# *BlockHammer*

## *Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows*

**Abdullah Giray Yağlıkçı**

Minesh Patel  Jeremie S. Kim  Roknoddin Azizi

Ataberk Olgun  Lois Orosa  Hasan Hassan  Jisung Park

Konstantinos Kanellopoulos  Taha Shahroodi

Saugata Ghose*  Onur Mutlu

SAFARI

ETH zürich

* UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

# Problem and Key Idea in a Nutshell

- **Motivation**: RowHammer is a worsening DRAM reliability/security problem

- **Problem**: Mitigation mechanisms provide **limited support** for DRAM chips

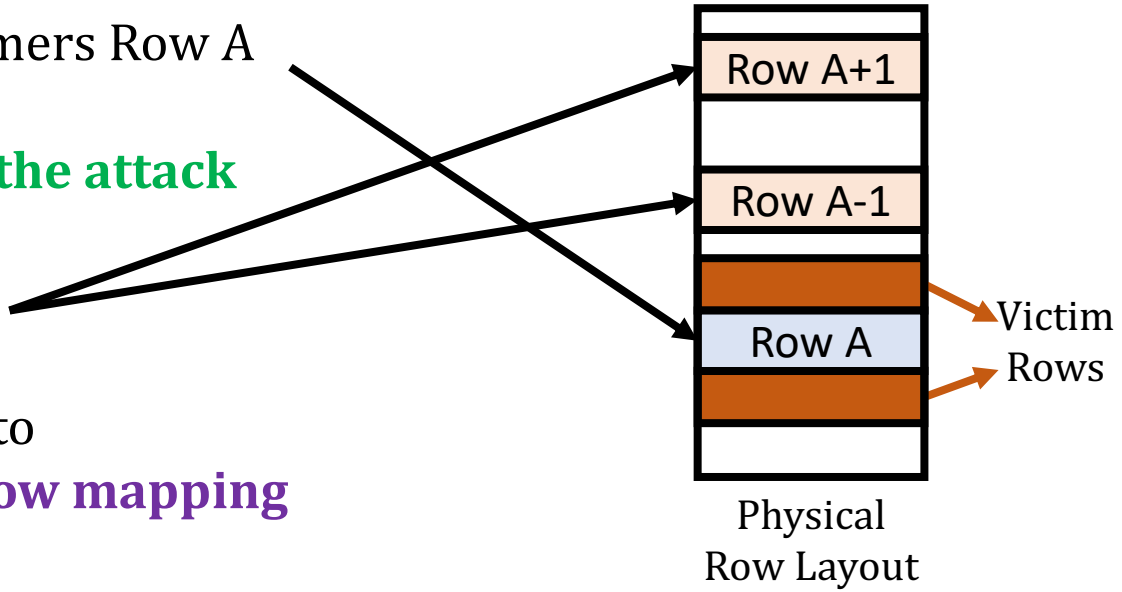
  1. **Scalability** with worsening RowHammer vulnerability
     Existing RowHammer mitigation mechanisms become **prohibitively expensive** when applied to **increasingly vulnerable** DRAM chips [J.S. Kim+, ISCA 2020]

  2. **Compatibility** with commodity DRAM chips
     Existing mechanisms **rely on proprietary information** that is *not available* for **all commodity DRAM chips**

**Key Idea: Selectively throttle memory accesses that may cause RowHammer bit flips**

# Major Problem with Past RowHammer Mitigations

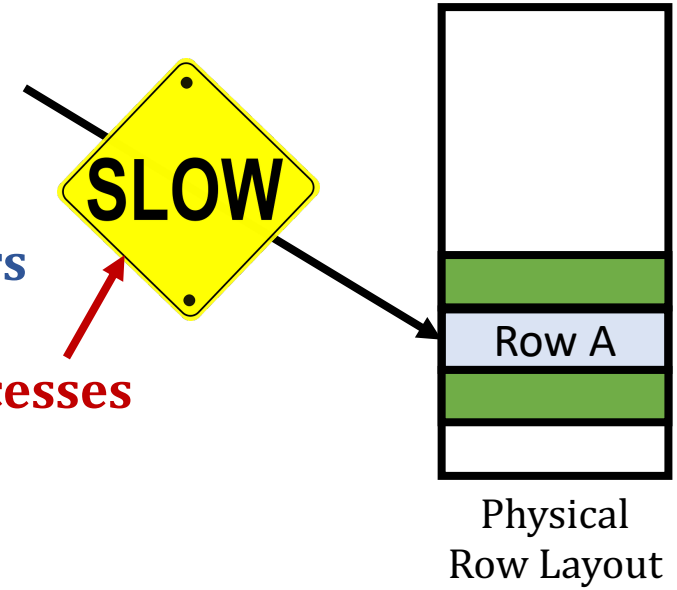

- A **RowHammer attack** hammers Row A
- Existing mechanisms **detect the attack**
- Refresh rows **A+1** and **A-1**
- Bit flips **still may occur** due to **unknown DRAM-internal row mapping**




