Revisiting RowHammer
An Experimental Analysis of Modern Devices and Mitigation Techniques

Jeremie S. Kim   **Minesh Patel**

A. Giray Yağlıkçı   **Hasan Hassan**

Roknoddin Azizi   **Lois Orosa**   **Onur Mutlu**

SAFARI

ETH *Zürich*   **Carnegie Mellon**

Executive Summary

- **<u>Motivation</u>**: Denser DRAM chips are more vulnerable to RowHammer but no characterization-based study demonstrates how vulnerability scales
- **<u>Problem</u>**: Unclear if existing mitigation mechanisms will remain viable for future DRAM chips that are likely to be more vulnerable to RowHammer
- **<u>Goal</u>**:
    1. Experimentally demonstrate how vulnerable modern DRAM chips are to RowHammer and study how this vulnerability will scale going forward
    2. Study viability of existing mitigation mechanisms on more vulnerable chips
- **<u>Experimental Study</u>**: First rigorous RowHammer characterization study across a broad range of DRAM chips
    - 1580 chips of different DRAM {types, technology node generations, manufacturers}
    - We find that RowHammer vulnerability worsens in newer chips
- **<u>RowHammer Mitigation Mechanism Study</u>**: How five state-of-the-art mechanisms are affected by worsening RowHammer vulnerability
    - Reasonable performance loss (8% on average) on modern DRAM chips
    - Scale poorly to more vulnerable DRAM chips (e.g., 80% performance loss)
- **<u>Conclusion:</u>** it is critical to research more effective solutions to RowHammer for future DRAM chips that will likely be even more vulnerable to RowHammer

SAFARI

Outline

Outline

SAFARI
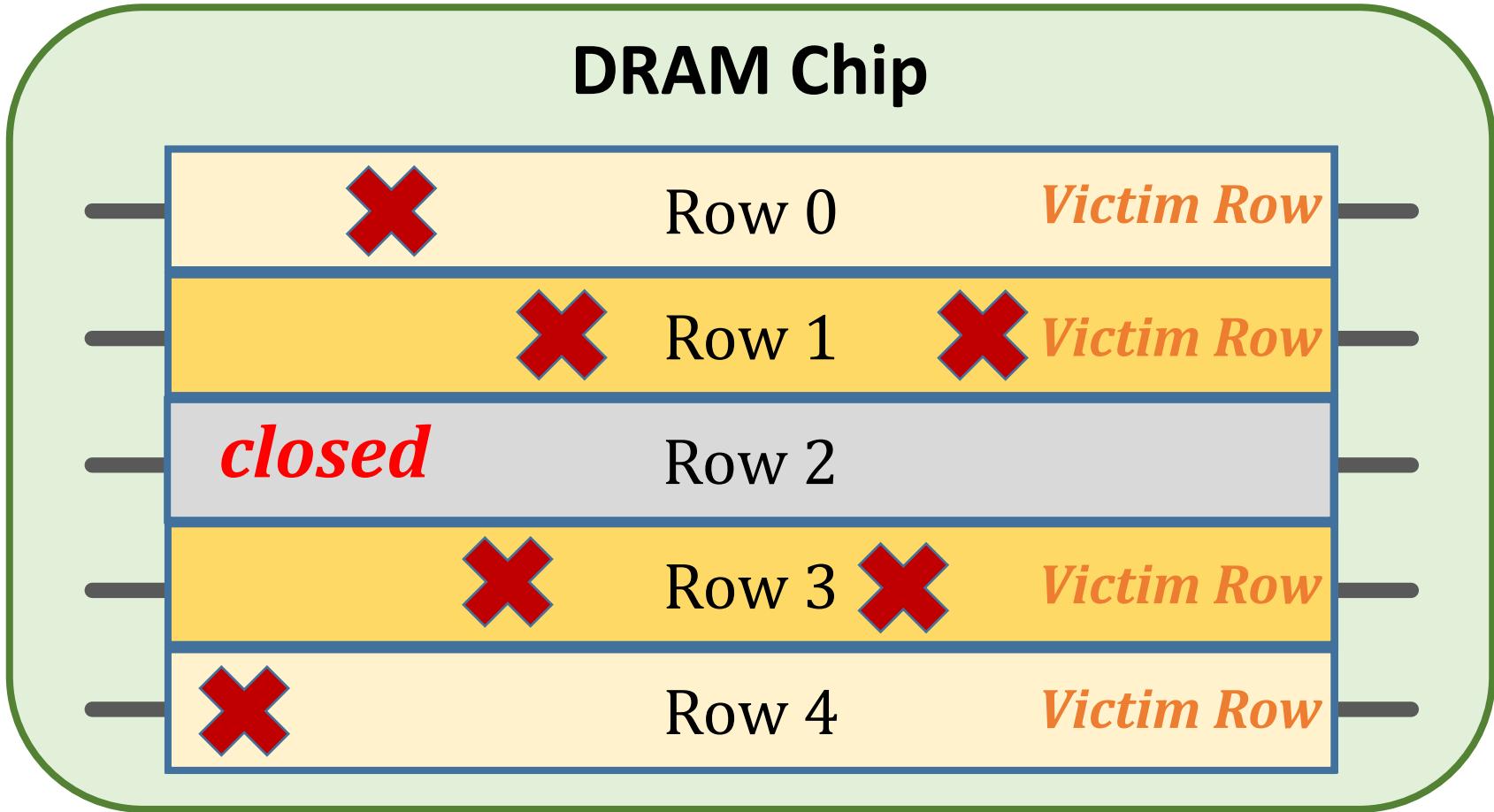
The RowHammer Vulnerability

Repeatedly **opening** (activating) and **closing** (precharging)
a DRAM row causes **RowHammer bit flips** in nearby cells

SAFARI

Outline

SAFARI

DRAM Organization

Row of storage cells

DRAM Bank

Row 1

Row 2

Row 3

Row 4
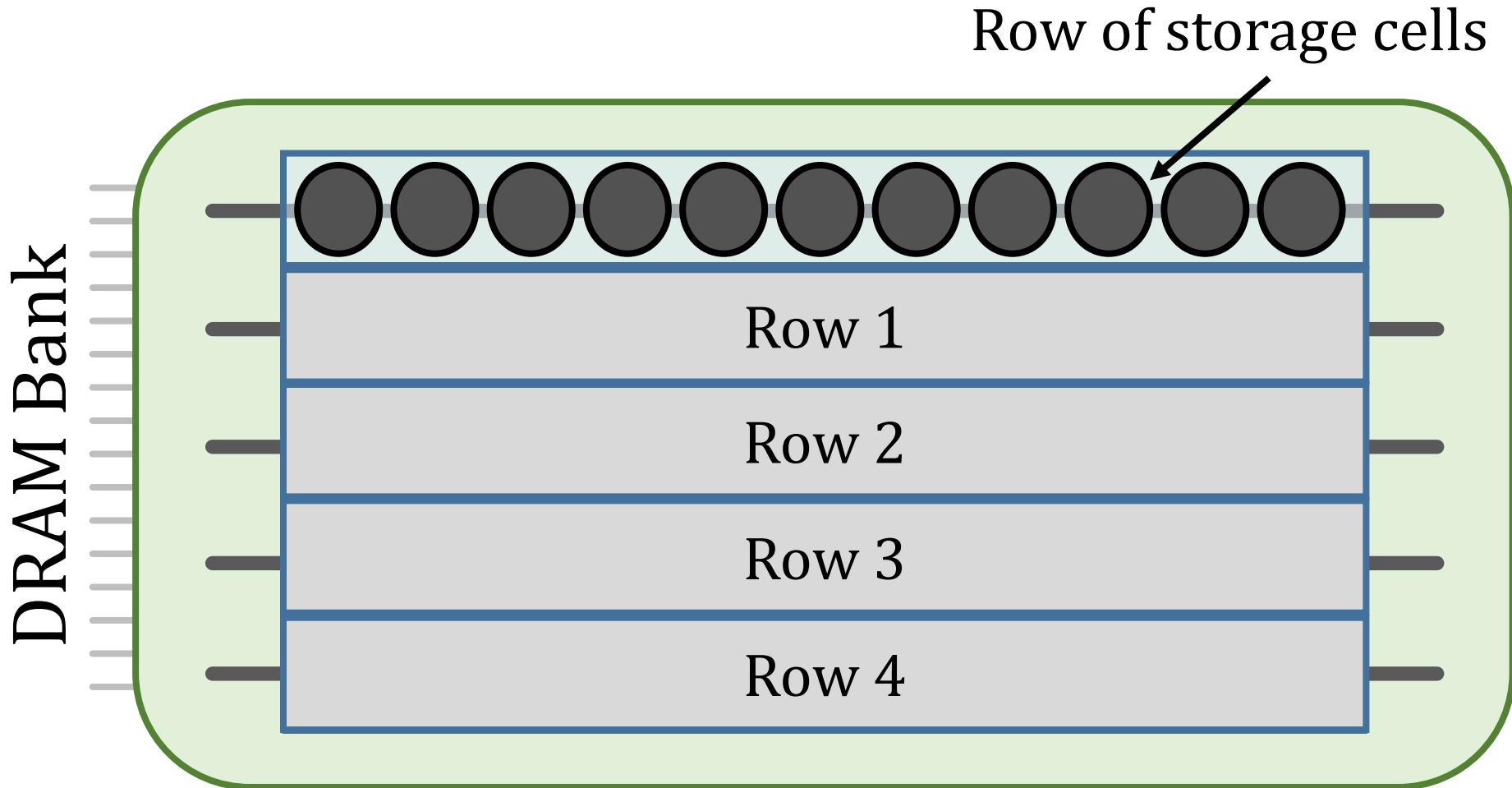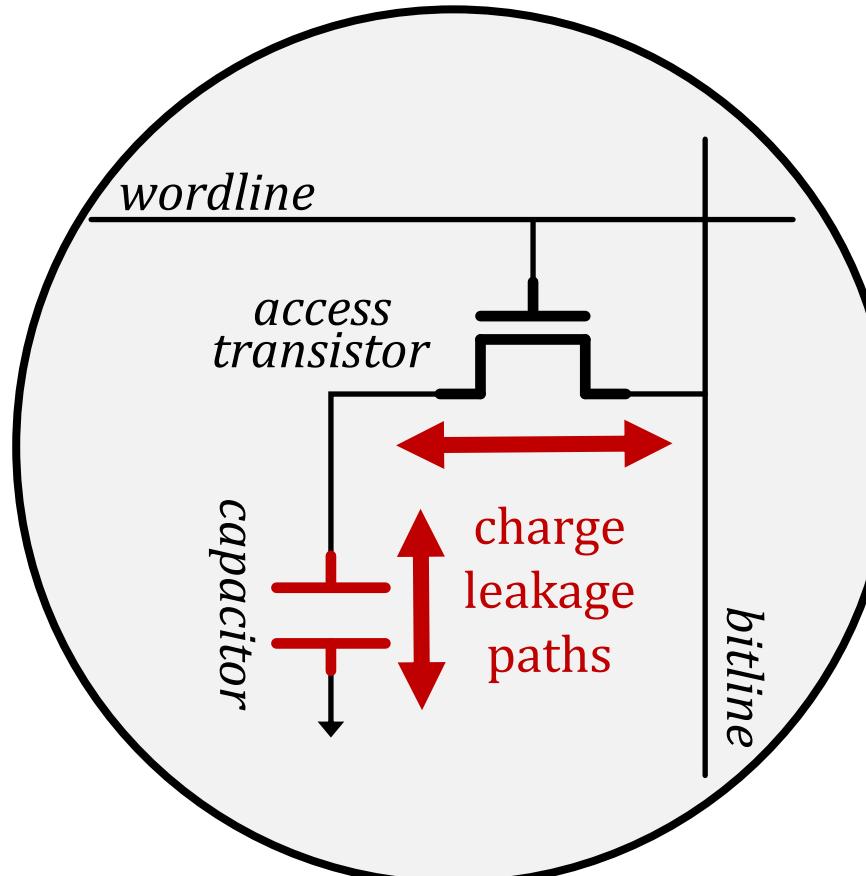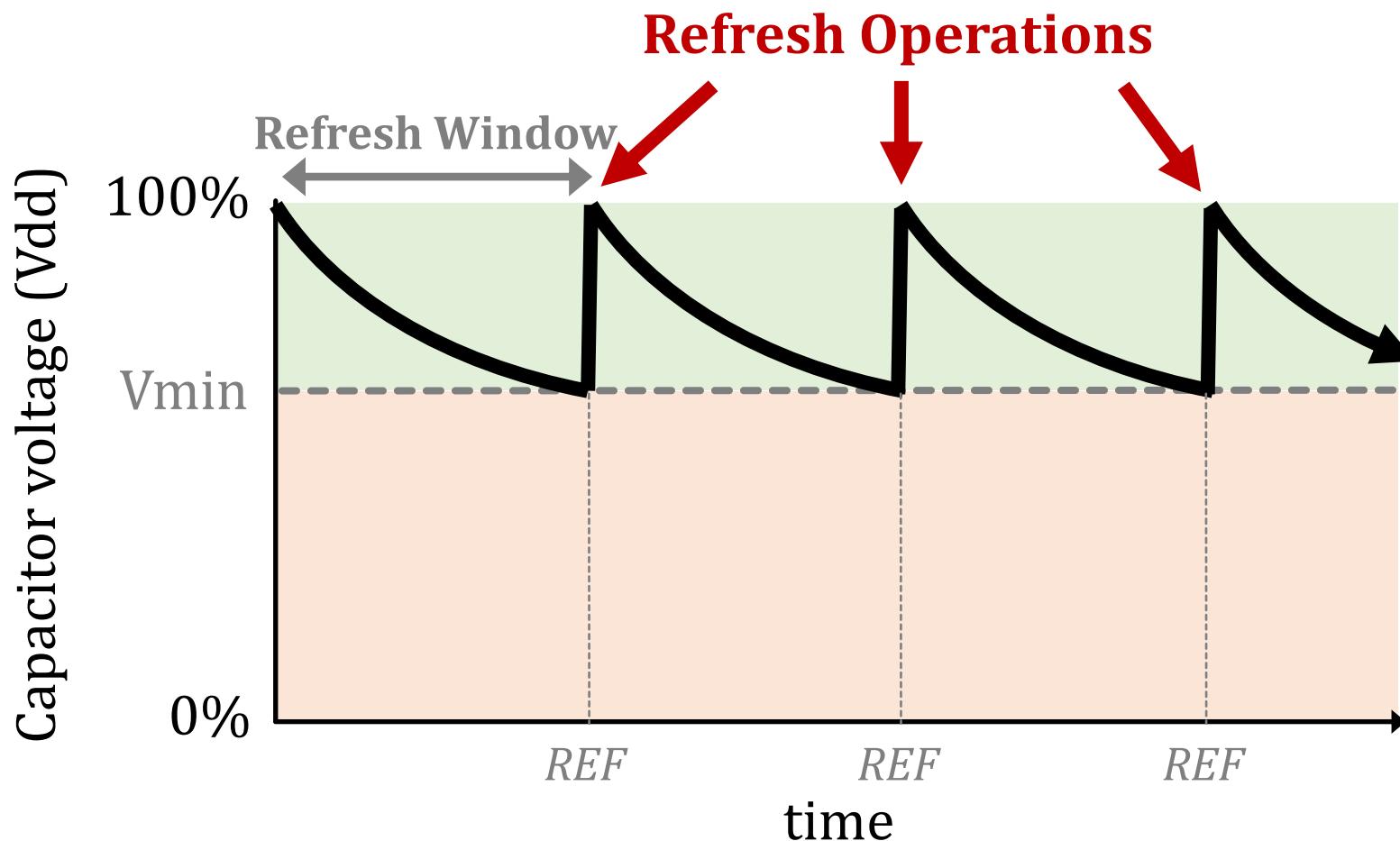
DRAM Cell Leakage

