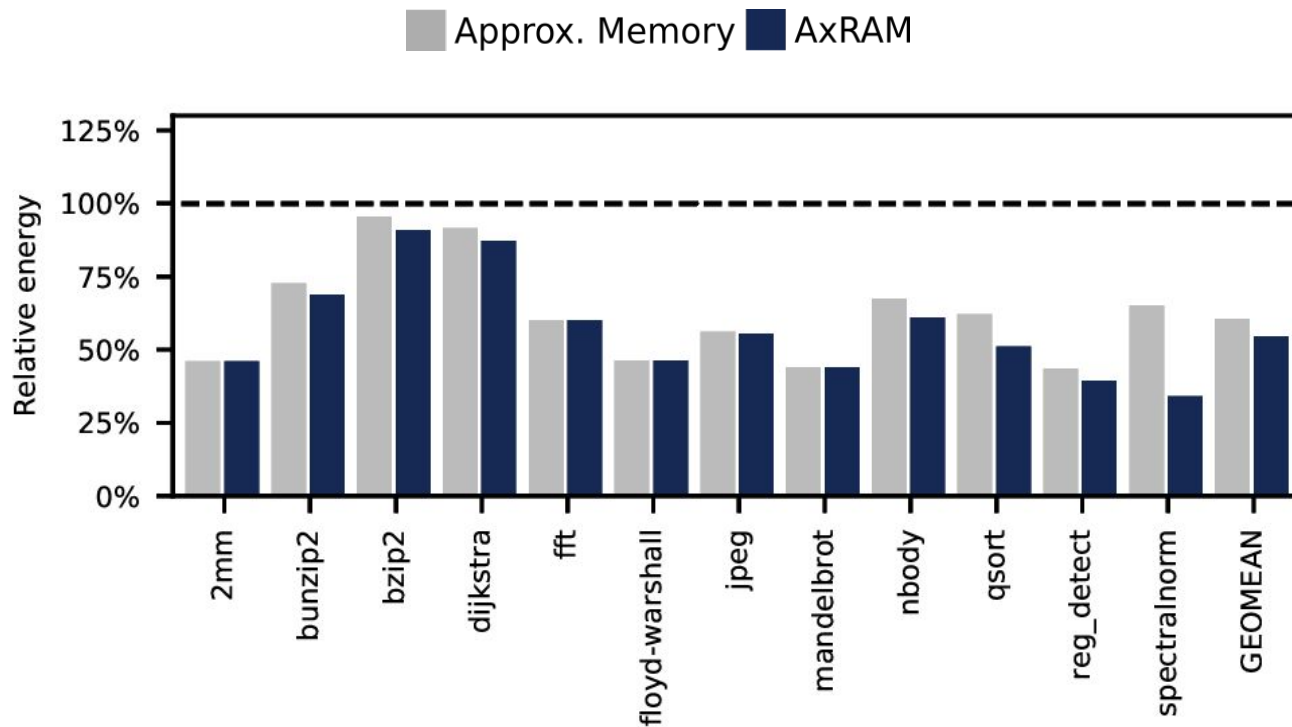


Probability of Quality < 80%

—●— Approx. Memory —●— AxRAM



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!

varhc.github.io/sbesc/

`isaias.felzmann@ic.unicamp.br`