**Existing RowHammer mitigation mechanisms need to know proprietary DRAM-internal row address mapping**

# BlockHammer: Practical Throttling-based Mechanism

- A RowHammer attack hammers Row A
- BlockHammer detects a RowHammer attack using **area-efficient Bloom filters**
- BlockHammer **selectively throttles accesses** from within **the memory controller**
- Bit flips **do not** occur
- BlockHammer can *optionally* **inform the system software** about the attack




**BlockHammer is compatible with commodity DRAM chips**
**No need for proprietary info of or modifications to DRAM chips**

# Evaluation

- **Scalability** with Worsening RowHammer Vulnerability:
  - **Competitive** with (less than 0.6% performance and energy overhead) state-of-the-art mechanisms **when there is no RowHammer attack**
  - **Superior** performance (71% speedup) and DRAM energy (32% reduction) **when a RowHammer attack is present**
- Evaluation of **14 mechanisms** representing **four mitigation approaches**
  - Comprehensive Protection
  - Compatibility with Commodity DRAM Chips
  - Scalability with RowHammer Vulnerability
  - Deterministic Protection

**BlockHammer is the only solution that addresses all four desirable properties**

Four mitigation approaches

| Approach | Mechanism | Comprehensive Protection | Compatible w/ Commodity DRAM Chips | Scaling with RowHammer Vulnerability | Deterministic Protection |
|---|---|---|---|---|---|
| Increased Refresh Rate [2, 73] | | ✓ | ✓ | ✗ | ✓ |
| Physical Isolation | CATT [14] | ✗ | ✗ | ✗ | ✓ |
| | GuardION [148] | ✗ | ✗ | ✗ | ✓ |
| | ZebRAM [78] | ✗ | ✗ | ✗ | ✓ |
| Reactive Refresh | ANVIL [5] | ✗ | ✗ | ✗ | ✓ |
| | PARA [73] | ✓ | ✗ | ✗ | ✗ |
| | PRoHIT [137] | ✓ | ✗ | ✗ | ✗ |
| | MRLoc [161] | ✓ | ✗ | ✗ | ✗ |
| | CBT [132] | ✓ | ✗ | ✗ | ✓ |
| | TWiCe [84] | ✓ | ✗ | ✗ | ✓ |
| | Graphene [113] | ✓ | ✗ | ✓ | ✓ |
| Proactive Throttling | Naive Throtl. [102] | ✓ | ✓ | ✗ | ✓ |
| | Throtl. Supp. [40] | ✓ | ✗ | ✗ | ✓ |
| **BlockHammer** | | ✓ | ✓ | ✓ | ✓ |

# More in the Paper

- Using **area-efficient Bloom filters** for RowHammer detection
- Security Proof
  - Mathematically represent **all possible** access patterns
  - **No row can be activated high-enough times** to induce bit-flips
- BlockHammer prevents **many-sided attacks**
  - TRRespass [Frigo+, S&P'20]
  - U-TRR [Hassan+, MICRO'21]
  - BlackSmith [Jattke+, S&P'22]
  - Half-Double [Kogler+, USENIX Security'22]
- System Integration
  - **BlockHammer** can detect **RowHammer attacks** with **high accuracy** and **inform system software**
  - Measures **RowHammer likelihood of each thread**
- **Hardware complexity** analysis

Full Paper

# Summary

- BlockHammer is **the first work to practically enable throttling-based RowHammer mitigation**
- BlockHammer is implemented in **the memory controller** (*no proprietary information of / no modifications* to DRAM chips)
- BlockHammer is *both* **scalable with worsening RowHammer** and **compatible with commodity DRAM chips**
- BlockHammer is **open-source** along with **six state-of-the-art mechanisms**: https://github.com/CMU-SAFARI/BlockHammer