Each cell encodes information in **leaky** capacitors

wordline

access transistor

capacitor

charge leakage paths

bitline
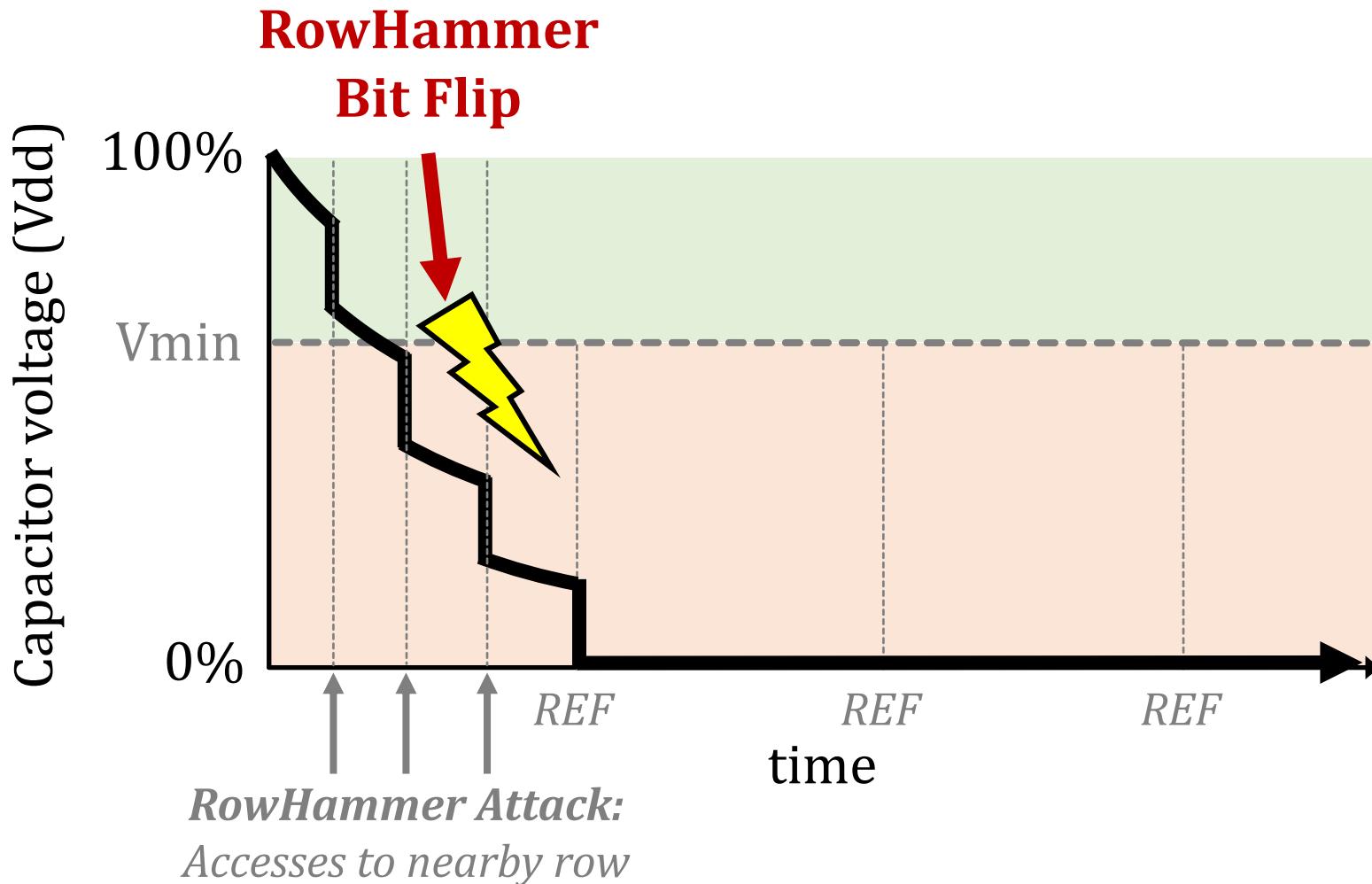
Stored data is **corrupted** if too much charge leaks (i.e., the capacitor voltage degrades too much)

SAFARI

[Patel+, ISCA'17]
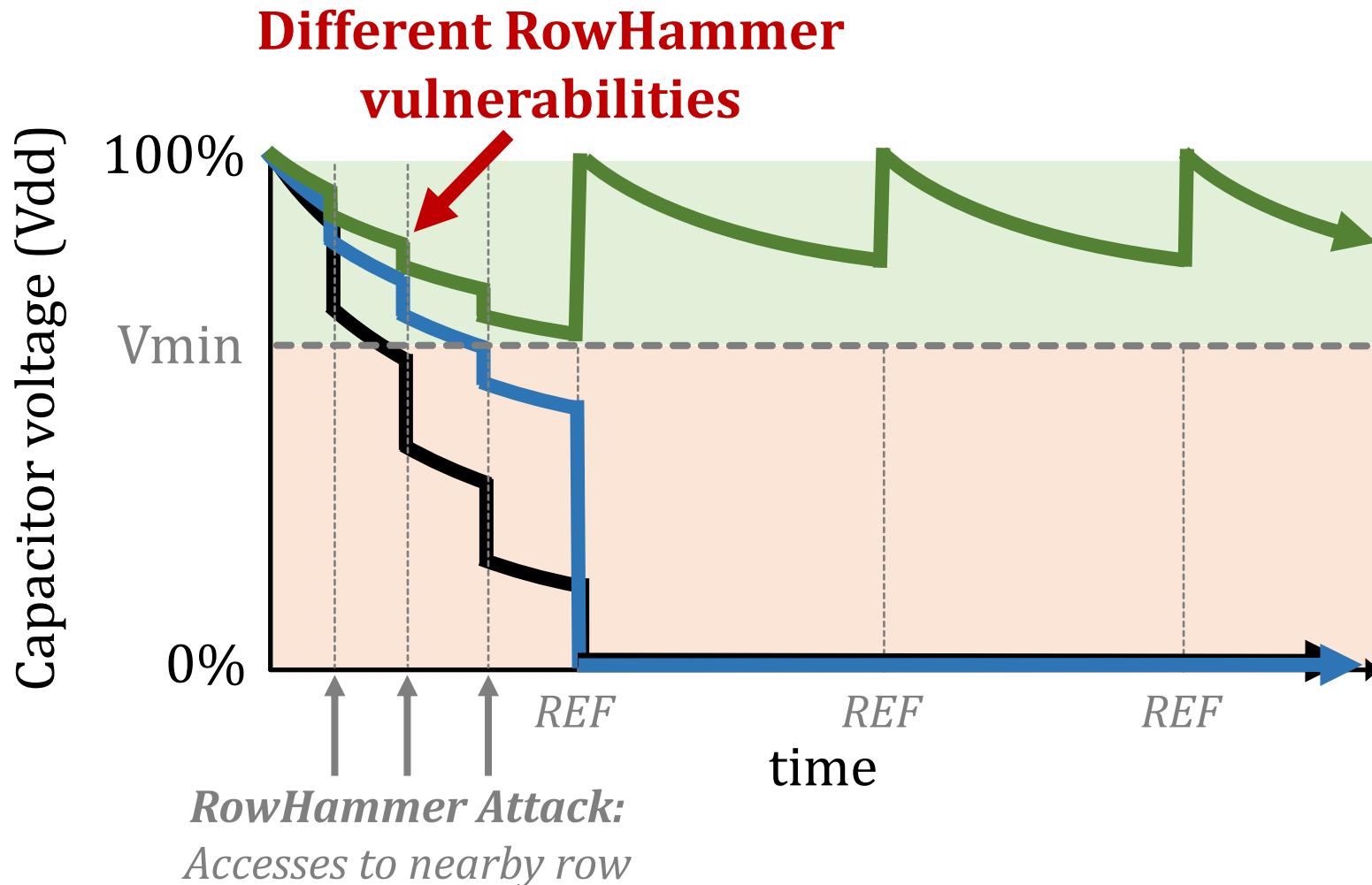
DRAM Refresh

Periodic **refresh operations** preserve stored data

RowHammer Bit Flips

Cell-to-Cell Variation

Different RowHammer vulnerabilities

Capacitor voltage (Vdd)

100%

Vmin

0%

RowHammer Attack:
Accesses to nearby row

REF   REF   REF

time

Some cells are more vulnerable due to **process variation**

Outline

SAFARI

12

Motivation

- Denser DRAM chips are **more vulnerable** to RowHammer

- Three prior works **[Kim+, ISCA'14], [Park+, MR'16], [Park+, MR'16]**, **over the last six years** provide RowHammer characterization data on real DRAM

- However, there is **no comprehensive experimental study** that demonstrates **how vulnerability scales** across DRAM types and technology node generations

- It is **unclear whether current mitigation mechanisms will remain viable** for future DRAM chips that are likely to be more vulnerable to RowHammer

SAFARI

Goal

1. **Experimentally demonstrate** how vulnerable modern DRAM chips are to RowHammer and **predict how this vulnerability will scale** going forward

2. Examine the viability of current mitigation mechanisms on **more vulnerable chips**

SAFARI

Outline

SAFARI

DRAM Testing Infrastructures
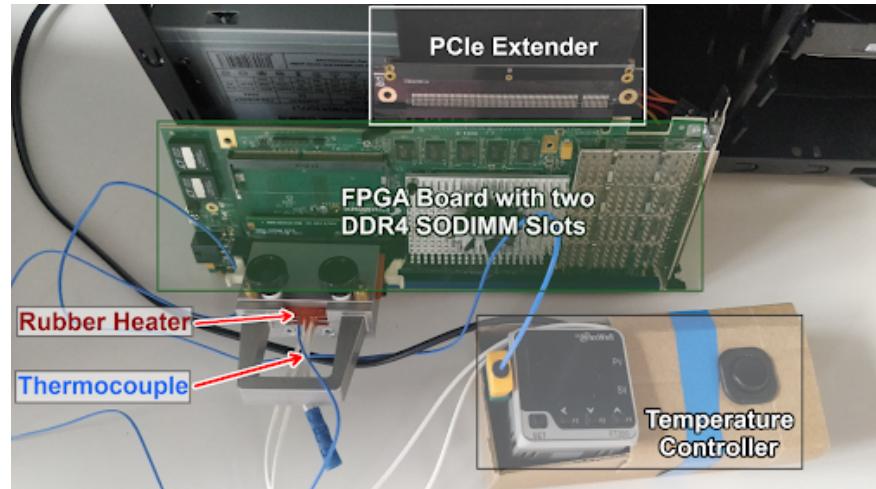
Three separate testing infrastructures
1. **DDR3:** FPGA-based SoftMC [Hassan+, HPCA'17] (Xilinx ML605)
2. **DDR4:** FPGA-based SoftMC [Hassan+, HPCA'17] (Xilinx Virtex UltraScale 95)
3. **LPDDR4:** In-house testing hardware for LPDDR4 chips

