34th International Conference on Architecture of Computing Systems (ARCS)

Transparent Resilience for Approximate DRAM

João Fabrício Filho[1 2]
Isaías Felzmann[1]
Lucas Wanner[1]

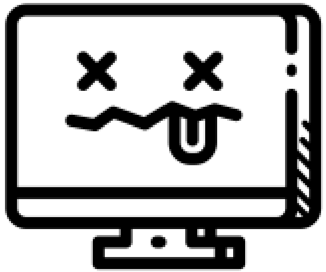
[1]University of Campinas, Campinas-SP, Brazil
[2]Federal University of Technology – Paraná, Campo Mourão-PR, Brazil

Approximate Computing

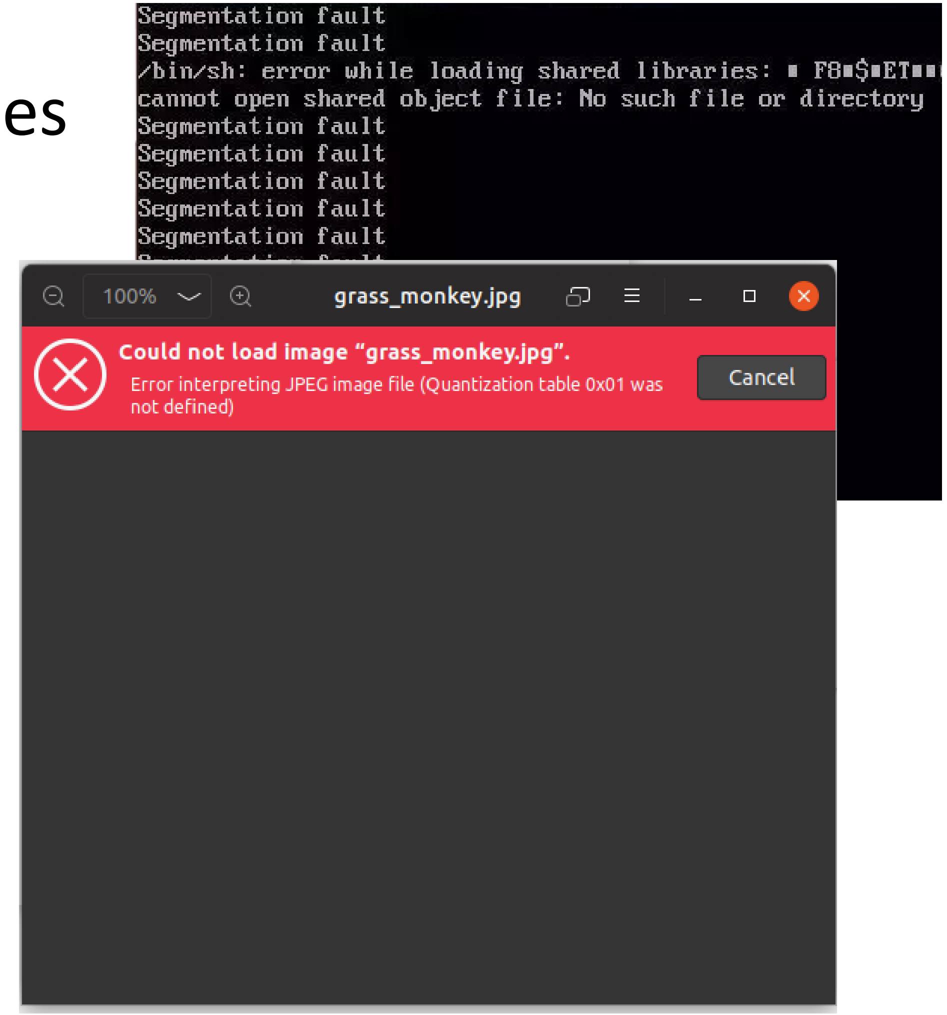
- Explores the inaccuracy tolerance of applications
- Obtain energy efficiency at the cost of errors
- Several computation can tolerate errors

```
            9.8
         ÷ 10.1
```
~~~0.970297029702970297…~~

about 0.97

Original image

error in 25% of the pixels

error in 50% of the pixels
~~~

Problem Statement

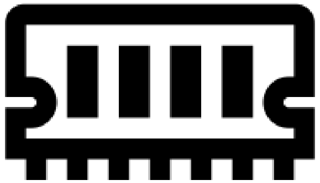


- Uncontrolled errors lead to execution crashes

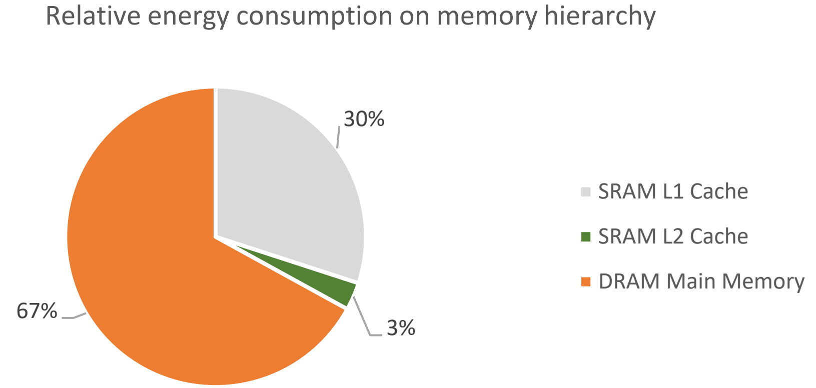


- Execution crashes cause output data loss
  - Wasting of computational efforts
  - Reduce energy savings
- All applications have critical data
- Invalid results can be generated
  - We need to recover these results