Source

# BlockHammer

## Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows

**Abdullah Giray Yağlıkçı**

Minesh Patel  Jeremie S. Kim  Roknoddin Azizi

Ataberk Olgun  Lois Orosa  Hasan Hassan  Jisung Park

Konstantinos Kanellopoulos  Taha Shahroodi

Saugata Ghose*  Onur Mutlu

Full Talk

Full Paper

Source

ETH zürich  SAFARI  * UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

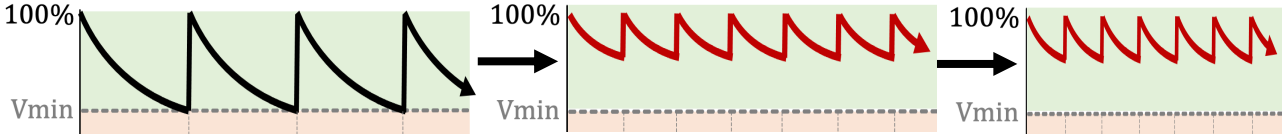
# *BlockHammer*

## *Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows*

# Backup Slides

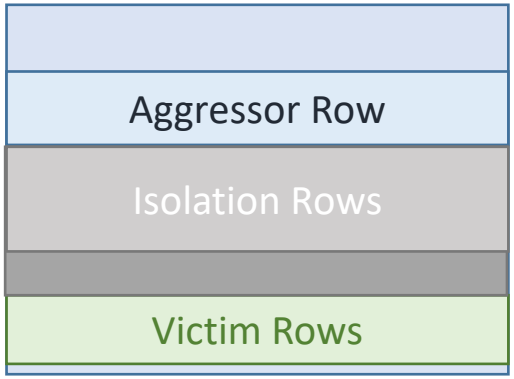
# Mitigation Approaches
## with Worsening RowHammer Vulnerability

- Increased refresh rate



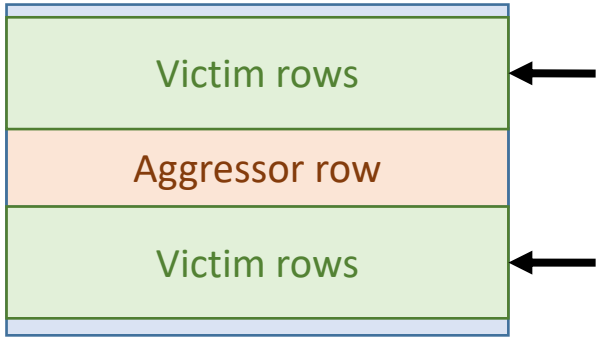

REF-to-REF time further reduces
Even fewer activations can fit

- Physical isolation




Larger distance
more isolation rows

- Reactive refresh




Refresh more frequently
Refresh more rows

Refresh more frequently
Refresh more rows

- Proactive throttling

More aggressively throttles row activations

# Scalability
## with Worsening RowHammer Vulnerability

- DRAM chips are more vulnerable to RowHammer today
- RowHammer bit-flips occur at much lower activation counts (more than an order of magnitude decrease):
  - 139.2K [Y. Kim+, ISCA 2014]
  - 9.6K [J. S. Kim+, ISCA 2020]
- RowHammer blast radius has increased by 33%:
  - 9 rows [Y. Kim+, ISCA 2014]
  - 12 rows [J. S. Kim+, ISCA 2020]
- In-DRAM mitigation mechanisms are ineffective [Frigo+, S&P 2020]

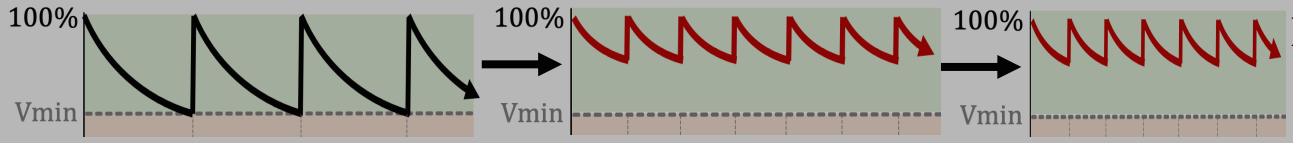
RowHammer is a **more serious problem** than ever

- Newer chips require **more aggressive** RowHammer mitigation mechanisms
- Existing mechanisms become **prohibitively expensive** [J.S. Kim+, ISCA 2020]

Defenses **should scale** with worsening RowHammer

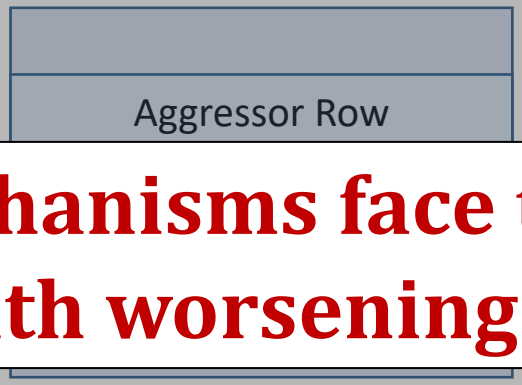
# Mitigation Approaches
## with Worsening RowHammer Vulnerability