All provide fine-grained control over DRAM commands, timing parameters and temperature

DDR4 DRAM testing infrastructure

SAFARI

DRAM Chips Tested

DRAM type-node	Number of Chips (Modules) Tested			
	Mfr. A	*Mfr. B*	*Mfr. C*	*Total*
DDR3-old	56 (10)	88 (11)	28 (7)	**172 (28)**
DDR3-new	80 (10)	52 (9)	104 (13)	**236 (32)**
DDR4-old	112 (16)	24 (3)	128 (18)	**264 (37)**
DDR4-new	264 (43)	16 (2)	108 (28)	**388 (73)**
LPDDR4-1x	12 (3)	180 (45)	N/A	**192 (48)**
LPDDR4-1y	184 (46)	N/A	144 (36)	**328 (82)**

1580 total DRAM chips tested from **300** DRAM modules

- **Three** major DRAM manufacturers {A, B, C}
- **Three** DRAM *types* or *standards* {DDR3, DDR4, LPDDR4}
  - LPDDR4 chips we test implement on-die ECC
- **Two** technology nodes per DRAM type {old/new, 1x/1y}
  - Categorized based on manufacturing date, datasheet publication date, purchase date, and characterization results

Type-node: configuration describing a chip's type and technology node generation: **DDR3-old/new, DDR4-old/new, LPDDR4-1x/1y**

Effective RowHammer Characterization

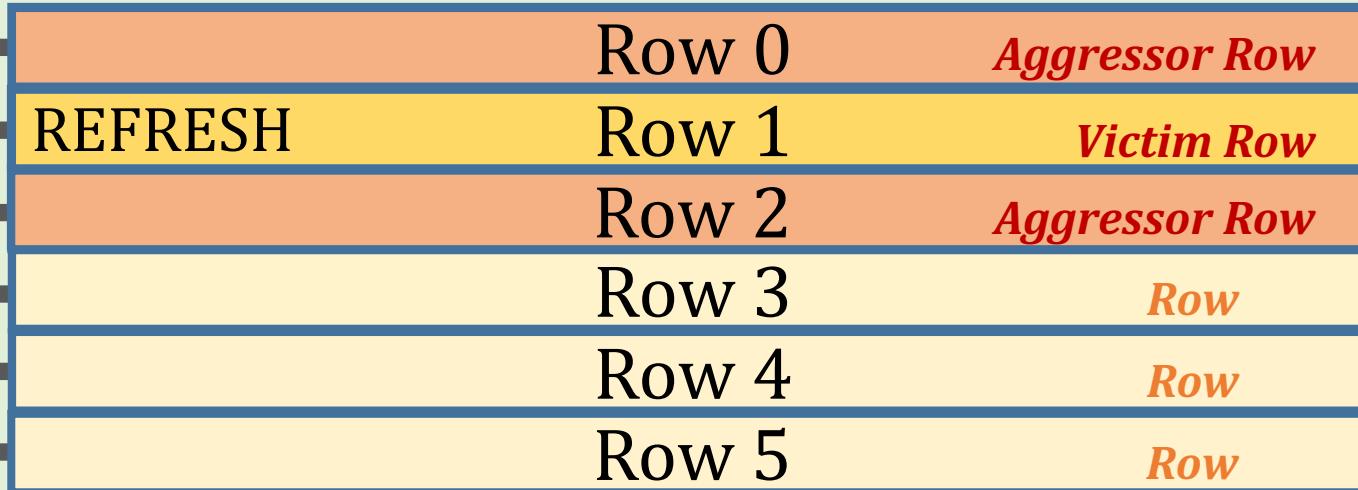To characterize our DRAM chips at **worst-case** conditions, we:

1. **Prevent sources of interference during core test loop**
   - We disable:
     - **DRAM refresh**: to avoid refreshing victim row
     - **DRAM calibration events**: to minimize variation in test timing
     - **RowHammer mitigation mechanisms**: to observe circuit-level effects
   - Test for **less than refresh window (32ms)** to avoid retention failures

2. **Worst-case access sequence**

   - We use **worst-case** access sequence based on prior works' observations

   - For each row, **repeatedly access the two directly physically-adjacent rows as fast as possible**

[More details in the paper]

SAFARI

Testing Methodology

	Row 0	*Aggressor Row*
REFRESH	Row 1	*Victim Row*
	Row 2	*Aggressor Row*
	Row 3	*Row*
	Row 4	*Row*
	Row 5	*Row*

DRAM_RowHammer_Characterization():
  foreach *row* in *DRAM*:
      set *victim_row* to *row*
      set *aggressor_row*1 to *victim_row* − 1
      set *aggressor_row*2 to *victim_row* + 1
      Disable DRAM refresh
      Refresh *victim_row*
      for *n* = 1 → *HC*: // core test loop
         activate *aggressor_row*1
         activate *aggressor_row*2
    Enable DRAM refresh
    Record RowHammer bit flips to storage
    Restore bit flips to original values

Disable refresh to **prevent interruptions** in the core loop of our test **from refresh operations**

Induce RowHammer bit flips on a **fully charged row**

Testing Methodology

closed	Row 0	*Aggressor Row*
	Row 1	*Aggressor Row*
	Row 2	*Row*
	Row 3	*Aggressor Row*
	Row 4	*Victim Row*
	Row 5	*Aggressor Row*

DRAM_RowHammer_Characterization():
   foreach *row* in *DRAM*:
       set *victim_row* to *row*
       set *aggressor_row*1 to *victim_row* − 1
       set *aggressor_row*2 to *victim_row* + 1
       Disable DRAM refresh
       Refresh *victim_row*
       for *n* = 1 → *HC*: // core test loop
          activate *aggressor_row*1
          activate *aggressor_row*2
       Enable DRAM refresh
       Record RowHammer bit flips to storage
       Restore bit flips to original values

Disable refresh to **prevent interruptions** in the core loop of our test **from refresh operations**

Induce RowHammer bit flips on a **fully charged row**

Core test loop where we alternate accesses to adjacent rows

1 Hammer (HC) = two accesses

Prevent further retention failures

Record bit flips for analysis

Outline

RowHammer Introduction

DRAM Background

Motivation and Goal

Experimental Methodology

Characterization Results

Evaluation of Mitigation Mechanisms

RowHammer Solutions Going Forward

Conclusion

SAFARI

Key Takeaways from 1580 Chips

- Chips of newer DRAM technology nodes are **more vulnerable** to RowHammer

- There are chips today whose weakest cells fail after **only 4800 hammers**

- Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

SAFARI

1. RowHammer Vulnerability

Q. Can we induce RowHammer bit flips in all of our DRAM chips?

All chips are vulnerable, except many DDR3 chips

- A total of 1320 out of all 1580 chips **(84%)** are vulnerable

- Within **DDR3-old** chips, **only 12%** of chips (24/204) are vulnerable

- Within **DDR3-new** chips, **65%** of chips (148/228) are vulnerable

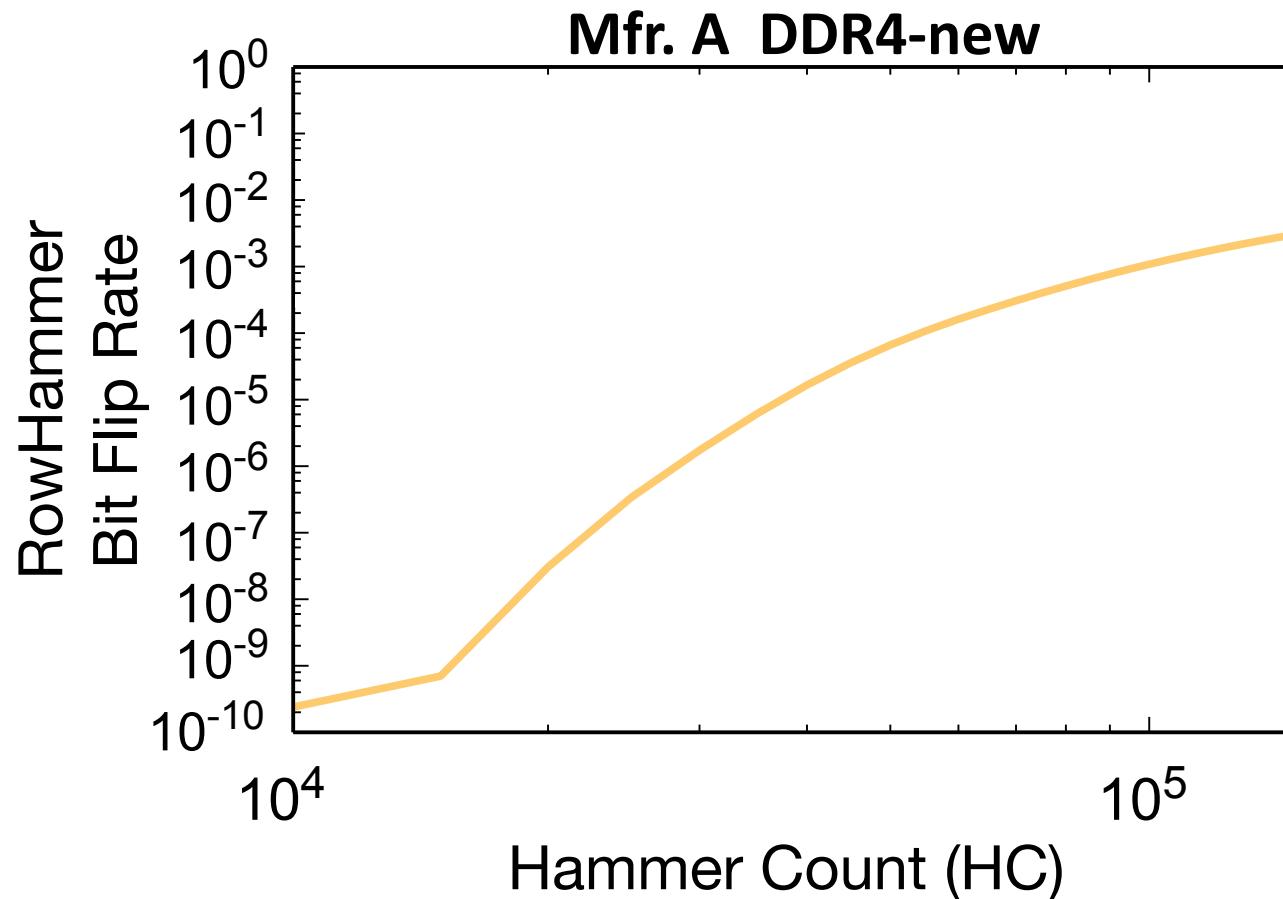**Newer DRAM chips are more vulnerable to RowHammer**

2. Data Pattern Dependence

Q. Are some data patterns more effective in inducing RowHammer bit flips?

- We test **several data patterns** typically examined in prior work to identify the worst-case data pattern

- The worst-case data pattern is **consistent across chips** of the same manufacturer and DRAM type-node configuration

- We use the **worst-case data pattern** per DRAM chip to characterize each chip at **worst-case conditions** and **minimize the extensive testing time**
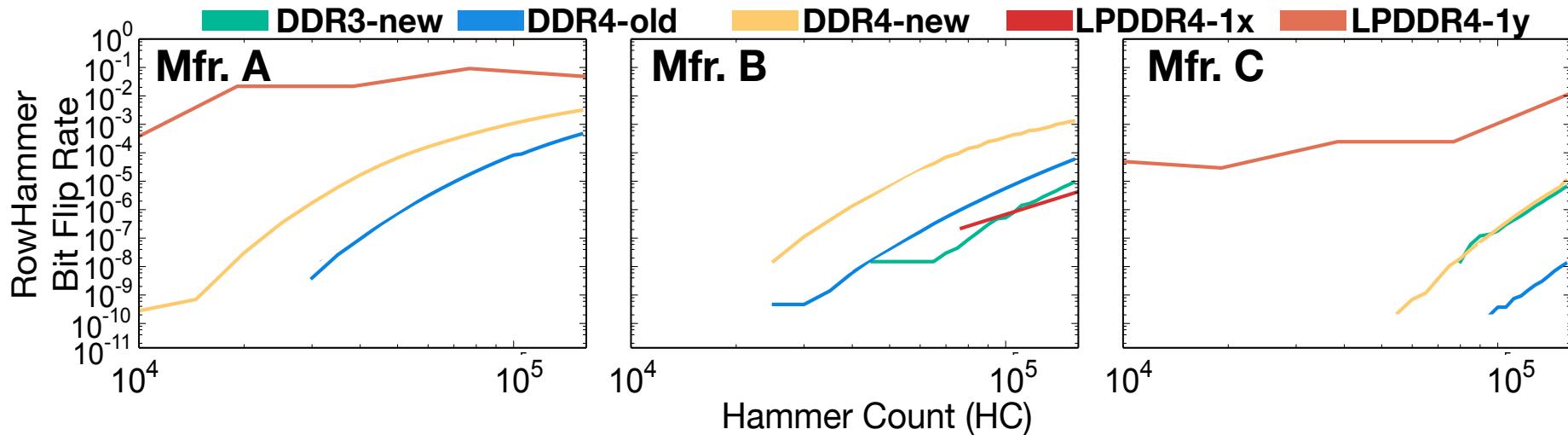
[More detail and figures in paper]

SAFARI

3. Hammer Count (HC) Effects

Q. How does the Hammer Count affect the number of bit flips induced?

Mfr. A DDR4-new

RowHammer Bit Flip Rate vs Hammer Count (HC)