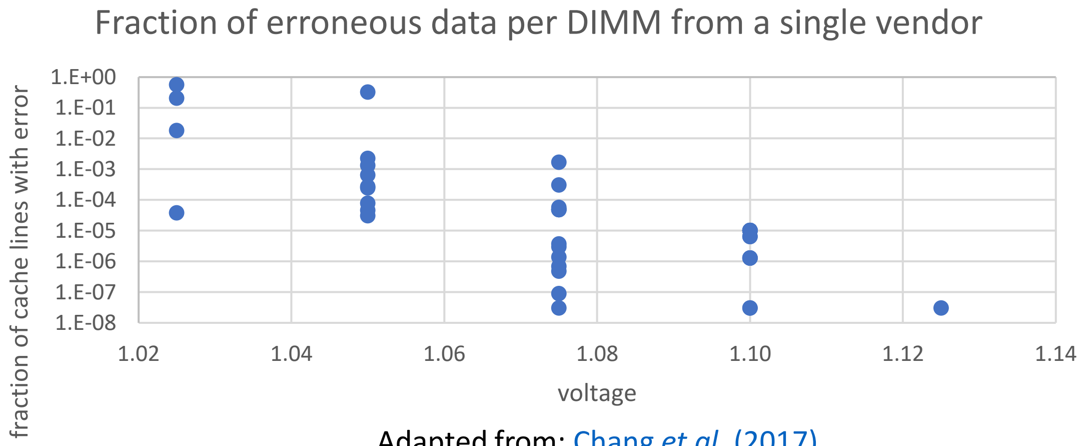
Approximate DRAM



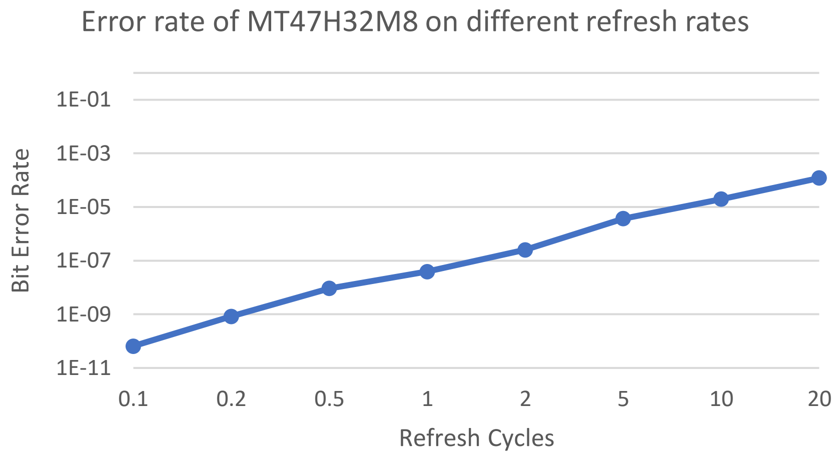
- Adjusting operational parameters
- Bitflips affect stored data



Adapted from: [Yarmand *et al.* (2019)].



Adapted from: [Chang *et al.* (2017)].



Adapted from: [Yarmand *et al.* (2019)].

Non-Transparent Interfaces

- EnerJ ([Sampson *et al.*, 2011](#))

```
@Approx class Mean {
  @Precise int length_sample;
  public float calculate(@Approx int[] nums) {
    @Approx float total = 0.0f;
    for (@Precise int i=0; i<length_sample; i++)
      total += nums[i];
    return total / length_sample;
  }
}
```

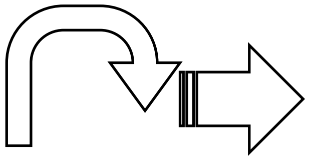
- Relax ([De Kruijf *et al.*, 2012](#))

```
int sum (int *list, int len) {
  relax (rate) {
    int sum = 0;
    for (int i=0; i<len; i++)
      total += list[i];
    recover { retry; }
  }
  return sum;
}
```