- Increased refresh rate



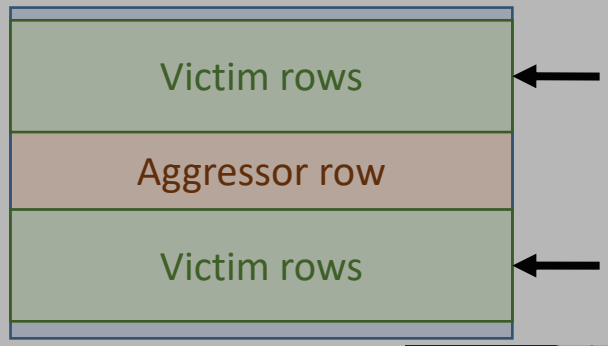

REF-to-REF time further reduces
Even fewer activations can fit

- Physical isolation




**Mitigation mechanisms face the challenge of scalability with worsening RowHammer**

- Reactive refresh



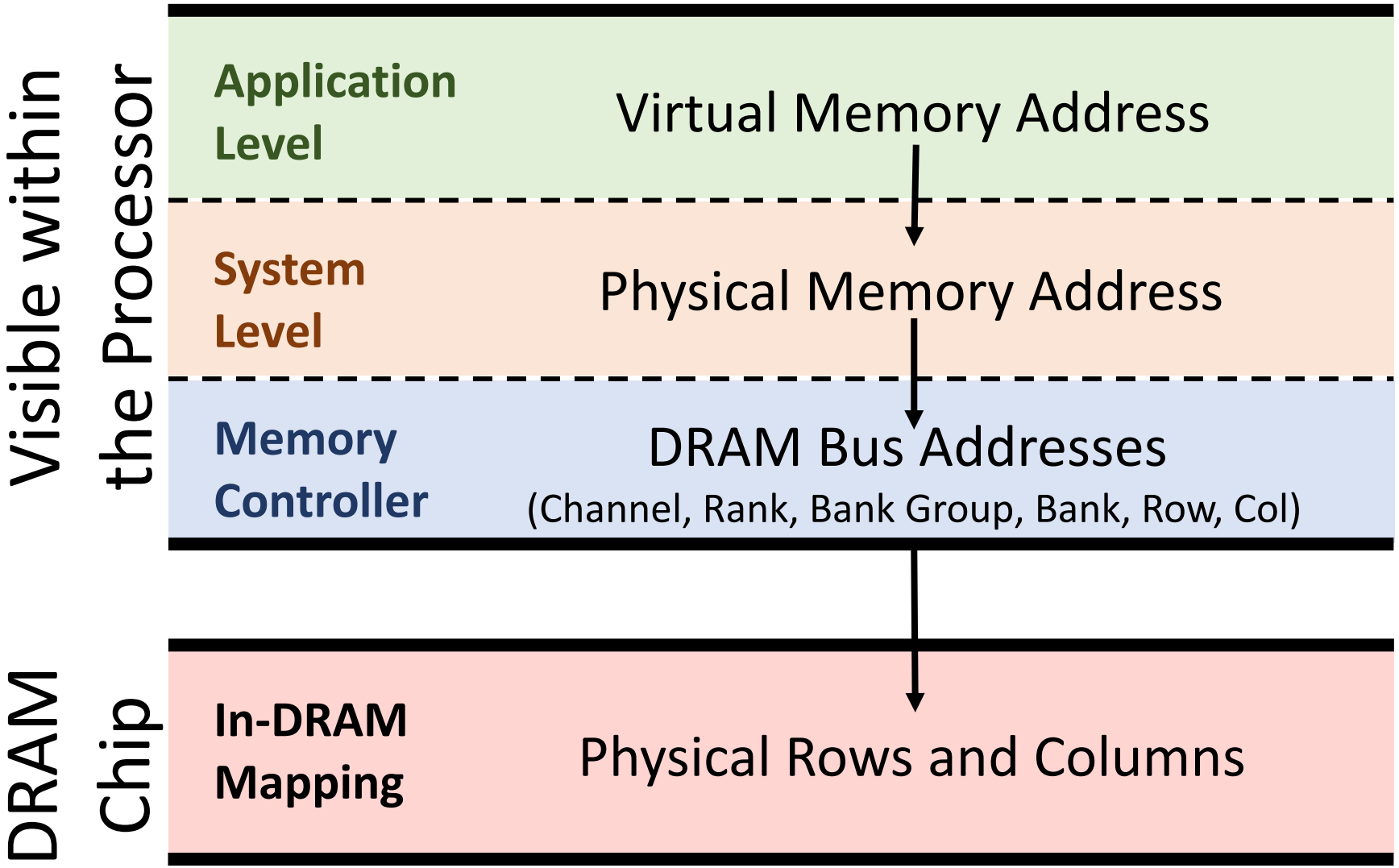

Refresh more frequently
Refresh more rows

Refresh more frequently
Refresh more rows

- Proactive throttling

More aggressively throttles row activations

# Compatibility
## with Commodity DRAM Chips



# Compatibility
## with Commodity DRAM Chips

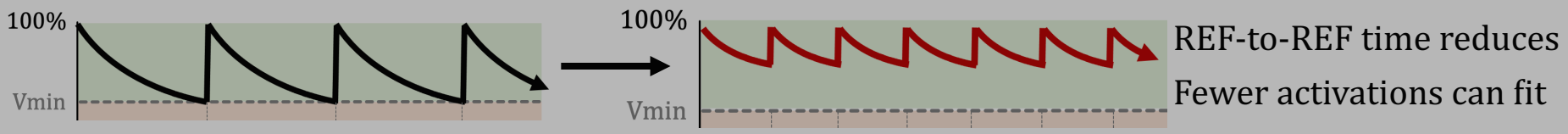
Vendors apply in-DRAM mapping for two reasons:

- **Design Optimizations:** By simplifying DRAM circuitry to provide better density, performance, and power
- **Yield Improvement:** By mapping faulty rows and columns to redundant ones
- In-DRAM mapping scheme includes insights into **chip design** and **manufacturing quality**

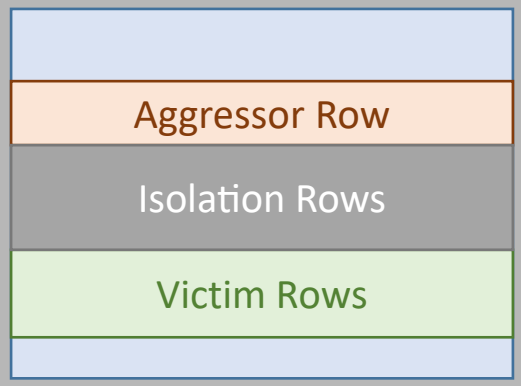
**In-DRAM mapping is proprietary information**

# RowHammer Mitigation Approaches

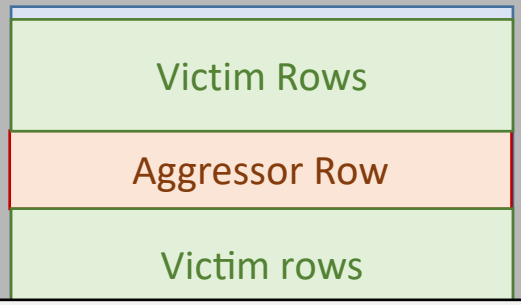
- Increased refresh rate




- Physical isolation




- Reactive refresh




Identifying ***victim*** and ***isolation*** rows requires ***proprietary*** knowledge of ***in-DRAM mapping***