Y-axis: RowHammer Bit Flip Rate, from 10^{-10} to 10^{0}

X-axis: Hammer Count (HC), from 10^{4} to 10^{5}

**Hammer Count = 2 Accesses,
one to each adjacent row of victim**

3. Hammer Count (HC) Effects
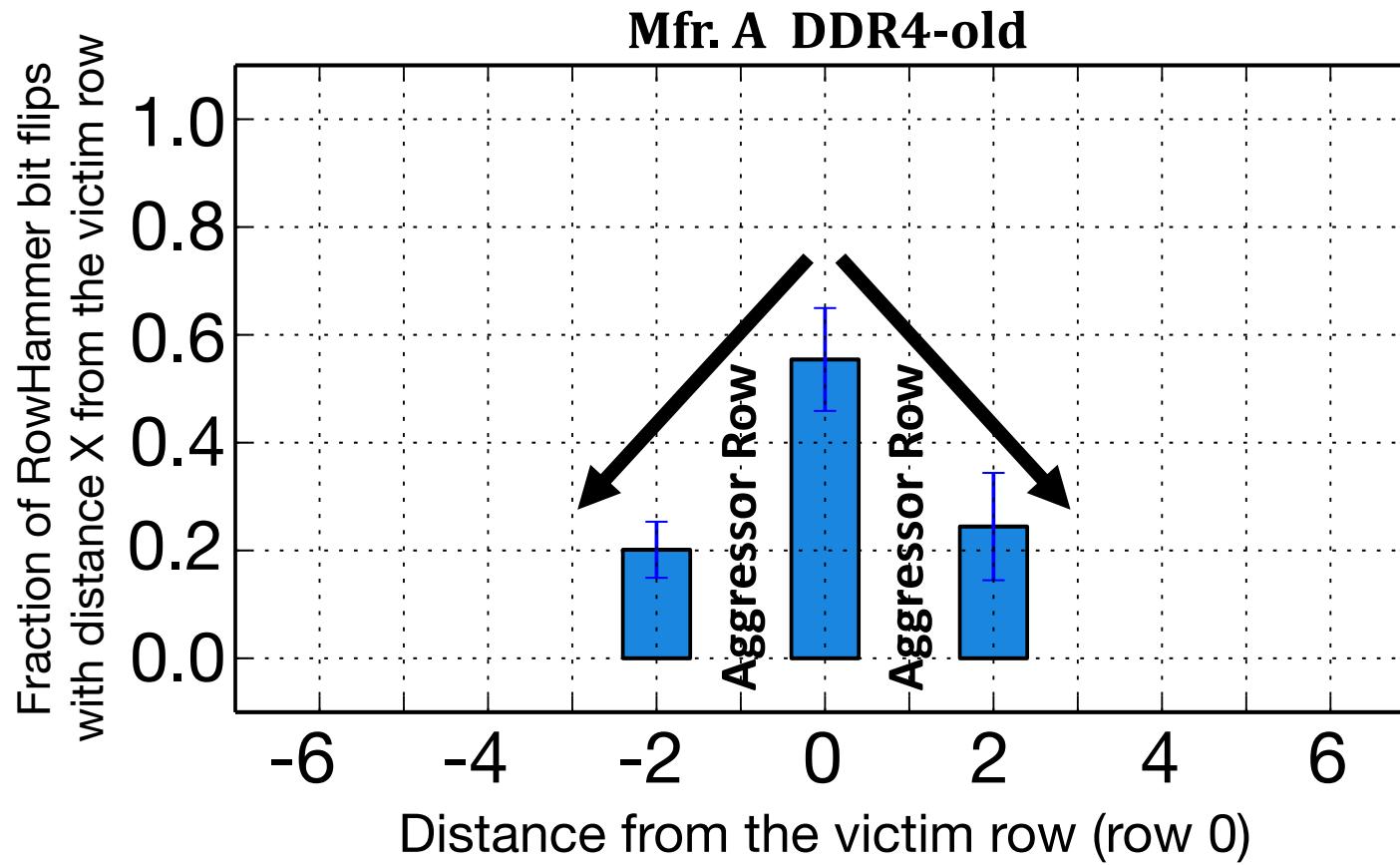
RowHammer bit flip rates **increase**
when going **from old to new** DDR4 technology node generations

RowHammer bit flip rates (i.e., RowHammer vulnerability) increase with technology node generation
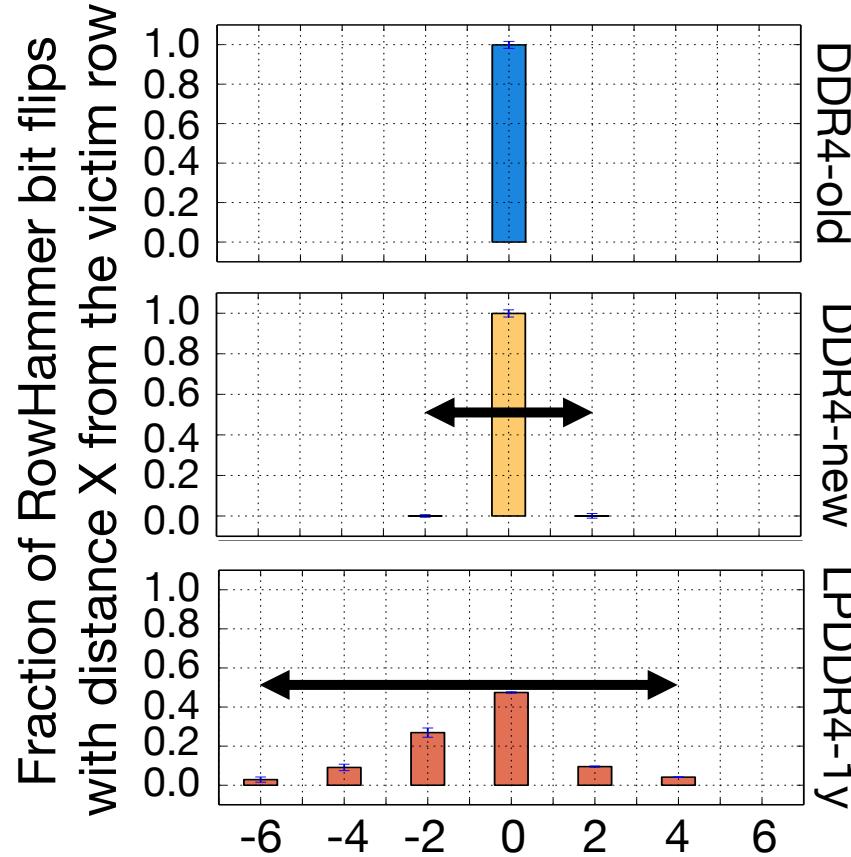
4. Spatial Effects: Row Distance

Q. Where do RowHammer bit flips occur relative to aggressor rows?

Mfr. A DDR4-old

The number of RowHammer bit flips that occur in a given row decreases as the distance from the **victim row (row 0)** increases.
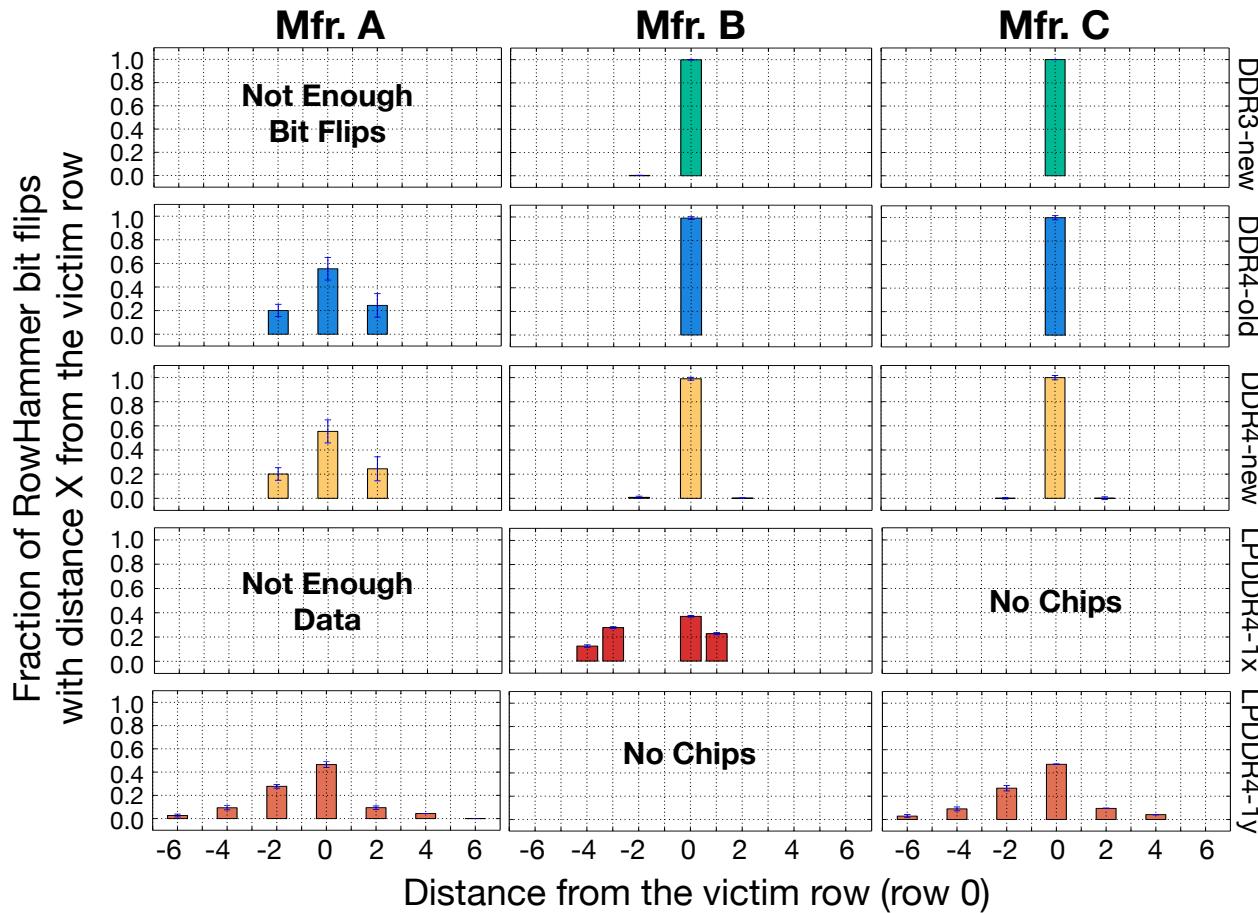
We normalize data by inducing a bit flip rate of 10^{-6} in each chip

Fraction of RowHammer bit flips with distance X from the victim row

Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

SAFARI

4. Spatial Effects: Row Distance

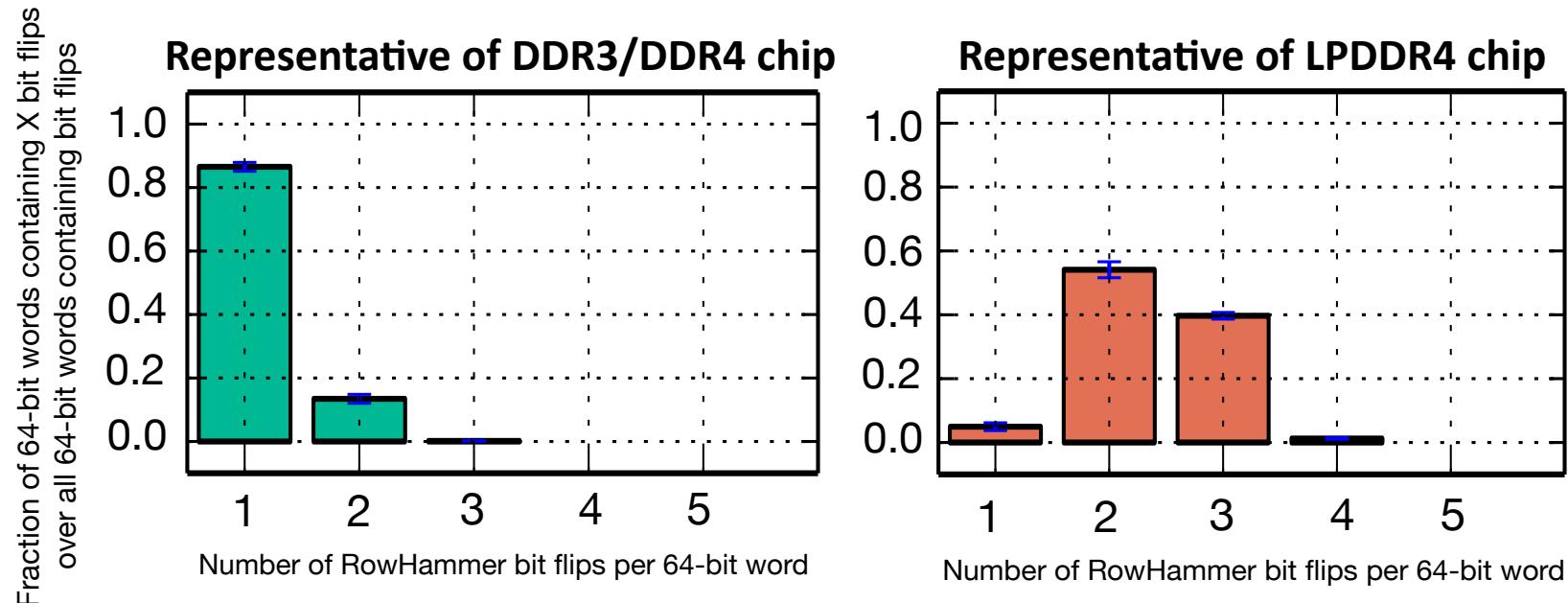We plot this data for each DRAM type-node configuration per manufacturer