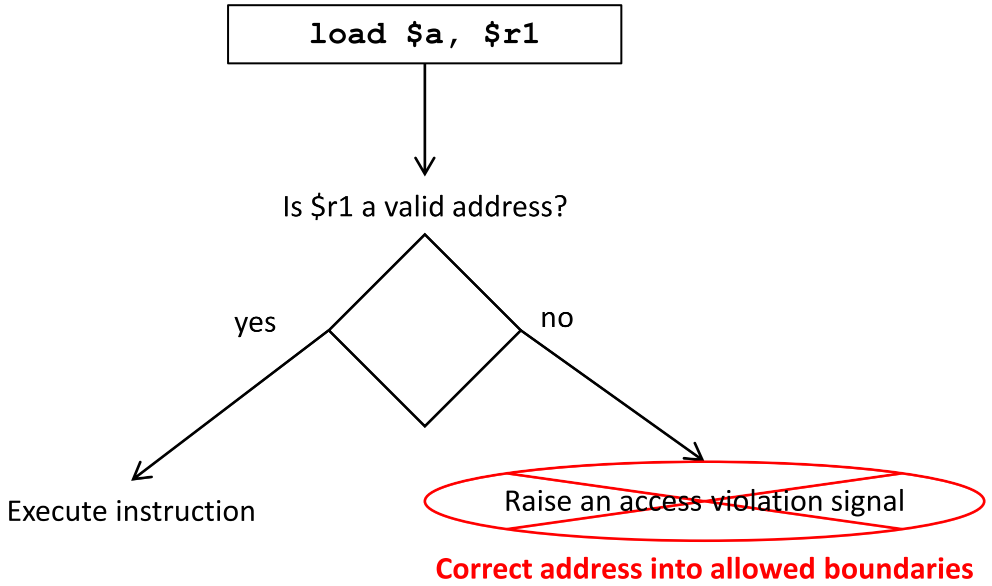
Transparent Interfaces

- Act based on general behavior of applications

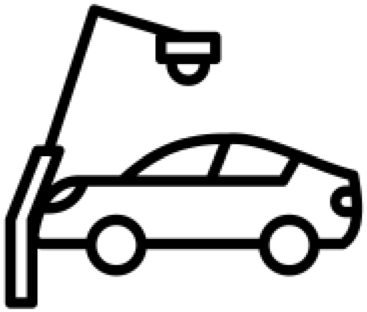
- Crash Skipping (Verdeja Herms & Li, 2019)
  - Replaces instructions that would crash execution by a “nop”



- AxRAM (Fabrício Filho *et al.*, 2020)
  - Protects common critical data regions
  - Application stack: usually small region
  - Validate memory instructions
  - Truncate memory references into allowed boundaries



Transparent Interface Design



- AxRAM mitigates data crashes
  - Caused by wrong fetched addresses

- Crash Skipping (CSi) mitigates flow crashes and execution stalling
  - Interruptions in the control flow
  - Counters of avoided crashes

- We propose a merge of these characteristics to model a single interface that avoids these three types of crashes

Transparent Resilience for Approximate DRAM

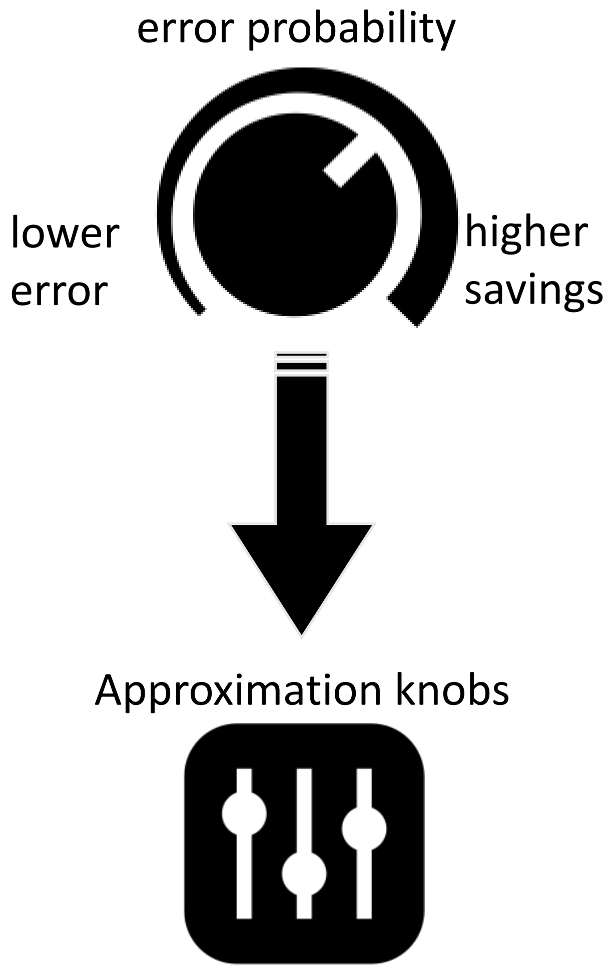
- Approximate DRAM mitigates a more energy-intensive point of the memory hierarchy
- Restarting invalid executions
  - Execution crashes are easily detected by an OS
  - Silent Data Corruptions (SDC) generate invalid output not easily detected
- Acceptance tests may detect invalid outputs generated by SDC

Transparent Re-execution

- Accurate re-execution
  - Generates a valid and accurate output
  - Nullifies the energy gains of the current instance