# BlockHammer
## Overview of Approach

### RowBlocker

Tracks row activation rates using area-efficient Bloom filters

Blacklists rows that are activated at a high rate

Throttles activations targeting a blacklisted row

**No row can be activated at a high enough rate to induce bit-flips**

### AttackThrottler

Identifies threads that perform a RowHammer attack

Reduces memory bandwidth usage of identified threads

Greatly reduces the **performance degradation**
and **energy wastage** a RowHammer attack inflicts on a system

# Evaluation

## BlockHammer's Hardware Complexity

- We analyze **six state-of-the-art mechanisms** and **BlockHammer**
- We calculate **area**, **access energy**, and **static power** consumption*

| | Mitigation Mechanism | SRAM KB | CAM KB | Area $mm^2$ | %CPU | Access Energy pJ | Static Power mW |
|---|---|---|---|---|---|---|---|
| $N_{RH}$=32K | BlockHammer | 51.48 | 1.73 | 0.14 | 0.06 | 20.30 | 22.27 |
| | PARA [73] | - | - | <0.01 | - | - | - |
| | ProHIT [137] | - | 0.22 | <0.01 | <0.01 | 3.67 | 0.14 |
| | MRLoc [161] | - | 0.47 | <0.01 | <0.01 | 4.44 | 0.21 |
| | CBT [132] | 16.00 | 8.50 | 0.20 | 0.08 | 9.13 | 35.55 |
| | TWiCe [84] | 23.10 | 14.02 | 0.15 | 0.06 | 7.99 | 21.28 |
| | Graphene [113] | - | 5.22 | 0.04 | 0.02 | 40.67 | 3.11 |