SAFARI

4. Spatial Distributions

We normalize data by inducing a bit flip rate of 10^{-6} in each chip

Representative of DDR3/DDR4 chip

Representative of LPDDR4 chip

Fraction of 64-bit words containing X bit flips over all 64-bit words containing bit flips

Number of RowHammer bit flips per 64-bit word

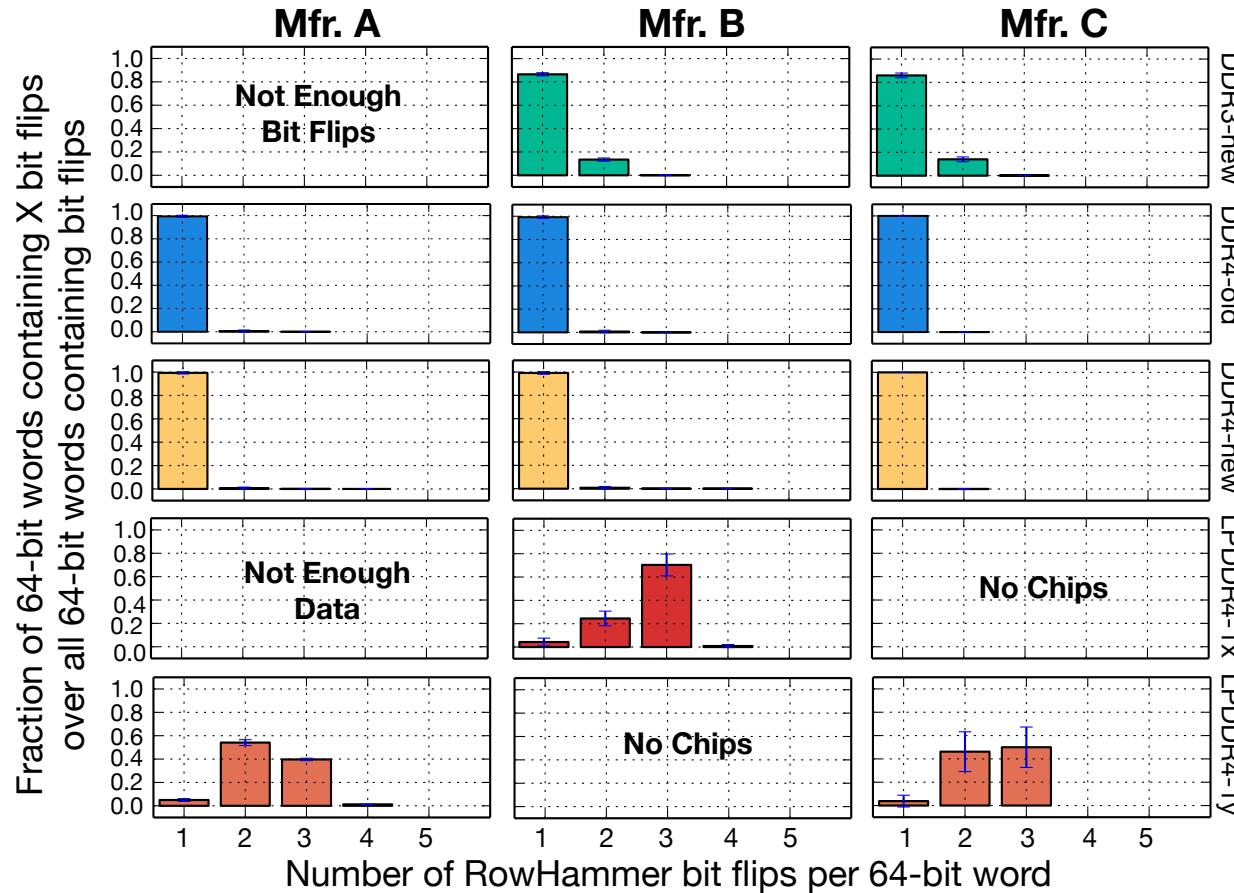Number of RowHammer bit flips per 64-bit word

The distribution of RowHammer bit flip density per word
changes significantly in LPDDR4 chips from other DRAM types

At a bit flip rate of 10^{-6}, a 64-bit word can contain up to **4 bit flips**.
Even at this very low bit flip rate, a **very strong ECC** is required
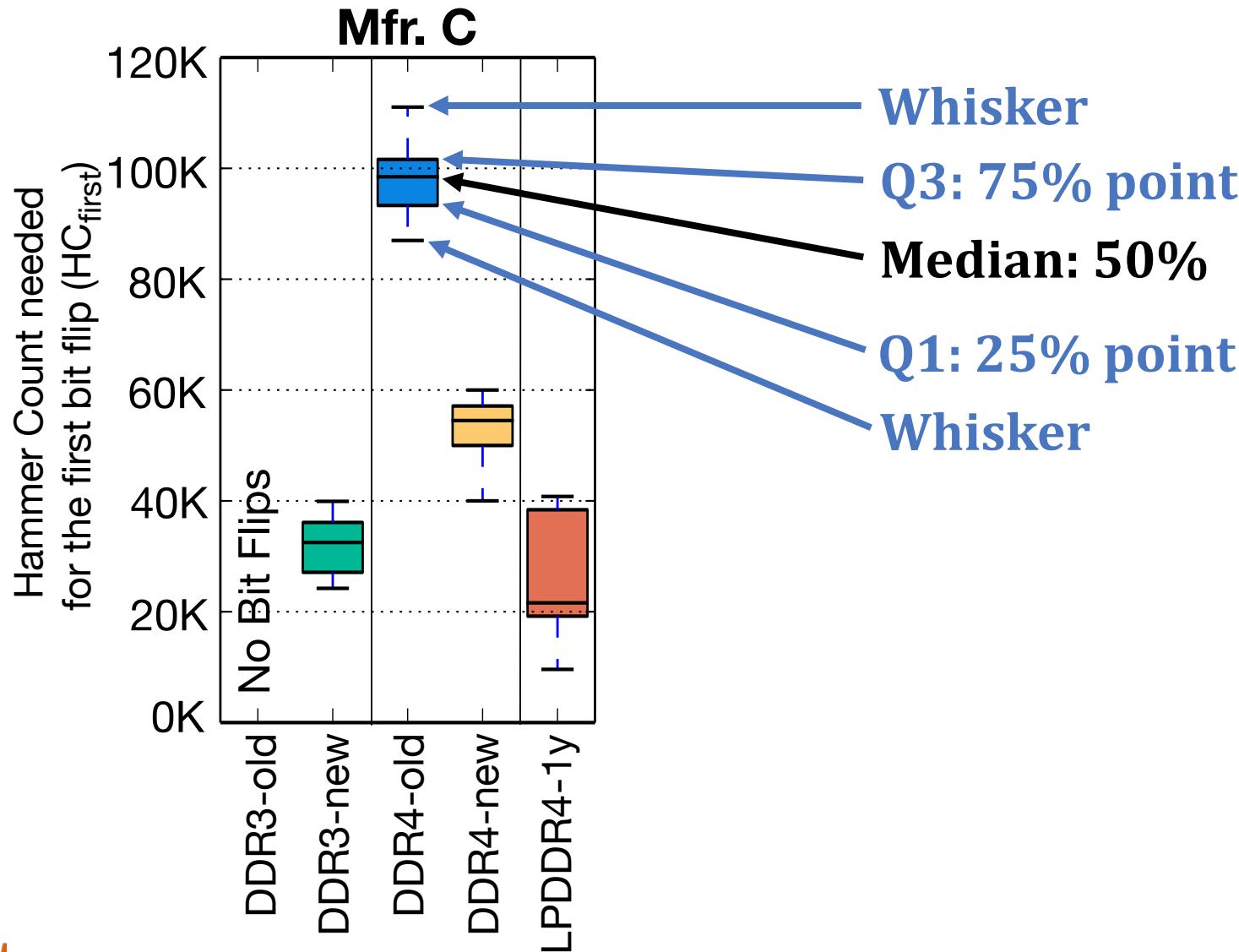
4. Spatial Distribution of Bit Flips

We plot this data for each DRAM type-node configuration per manufacturer
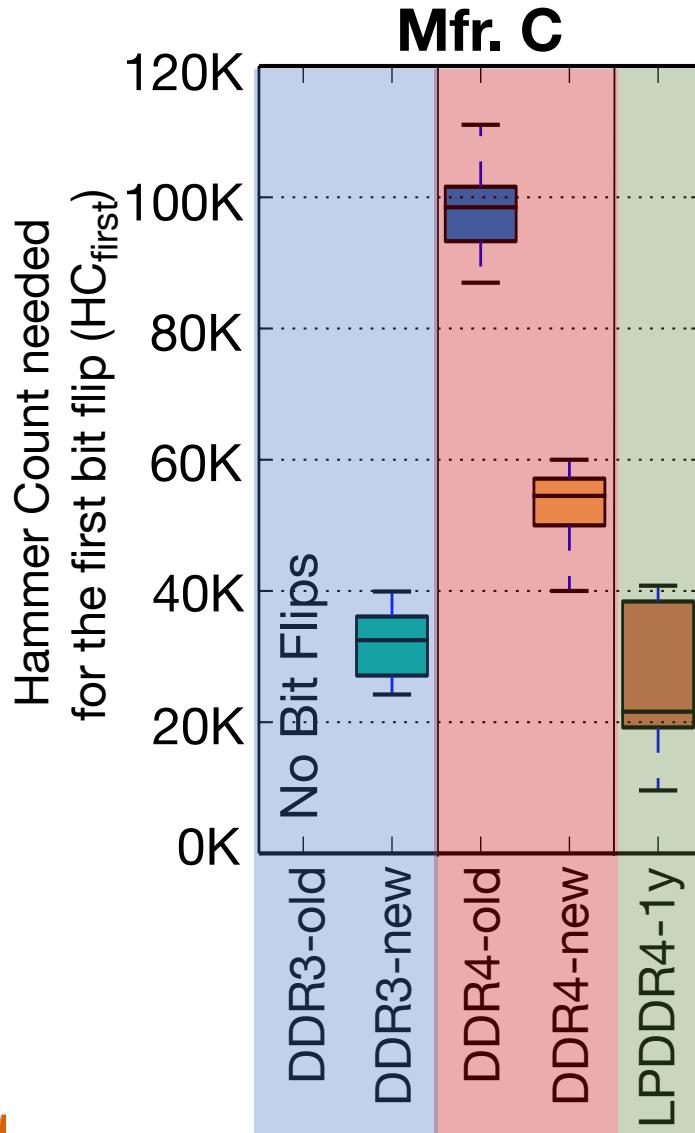

[More analysis in the paper]

5. First RowHammer Bit Flips per Chip

What is the minimum Hammer Count required to cause bit flips (HC_{first})?

Mfr. C

Whisker

Q3: 75% point

Median: 50%

Q1: 25% point

Whisker

Hammer Count needed for the first bit flip (HC_{first})

No Bit Flips

DDR3-old   DDR3-new   DDR4-old   DDR4-new   LPDDR4-1y
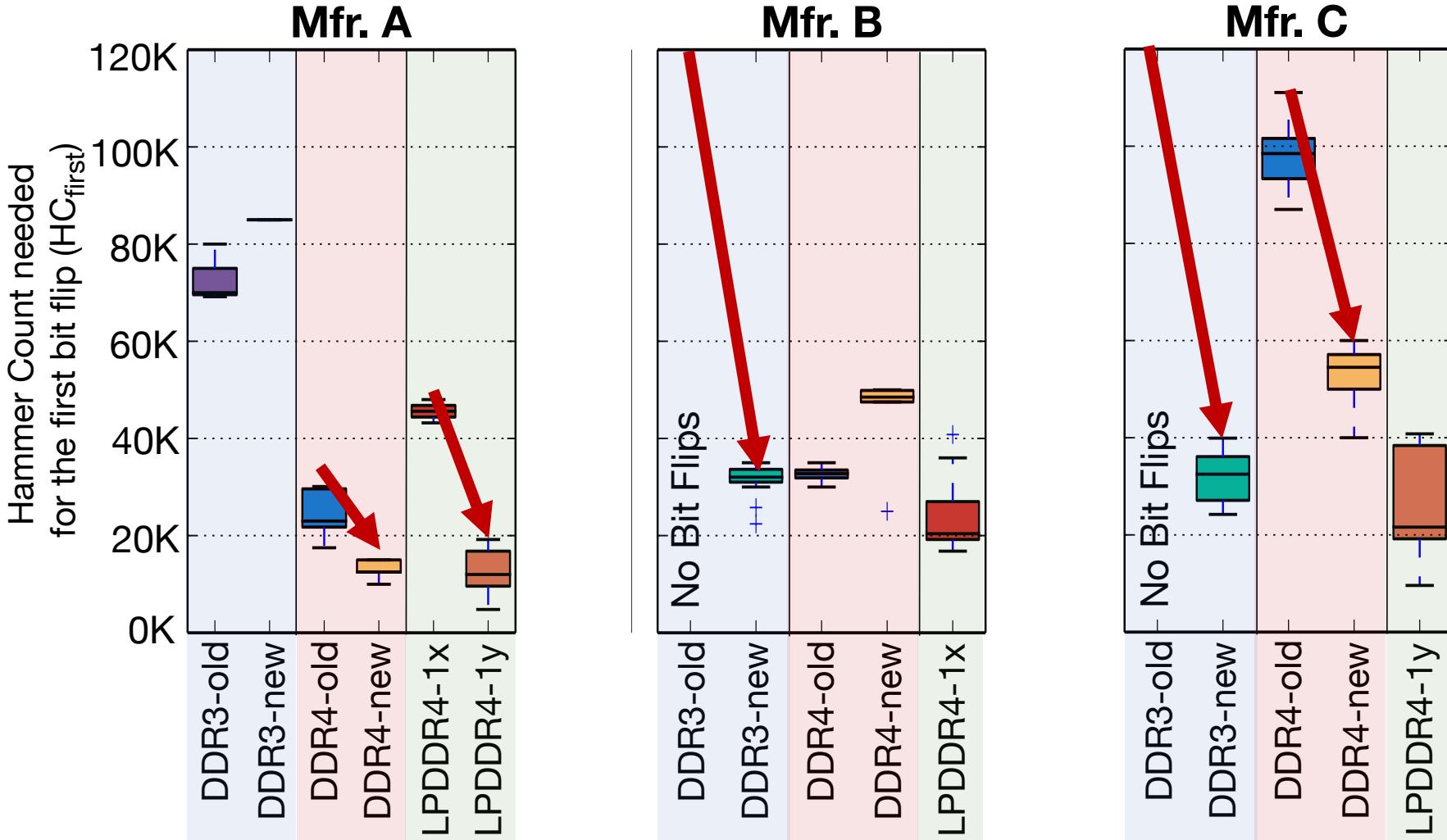
5. First RowHammer Bit Flips per Chip

What is the minimum Hammer Count required to cause bit flips (HC_{first})?