- Approximate re-execution
  - A new invalid output may be generated

- Proposal: approximation levels
  - Re-execution with lower error probability



Software-Level Addressing Scheme

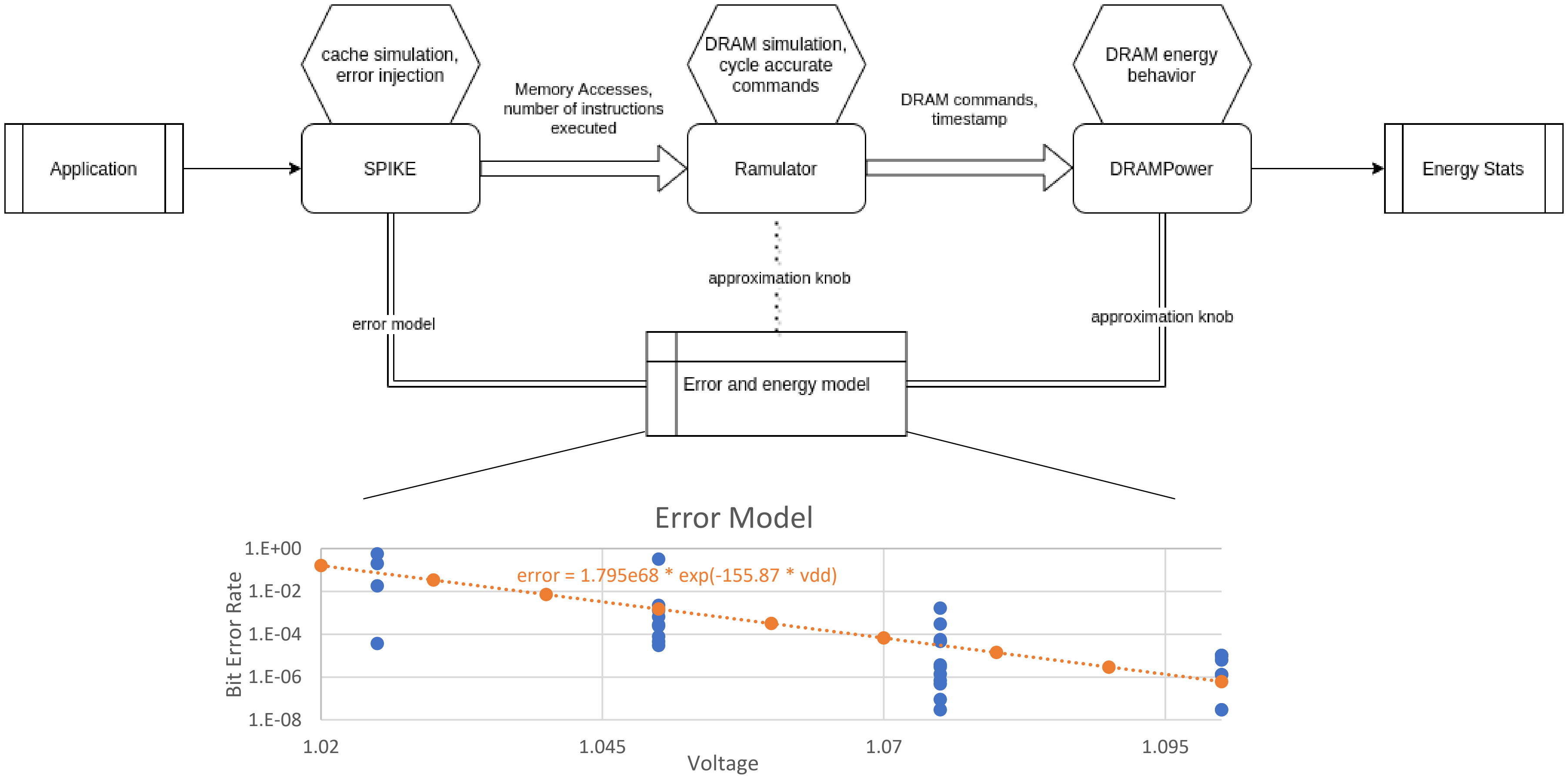
- AxRAM validates memory addresses into allowed boundaries

- Virtual addressing is not as simple as direct addressing
  - Truncating addresses does not validate the existence of a valid virtual page

- Searching for a valid Page Table Entry (PTE)
  - Starts from the higher level of the Virtual Page Number (VPN)
  - Search for a VPN with hamming distance=1 with the wrong address
  - If a correspondence is found, a new PTE is created to the same physical address