BlockHammer is **low cost** and **competitive** with state-of-the-art mechanisms

*Assuming a high-end 28-core Intel Xeon processor system with 4-channel single-rank DDR4 DIMMs with a RowHammer threshold (NRH) of 32K

# Evaluation

## BlockHammer's Hardware Complexity

| | Mitigation Mechanism | SRAM KB | CAM KB | Area mm² | %CPU | Access Energy pJ | Static Power mW |
|---|---|---|---|---|---|---|---|
| $N_{RH}$=32K | BlockHammer | 51.48 | 1.73 | 0.14 | 0.06 | 20.30 | 22.27 |
| | PARA [73] | - | - | <0.01 | - | - | - |
| | ProHIT [137] | - | 0.22 | <0.01 | <0.01 | 3.67 | 0.20 |
| | MRLoc [161] | - | 0.47 | <0.01 | <0.01 | 4.44 | 0.21 |
| | CBT [132] | 16.00 | 8.50 | 0.20 | 0.08 | 9.13 | 35.55 |
| | TWiCe [84] | 23.10 | 14.02 | 0.15 | 0.06 | 7.99 | 21.28 |
| | Graphene [113] | - | 5.22 | 0.04 | 0.02 | 40.67 | 3.11 |
| $N_{RH}$=1K | BlockHammer | 441.33 | 55.58 | 1.57 | 0.64 | 99.64 | 220.99 |
| | PARA [73] | - | - | <0.01 | - | - | - |
| | ProHIT [137] | x | x | x | x | x | x |
| | MRLoc [161] | x | x | x | x | x | x |
| | CBT [132] | 512.00 | 272.00 | 3.95 | 1.60 | 127.93 | 535.50 |
| | TWiCe [84] | 738.32 | 448.27 | 5.17 | 2.10 | 124.79 | 631.98 |
| | Graphene [113] | - | 166.03 | 1.14 | 0.46 | 917.55 | 93.96 |

10x 5x 10x

20x 35x 23x

23x

15x 30x 30x

BlockHammer's hardware complexity **scales more efficiently** than state-of-the-art mechanisms

# Evaluation
## Performance and DRAM Energy

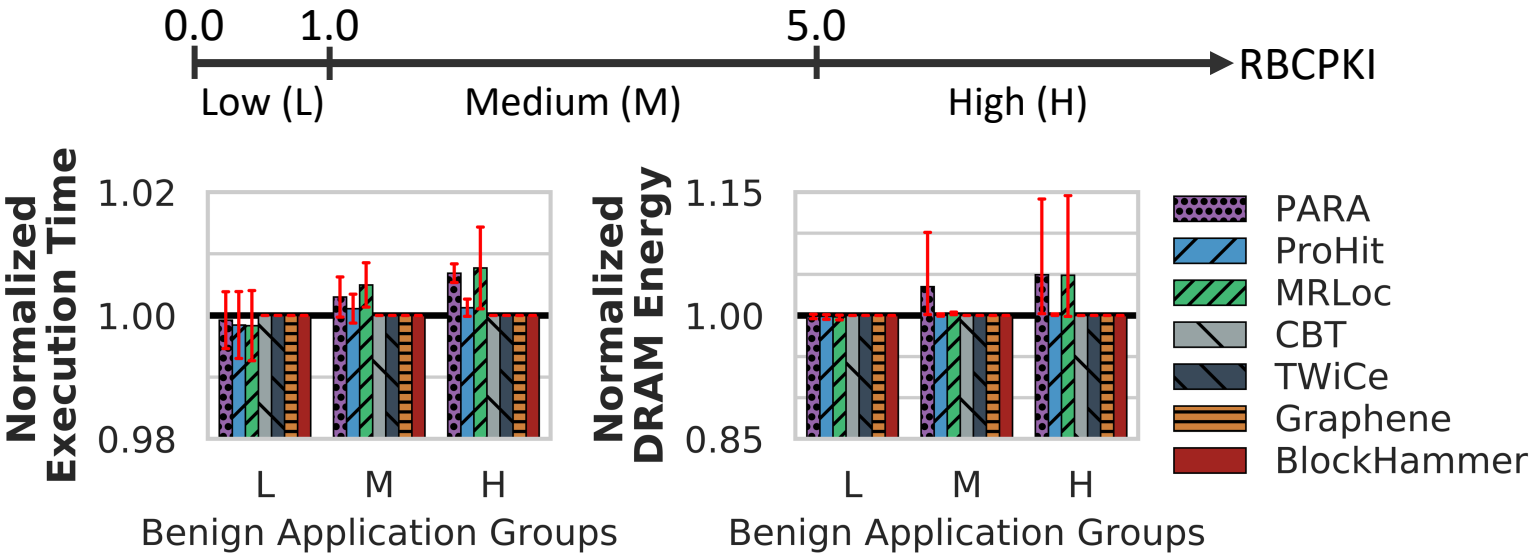
- Cycle-level simulations using **Ramulator** and **DRAMPower**
- System Configuration:

| | |
|---|---|
| **Processor** | 3.2 GHz, {1,8} core, 4-wide issue, 128-entry instr. window |
| **LLC** | 64-byte cacheline, 8-way set-associative, {2,16} MB |
| **Memory scheduler** | FR-FCFS |
| **Address mapping** | Minimalistic Open Pages |
| **DRAM** | DDR4 1 channel, 1 rank, 4 bank group, 4 banks per bank group |
| **RowHammer Threshold** | 32K |

- Single-Core Benign Workloads:
  - 22 SPEC CPU 2006
  - 4 YCSB Disk I/O
  - 2 Network Accelerator Traces
  - 2 Bulk Data Copy with Non-Temporal Hint (movnti)
- Randomly Chosen Multiprogrammed Workloads:
  - 125 workloads containing **8 benign applications**
  - 125 workloads containing **7 benign applications** and **1 RowHammer attack thread**

# Evaluation
## Performance and DRAM Energy

- We classify single-core workloads into three categories based on row buffer conflicts per thousand instructions



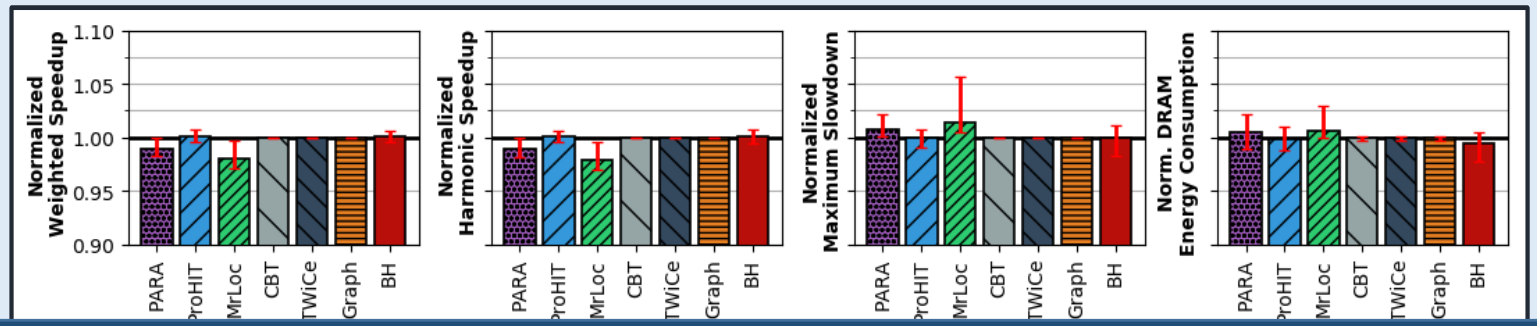

- No application's row activation count exceeds BlockHammer's blacklisting threshold ($N_{BL}$)

BlockHammer does not incur **performance** or **DRAM energy** overheads for single-core benign applications

# Evaluation
## Performance and DRAM Energy

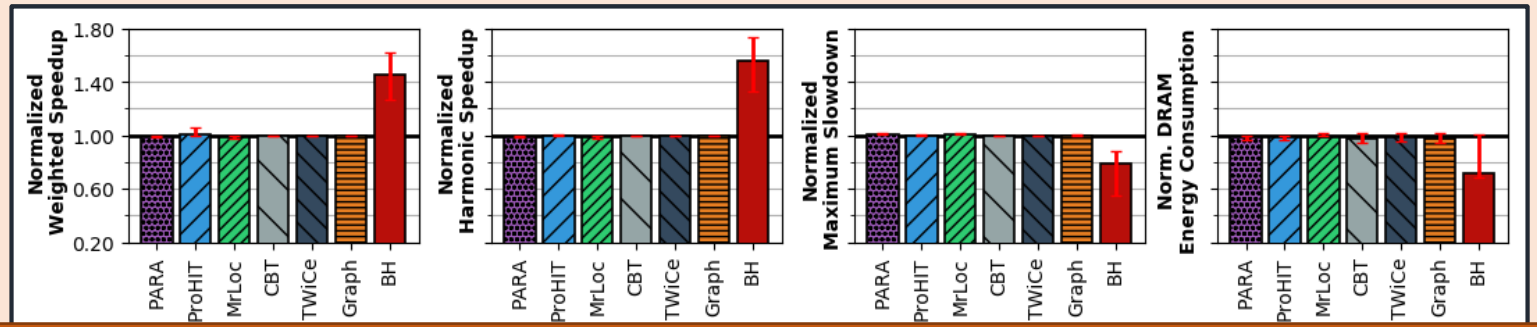
- System throughput (weighted speedup)
- Job turnaround time (harmonic speedup)
- Unfairness (maximum slowdown)
- DRAM energy consumption