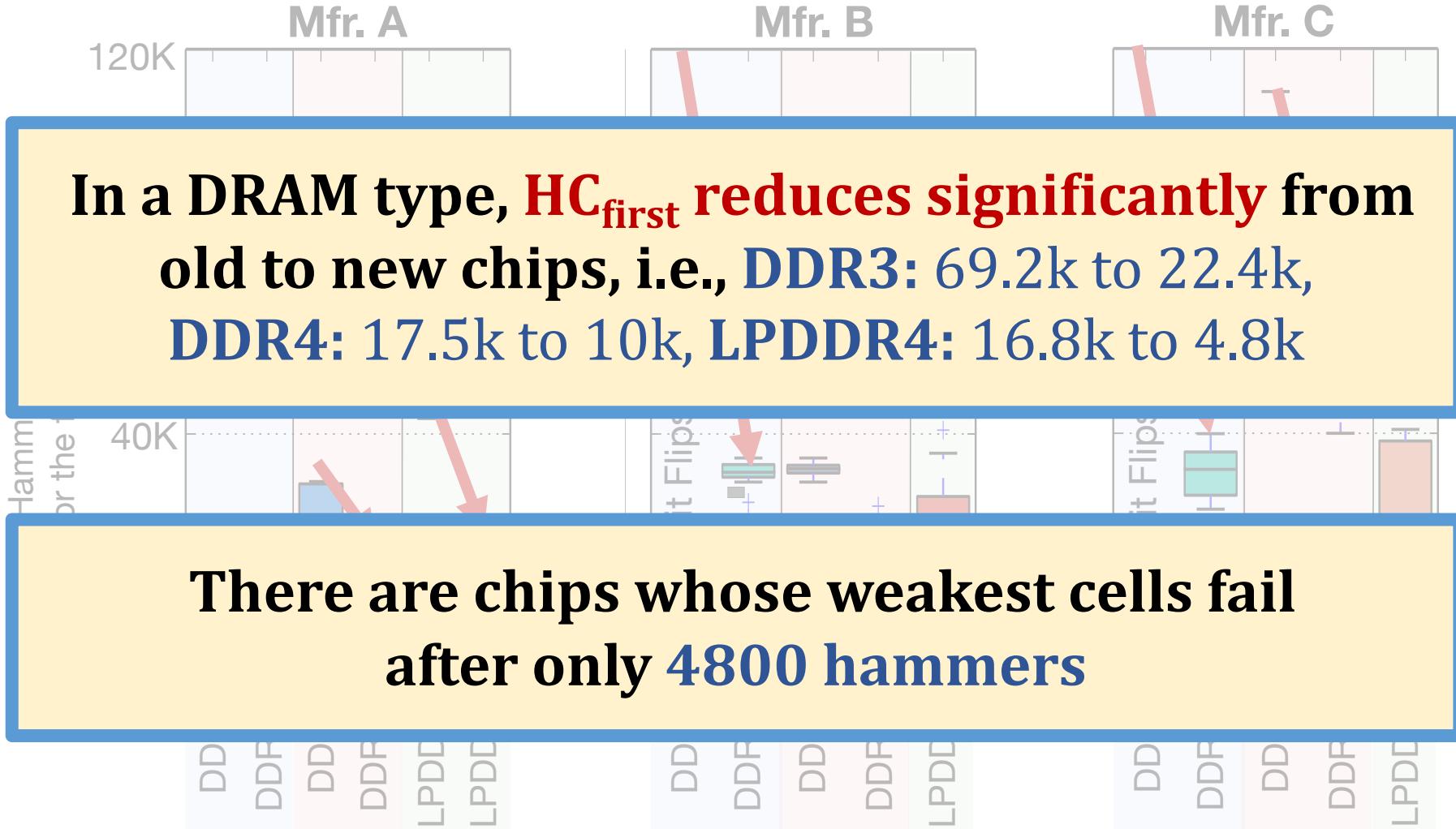We note the different DRAM types on the x-axis: **DDR3**, **DDR4**, **LPDDR4.**

We focus on trends across chips of the same DRAM type to draw conclusions

SAFARI

5. First RowHammer Bit Flips per Chip

Newer chips from a given DRAM manufacturer
more vulnerable to RowHammer

5. First RowHammer Bit Flips per Chip

In a DRAM type, HC_{first} reduces significantly from old to new chips, i.e., DDR3: 69.2k to 22.4k, **DDR4:** 17.5k to 10k, **LPDDR4:** 16.8k to 4.8k

There are chips whose weakest cells fail after only 4800 hammers

Newer chips from a given DRAM manufacturer **more** vulnerable to RowHammer

Key Takeaways from 1580 Chips

- Chips of newer DRAM technology nodes are **more vulnerable** to RowHammer

- There are chips today whose weakest cells fail after **only 4800 hammers**

- Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in **more rows** and 2) **farther away** from the victim row.

SAFARI

Outline

SAFARI

Evaluation Methodology

- **Cycle-level simulator:** Ramulator [Kim+, CAL'15]
  https://github.com/CMU-SAFARI/ramulator
  - 4GHz, 4-wide, 128 entry instruction window
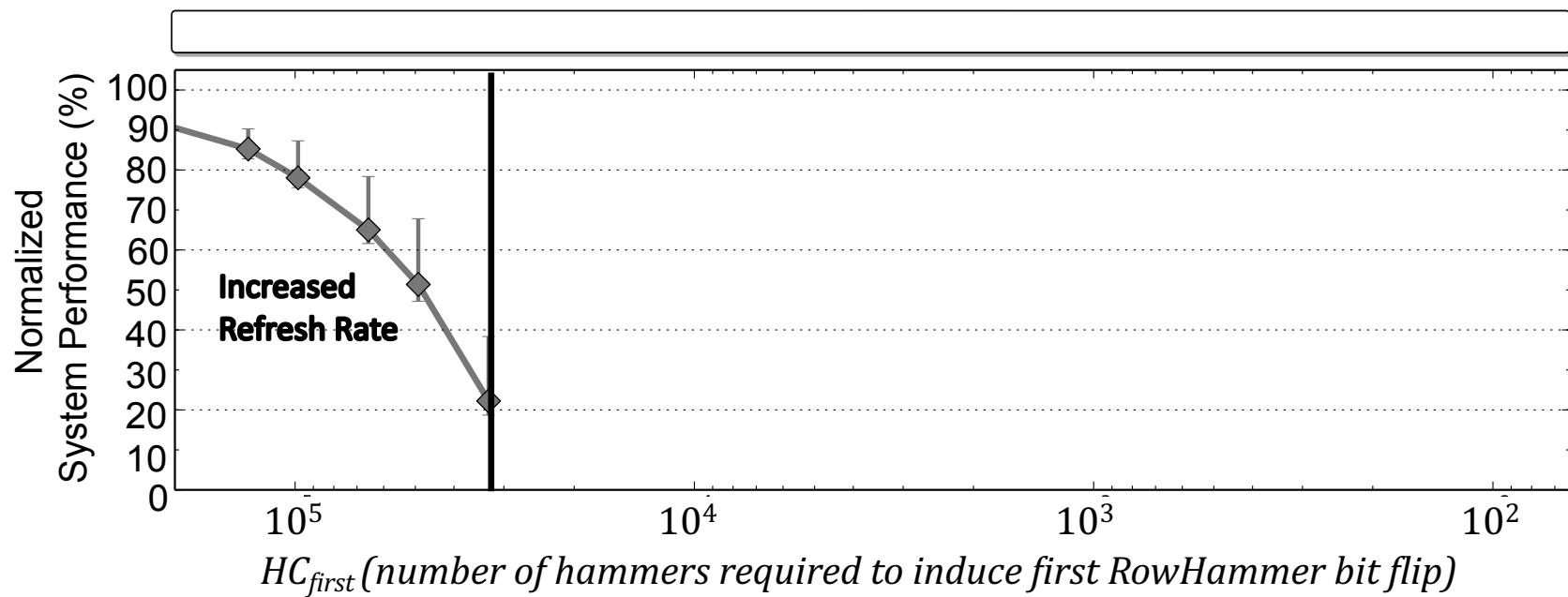 - 48 8-core workload mixes randomly drawn from SPEC CPU2006 **(10 < MPKI < 740)**

- **Metrics to evaluate mitigation mechanisms**
  1. *DRAM Bandwidth Overhead:* fraction of total system DRAM bandwidth consumption from mitigation mechanism
  2. *Normalized System Performance:* normalized weighted speedup to a 100% baseline

Evaluation Methodology

- We evaluate **five** state-of-the-art mitigation mechanisms:
  - **Increased Refresh Rate** **[Kim+, ISCA'14]**
  - **PARA** **[Kim+, ISCA'14]**
  - **ProHIT** **[Son+, DAC'17]**
  - **MRLoc** **[You+, DAC'19]**
  - **TWiCe** **[Lee+, ISCA'19]**

- and **one** ideal refresh-based mitigation mechanism:
  - **Ideal**

- **More detailed descriptions in the paper on:**
  - Descriptions of mechanisms in our paper and the original publications
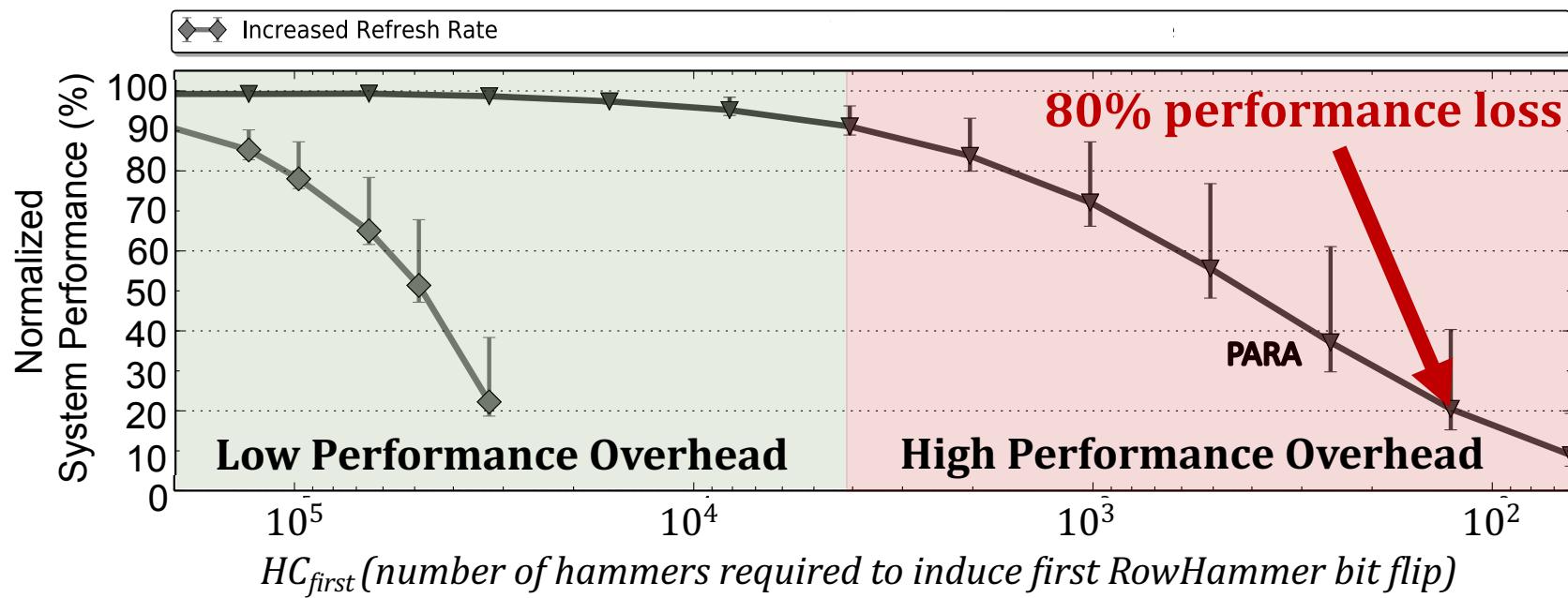 - How we scale each mechanism to more vulnerable DRAM chips (lower HC_{first})