Simulation Tools and Models

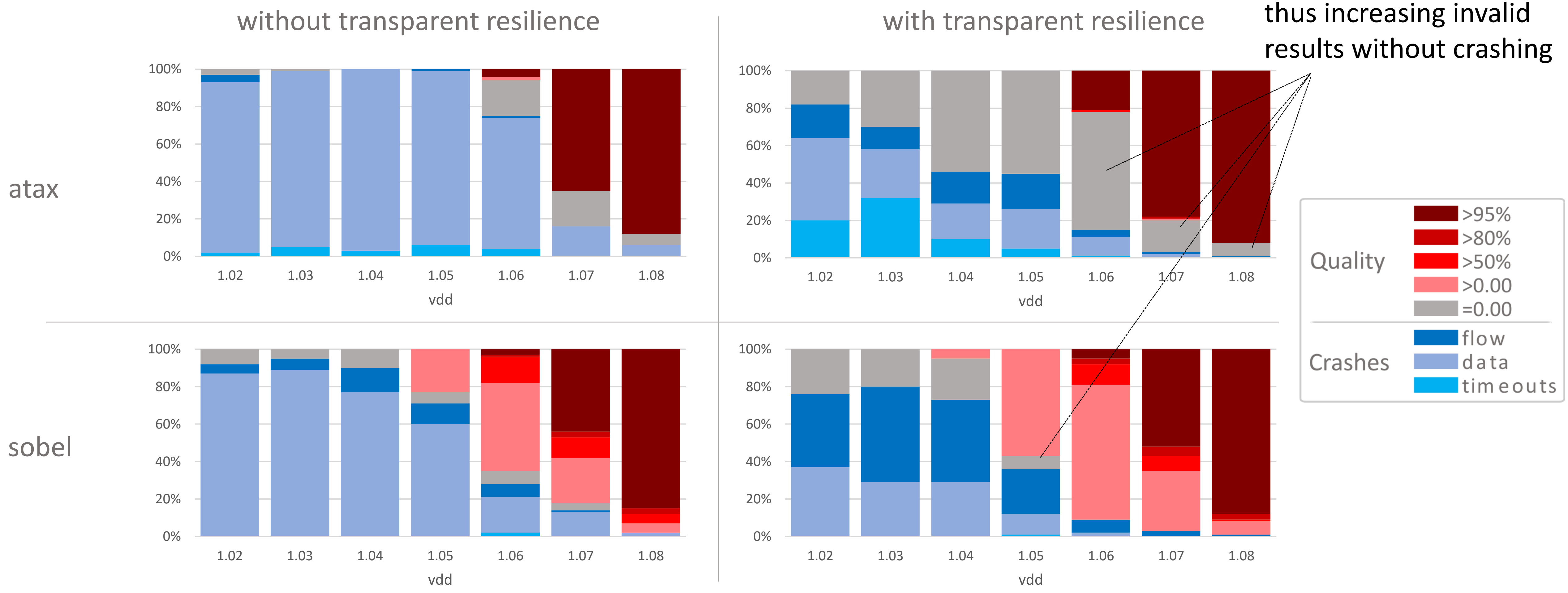
- Approximate DRAM levels:
  - Voltage ranging from 1.02 to 1.11V with 10mV steps



Based on data from [Chang *et al.* (2017)]