No RowHammer Attack




BlockHammer introduces **very low** performance (<0.5%) and DRAM energy (<0.4%) overheads

RowHammer Attack Present

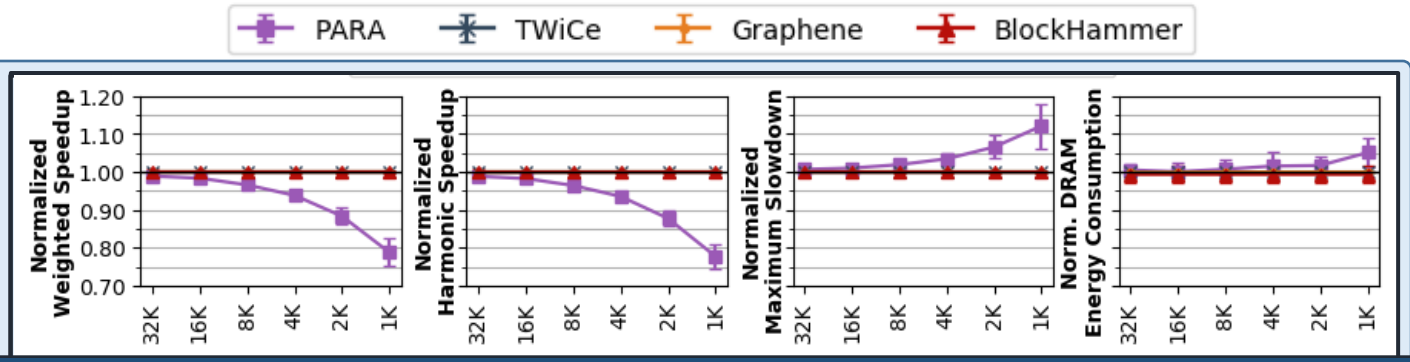



BlockHammer **significantly increases** benign application performance (by 45% on average) and **reduces** DRAM energy consumption (by 29% on average)

# Evaluation
## Scaling with RowHammer Vulnerability

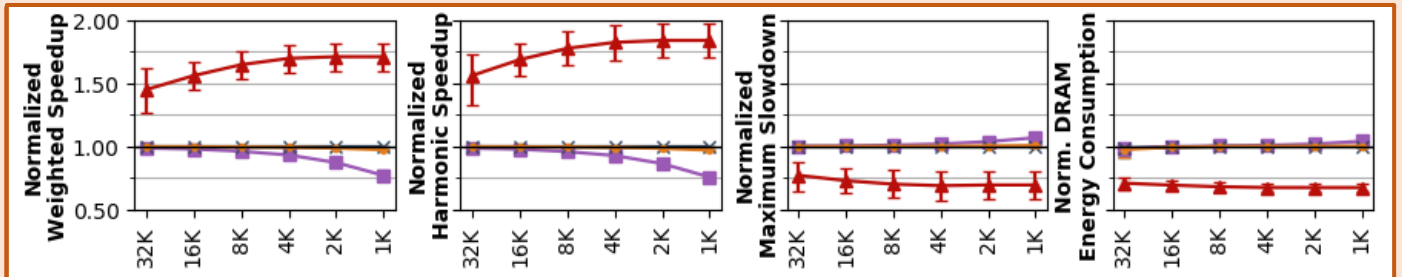
- System throughput (weighted speedup)
- Job turnaround time (harmonic speedup)
- Unfairness (maximum slowdown)
- DRAM energy consumption

No RowHammer Attack




BlockHammer's performance and energy overheads remain **negligible (<0.6%)**

RowHammer Attack Present




BlockHammer scalably provides **much higher performance** (71% on average) and **lower energy consumption** (32% on average) than state-of-the-art mechanisms