SAFARI

Normalized System Performance (%)

Increased Refresh Rate

100
90
80
70
60
50
40
30
20
10
0

10^5 10^4 10^3 10^2
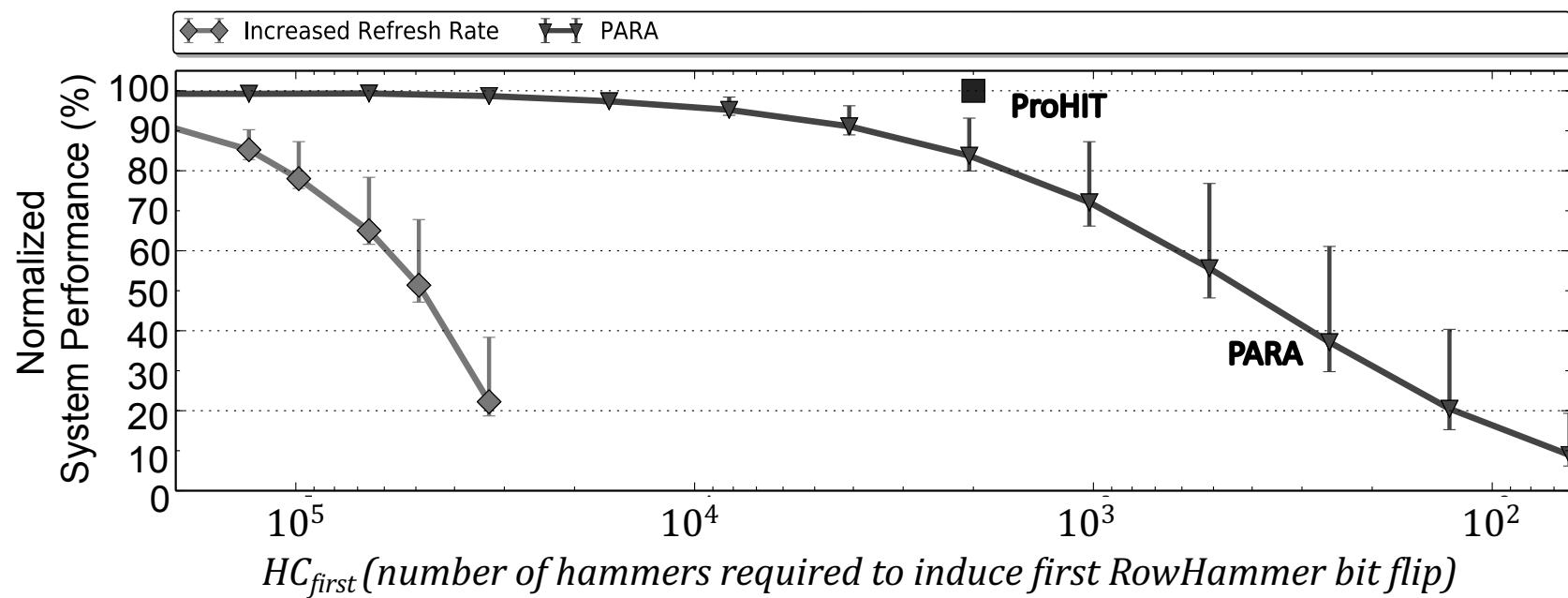
Substantial overhead for high HC_{first} values.

This mechanism does not support HC_{first} < 32k
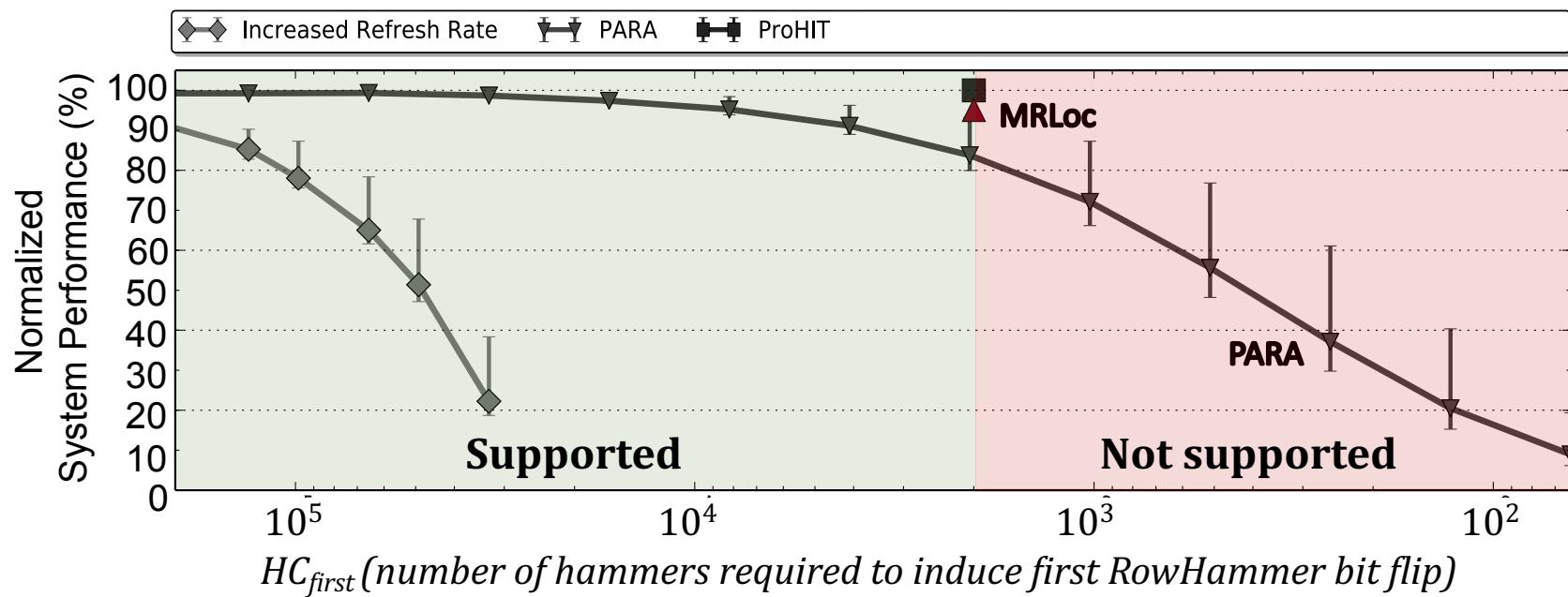due to the **prohibitively high refresh rates** required

Mitigation Mechanism Evaluation (PARA)

Mitigation Mechanism Evaluation (ProHIT)

Legend: Increased Refresh Rate, PARA

Y-axis: Normalized System Performance (%)

X-axis: HC_{first} (number of hammers required to induce first RowHammer bit flip)

ProHIT

PARA
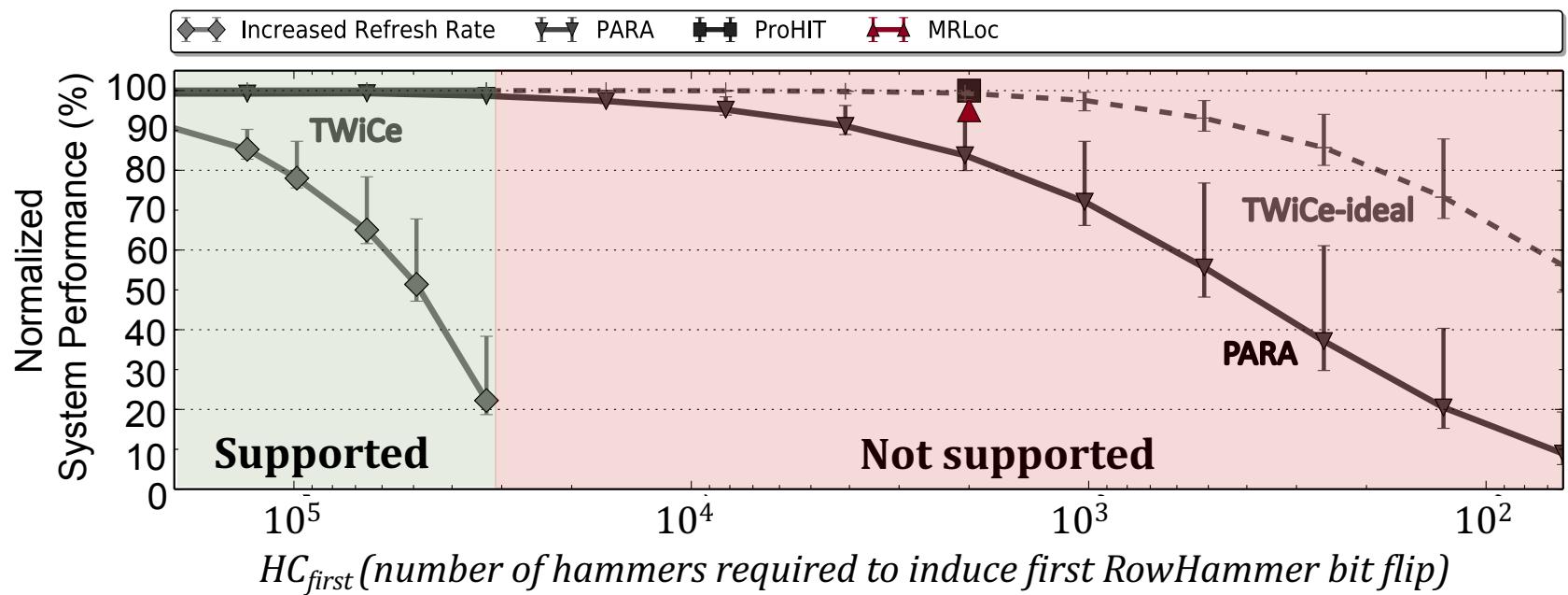
Mitigation Mechanism Evaluation (MRLoc)

**Models for scaling ProHIT and MRLoc for $HC_{first} < 2k$
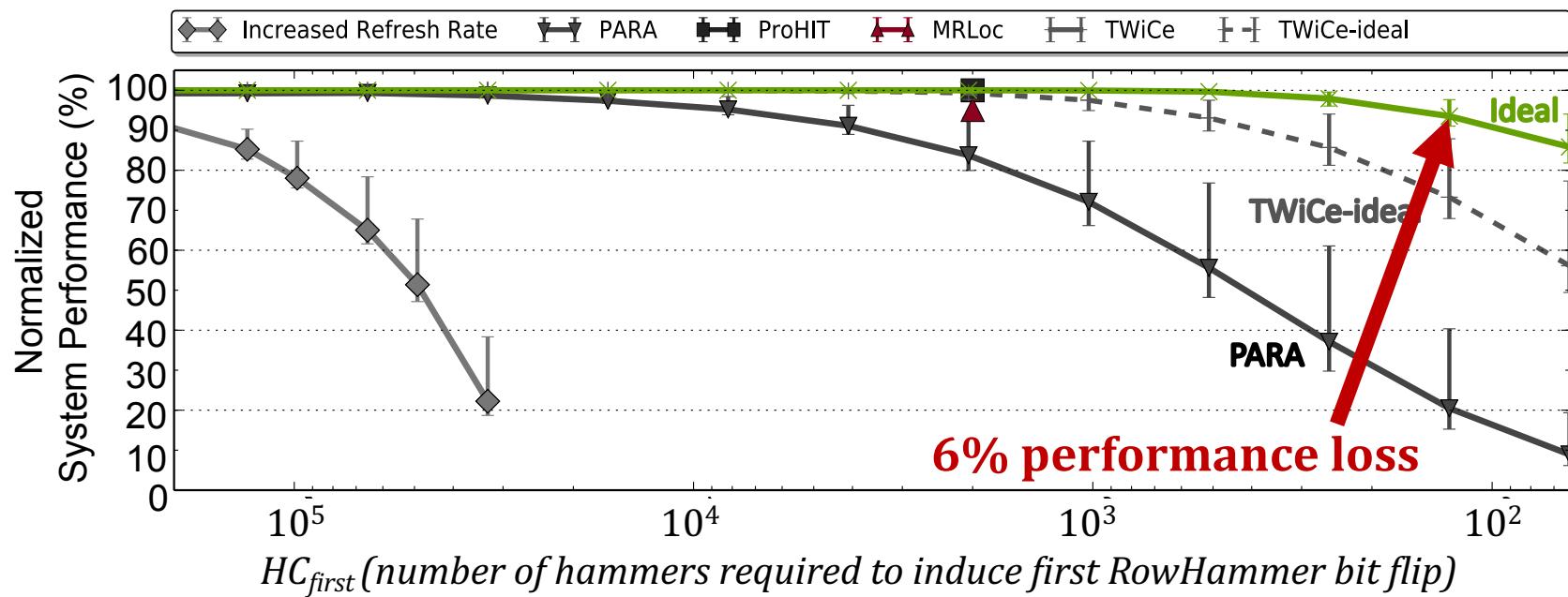are not provided and how to do so is not intuitive**

SAFARI

Mitigation Mechanism Evaluation (TWiCe)

Legend: Increased Refresh Rate — PARA — ProHIT — MRLoc

Y-axis: Normalized System Performance (%)

X-axis: HC_{first} (number of hammers required to induce first RowHammer bit flip)

Labels: TWiCe, Supported, Not supported, TWiCe-ideal, PARA

TWiCe does not support $HC_{first} < 32k$.