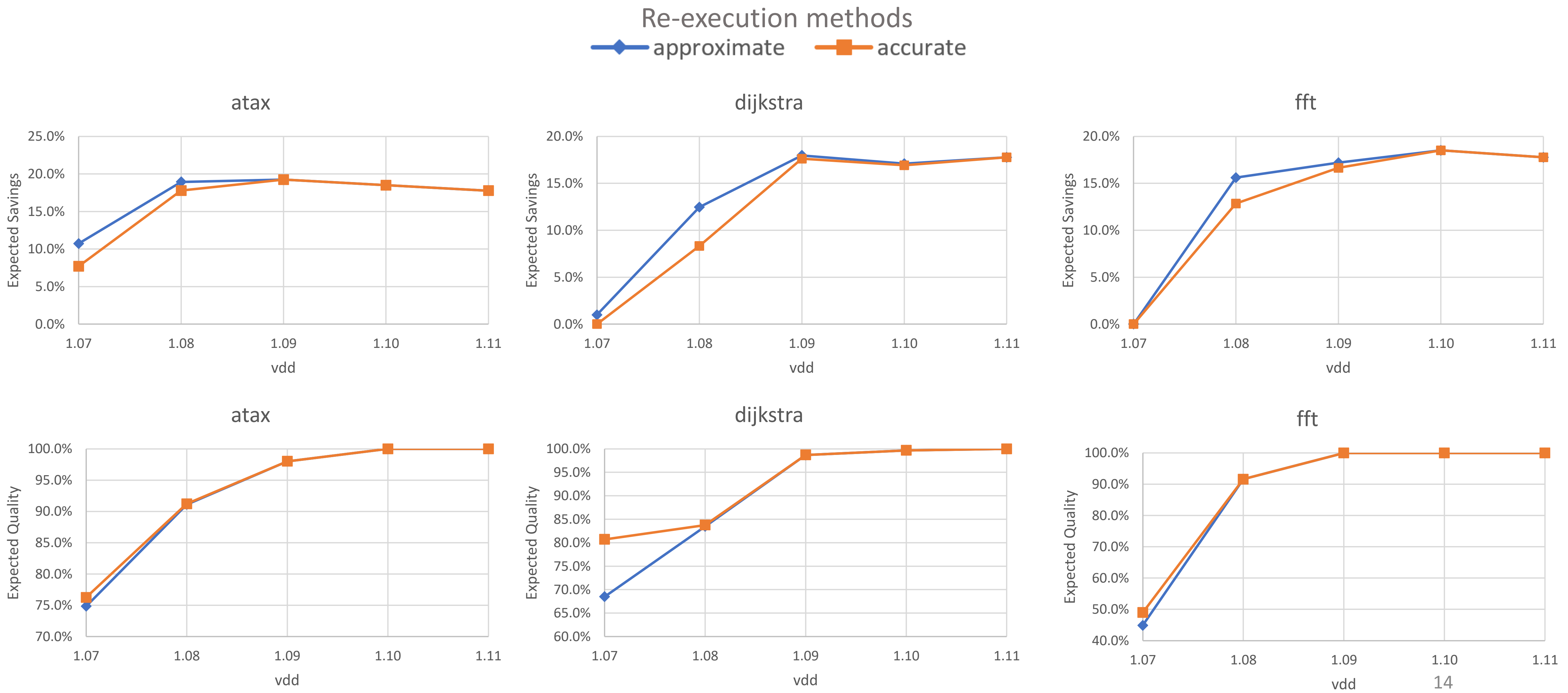
Frequencies of Quality and Crashes

resilience mechanisms tends to insist on executions with error, thus increasing invalid results without crashing



Acceptance Tests

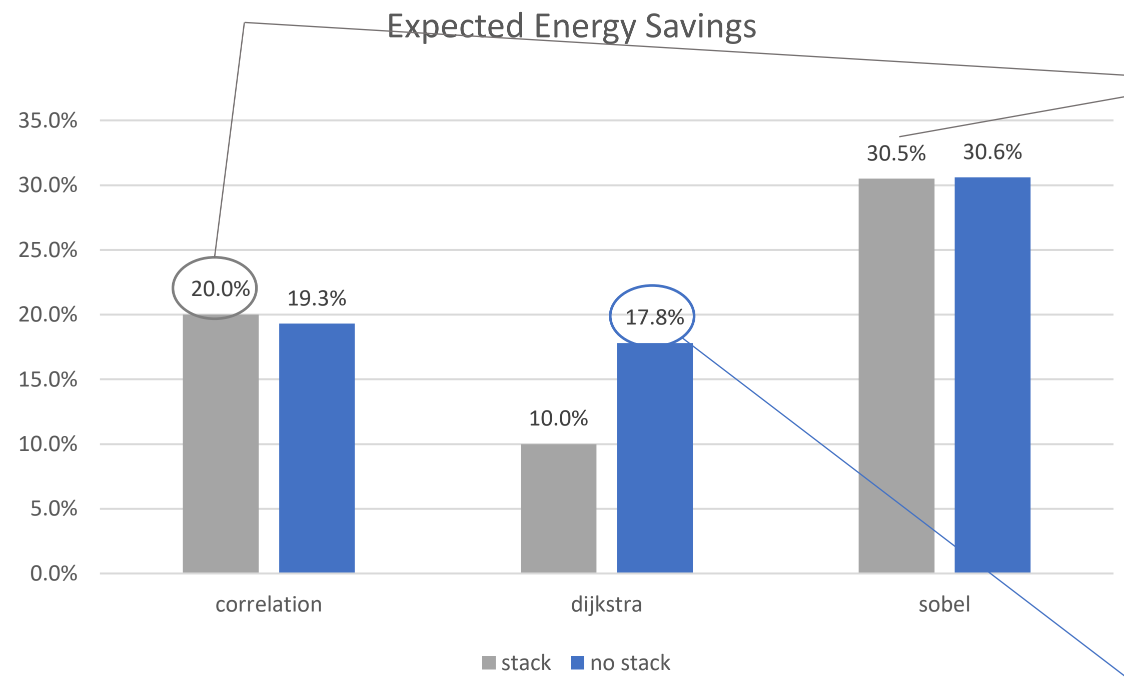
Approximate Re-execution

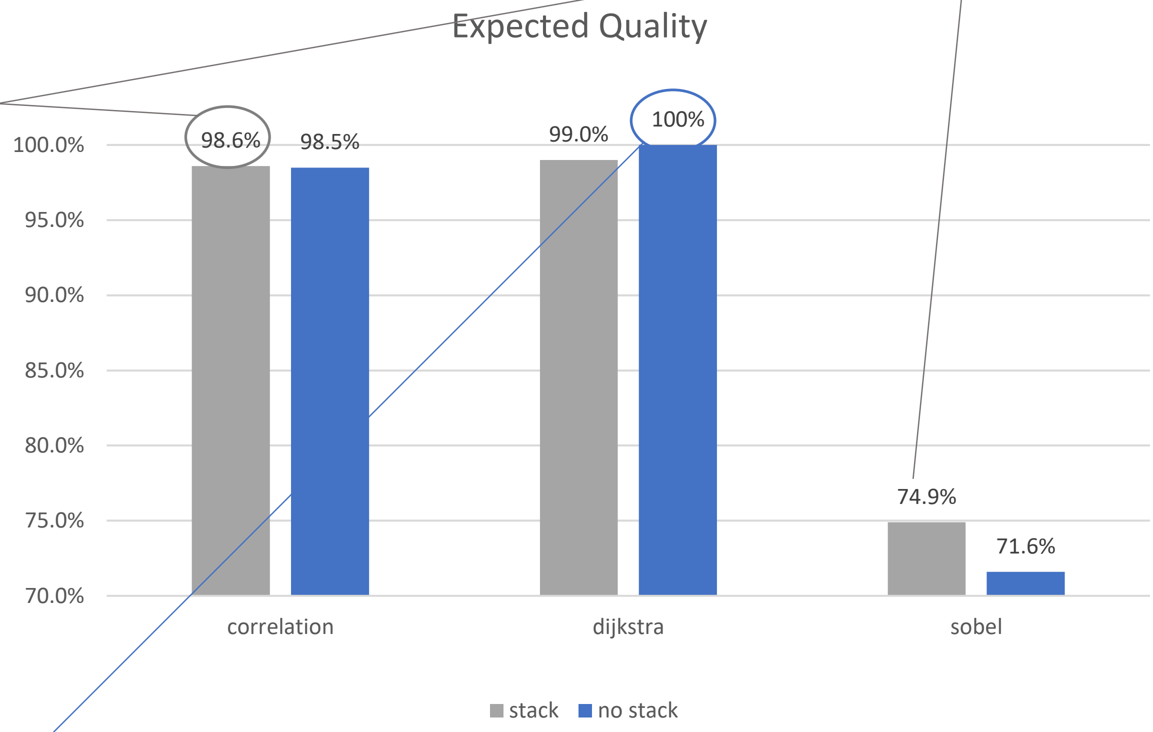


Interfaces and Stack Protection

correlation achieves higher savings and quality protecting application stack

sobel has a slight impact on energy to achieve higher quality with stack protection

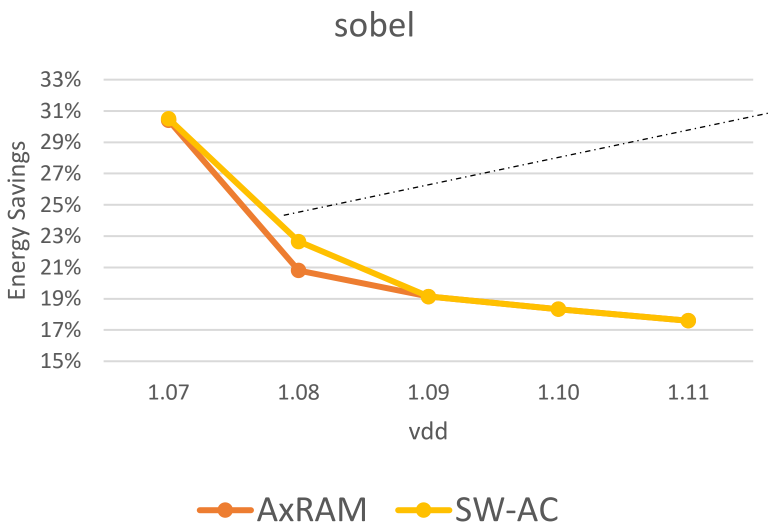


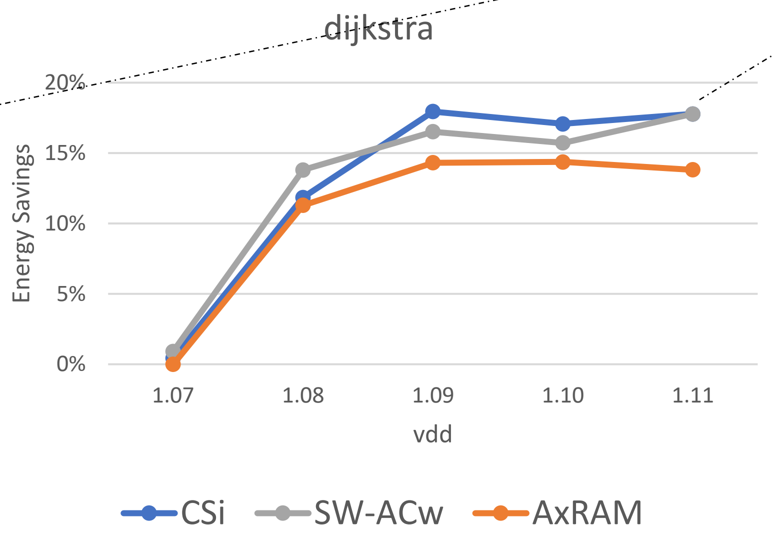


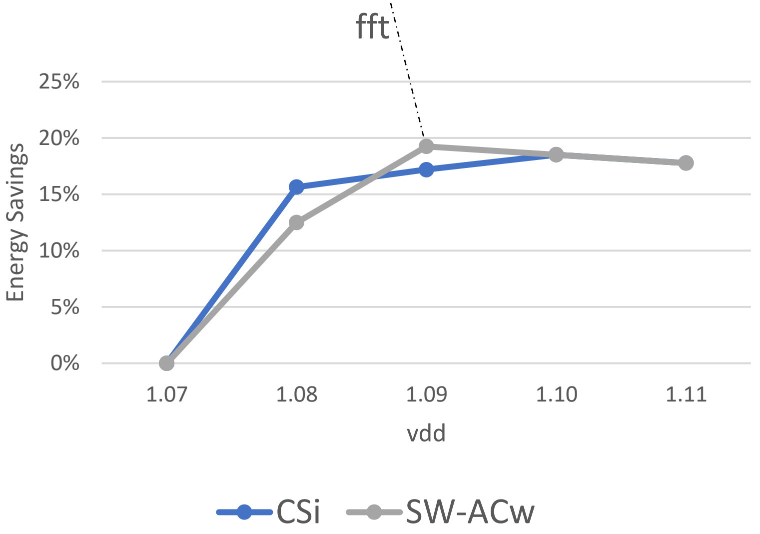
dijkstra has no benefits on protecting application stack

Comparison with AxRAM and CSi

SW-AC and SW-ACw achieve higher savings on lower and higher vdds, respectively, with advantages of AxRAM and CSi







On lower vdds, AxRAM achieves higher energy savings and SW-AC follows this trend and occasionally surpasses these benefits

On applications that have no benefits on protecting addresses and stack, SW-ACw follows the benefits of CSi due to the lower overhead

On higher vdds, CSi has the lower overhead due to less protections and SW-ACw achieves closer energy savings

Final Remarks

- Approximate DRAM
  - Less impact of error in application and higher energy savings

- Acceptance tests
  - Detects invalid results even with SDC
  - Improve detection up to 30%

- Approximate Re-execution
  - Up to 4p.p. of energy with negligible loss in quality

- Combined interface mechanisms
  - Lower overhead of CSi with lower error rate
  - Higher safeguard of AxRAM with higher error rate

- Transparent interfaces mechanisms
  - Improve execution resilience without changes in the source code
  - Increase average quality and energy savings among several approximation levels

Thanks!

Questions?

More information: http://varchc.github.io/arcs

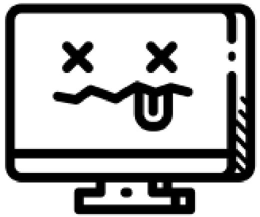
joaof@utfpr.edu.br

This work was supported by the São Paulo Research Foundation (FAPESP) grant #2018/24177-0; National Council for Scientific and Technological Development - Brazil (CNPq) grant #438445/2018-0; and Coordination for the Improvement of Higher Education Personnel - Brazil (CAPES) - Finance Code 001.

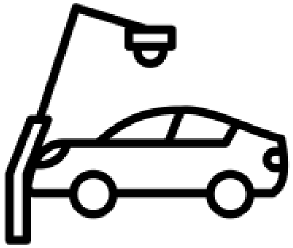
References

- Chandrasekar *et al.* (2012). DRAMPower: Open-source DRAM power & energy estimation tool. http://www.drampower.info
- Chang *et al.* (2017). Understanding Reduced-Voltage Operation in Modern DRAM Devices. *POMACS*. https://doi.org/10.1145/3084447
- Chang *et al.* (2016). Understanding latency variation in modern DRAM chips: Experimental characterization, analysis, and optimization. *SIGMETRICS*. https://doi.org/10.1145/2896377.2901453
- De Kruijf *et al.* (2010). Relax: An architectural framework for software recovery of hardware faults. *ISCA*. https://doi.org/10.1145/1815961.1816026
- Fabrício Filho *et al.* (2020). AxRAM: A lightweight implicit interface for approximate data access. *FGCS*, *113*, 556–570. https://doi.org/10.1016/j.future.2020.07.029
- Kim *et al.* (2016). Ramulator: A fast and extensible DRAM simulator. *IEEE CAL*, *15*(1), 45–49. https://doi.org/10.1109/LCA.2015.2414456
- Lee *et al.* (2014). Spike, a RISC-V ISA Simulator. https://github.com/riscv/riscv-isa-sim
- Sampson *et al.* (2011). EnerJ: Approximate data types for safe and general low-power computation. *PLDI*. https://doi.org/10.1145/1993498.1993518
- Verdeja Herms & Li (2019). Crash skipping: A minimal-cost framework for efficient error recovery in approximate computing environments. *GLSVLSI*. https://doi.org/10.1145/3299874.3317986
- Yarmand *et al.* (2020). DART: A Framework for Determining Approximation Levels in an Approximable Memory Hierarchy. *IEEE TVLSI*, *28*(1), 273–286. https://doi.org/10.1109/TVLSI.2019.2935832

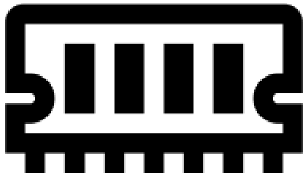
Icons from www.thenounproject.com with creative commons license



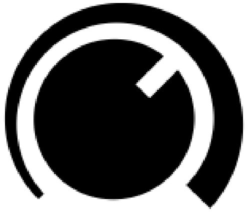
Peter van Driel, NL



Azam Ishaq, PK



muhammad rosikhan anwar, ID



Wuppdidu, DE

Azam Ishaq, PK

Anna Sophie, DE

Mavadee, TH