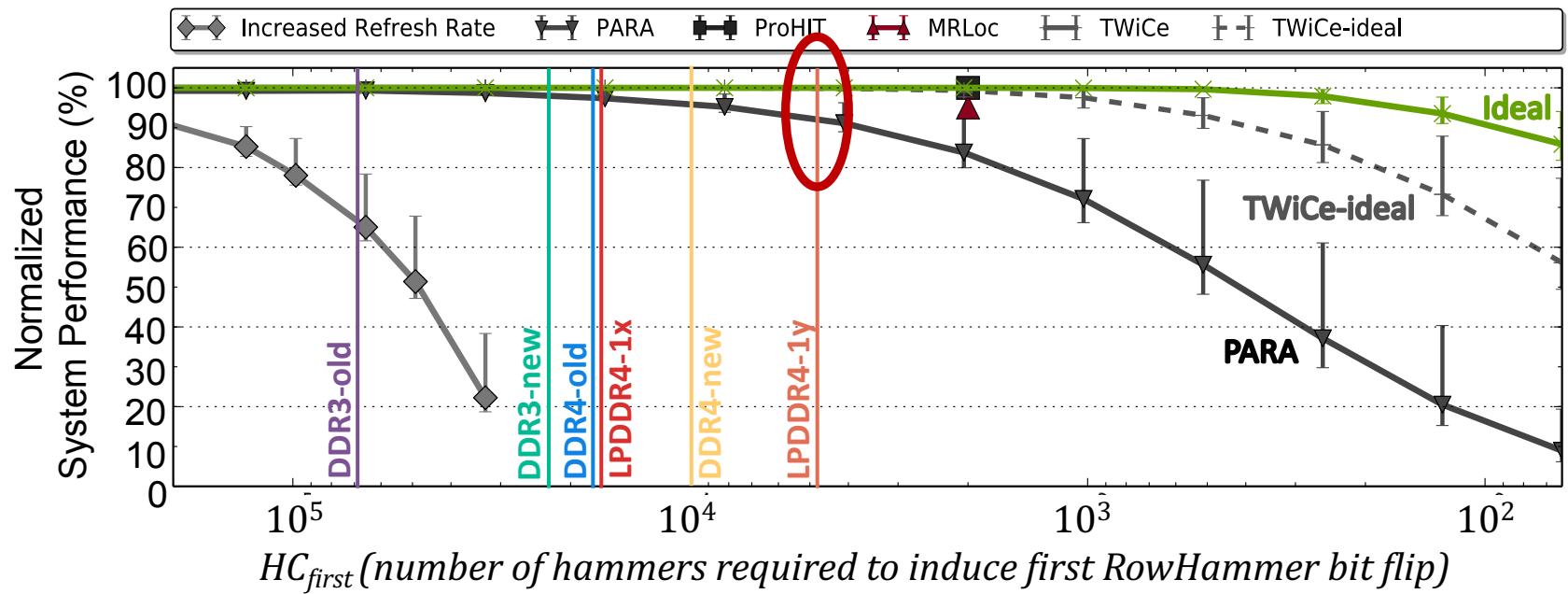**We evaluate an ideal scalable version (TWiCe-ideal) assuming it solves two critical design issues**

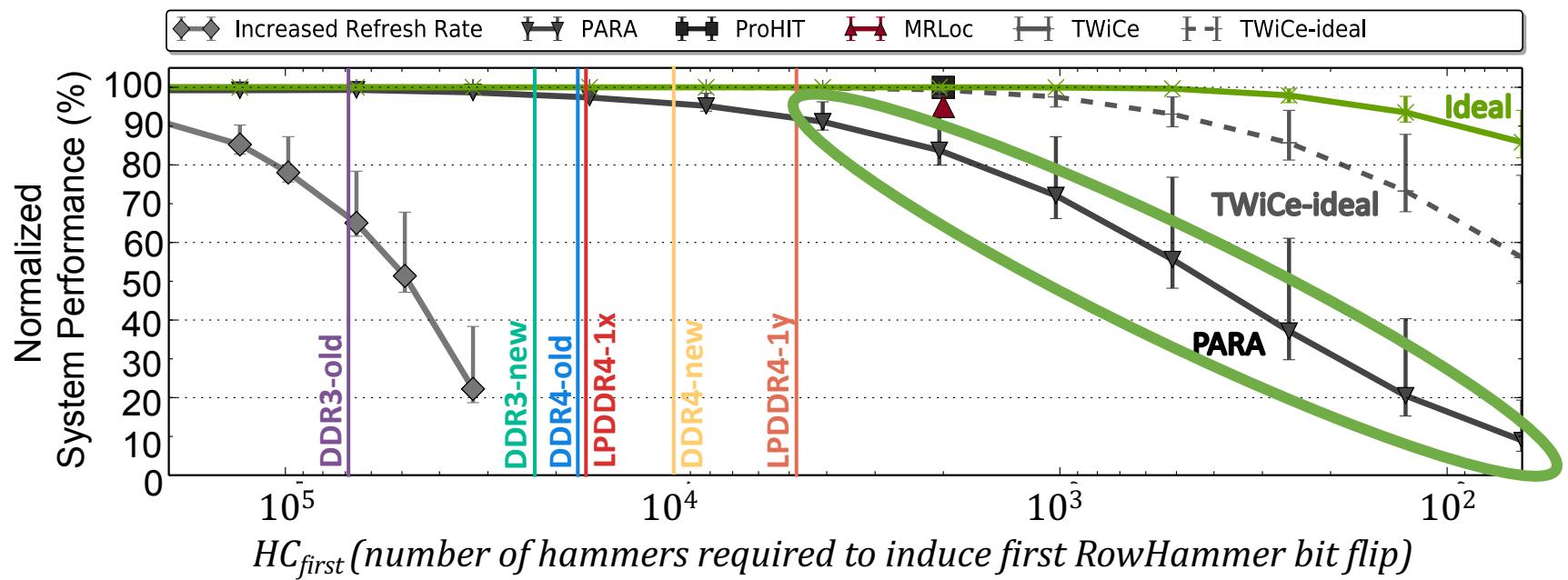# Mitigation Mechanism Evaluation (Ideal)

6% performance loss

Ideal mechanism issues a refresh command
to a row only right before the row
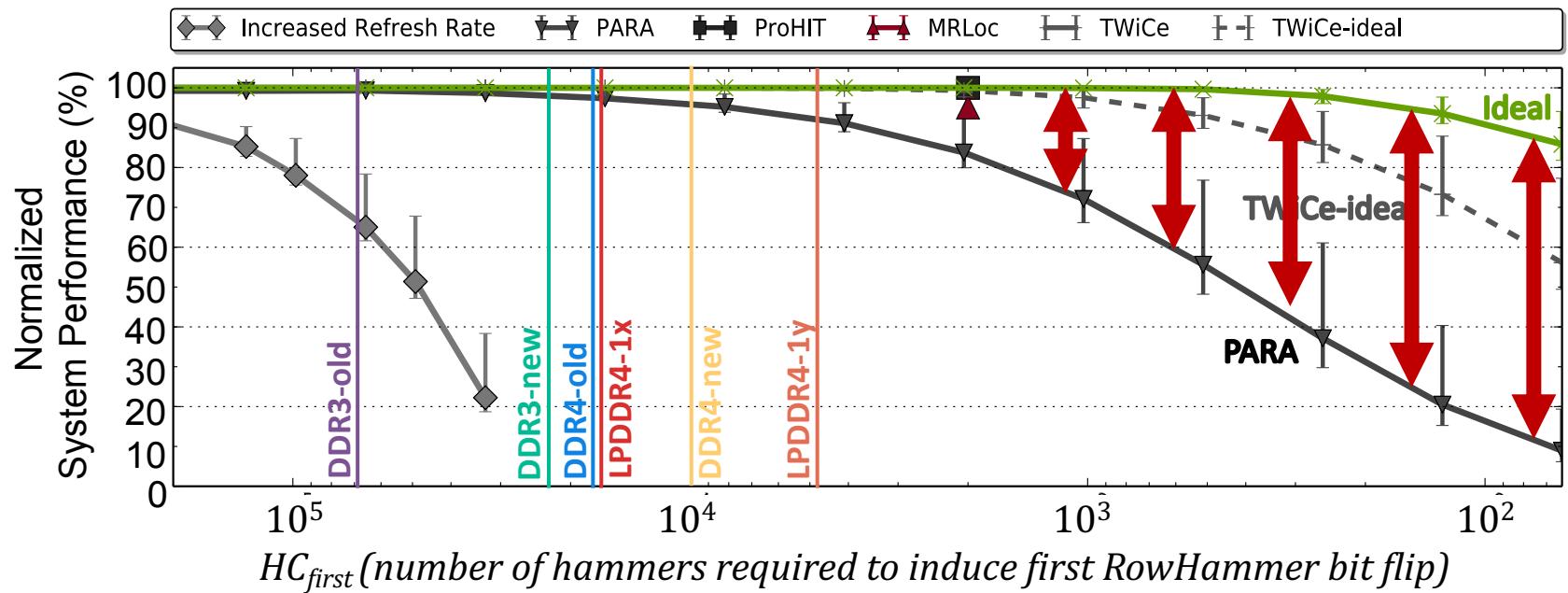can potentially experience a RowHammer bit flip

Mitigation Mechanism Evaluation

Legend: Increased Refresh Rate, PARA, ProHIT, MRLoc, TWiCe, TWiCe-ideal

Y-axis: Normalized System Performance (%)

X-axis: HC_{first} (number of hammers required to induce first RowHammer bit flip)

Vertical markers: DDR3-old, DDR3-new, DDR4-old, LPDDR4-1x, DDR4-new, LPDDR4-1y

Curve labels: Ideal, TWiCe-ideal, PARA

PARA, ProHIT, and MRLoc mitigate RowHammer bit flips
in **worst chips** today with reasonable system performance
(92%, 100%, 100%)

Mitigation Mechanism Evaluation

**Only PARA's design scales to low HC$_{first}$ values
but has very low normalized system performance**

SAFARI

Mitigation Mechanism Evaluation

Ideal mechanism is **significantly better**
than any existing mechanism for $HC_{first} < 1024$

Significant opportunity for developing a RowHammer solution
with **low performance overhead that supports low HC_{first}**

Key Takeaways from Mitigation Mechanisms

- Existing RowHammer mitigation mechanisms can prevent RowHammer attacks with **reasonable system performance overhead** in DRAM chips today

- Existing RowHammer mitigation mechanisms **do not scale well** to DRAM chips more vulnerable to RowHammer

- There is still **significant opportunity** for developing a mechanism that is **scalable with low overhead**

Additional Details in the Paper

- **Single-cell RowHammer bit flip probability**

- More details on our **data pattern dependence** study

- Analysis of **Error Correcting Codes (ECC)** in mitigating RowHammer bit flips

- Additional **observations** on our data

- **Methodology details** for characterizing DRAM

- Further discussion on comparing data across different infrastructures

- **Discussion on scaling** each mitigation mechanism

Outline

RowHammer Introduction

DRAM Background

Motivation and Goal

Experimental Methodology

Characterization Results

Evaluation of Mitigation Mechanisms

RowHammer Solutions Going Forward

Conclusion

RowHammer Solutions Going Forward

Two promising directions for new RowHammer solutions:

1. DRAM-system cooperation

- We believe the DRAM and system should cooperate more to provide a **holistic** solution can prevent RowHammer at **low cost**

2. Profile-guided

- Accurate **profile of RowHammer-susceptible cells** in DRAM provides a powerful substrate for building **targeted** RowHammer solutions, e.g.:
  - Only increase the refresh rate for rows containing RowHammer-susceptible cells

- A **fast and accurate** profiling mechanism is a key research challenge for developing low-overhead and scalable RowHammer solutions

SAFARI

Outline

RowHammer Introduction

DRAM Background

Motivation and Goal

Experimental Methodology

Characterization Results

Evaluation of Mitigation Mechanisms

RowHammer Solutions Going Forward

Conclusion

SAFARI

Conclusion

- We characterized **1580 DRAM** chips of different DRAM types, technology nodes, and manufacturers.

- We studied **five** state-of-the-art RowHammer mitigation mechanisms and an ideal refresh-based mechanism

- We made **two key observations**

  1. **RowHammer is getting much worse**. It takes much fewer hammers to induce RowHammer bit flips in newer chips
     - e.g., **DDR3:** 69.2k to 22.4k, **DDR4:** 17.5k to 10k, **LPDDR4:** 16.8k to 4.8k
  2. **Existing mitigation mechanisms do not scale** to DRAM chips that are more vulnerable to RowHammer
     - e.g., 80% performance loss when the hammer count to induce the first bit flip is 128

- We **conclude** that it is **critical** to do more research on RowHammer and develop scalable mitigation mechanisms to prevent RowHammer in future systems

SAFARI

Revisiting RowHammer

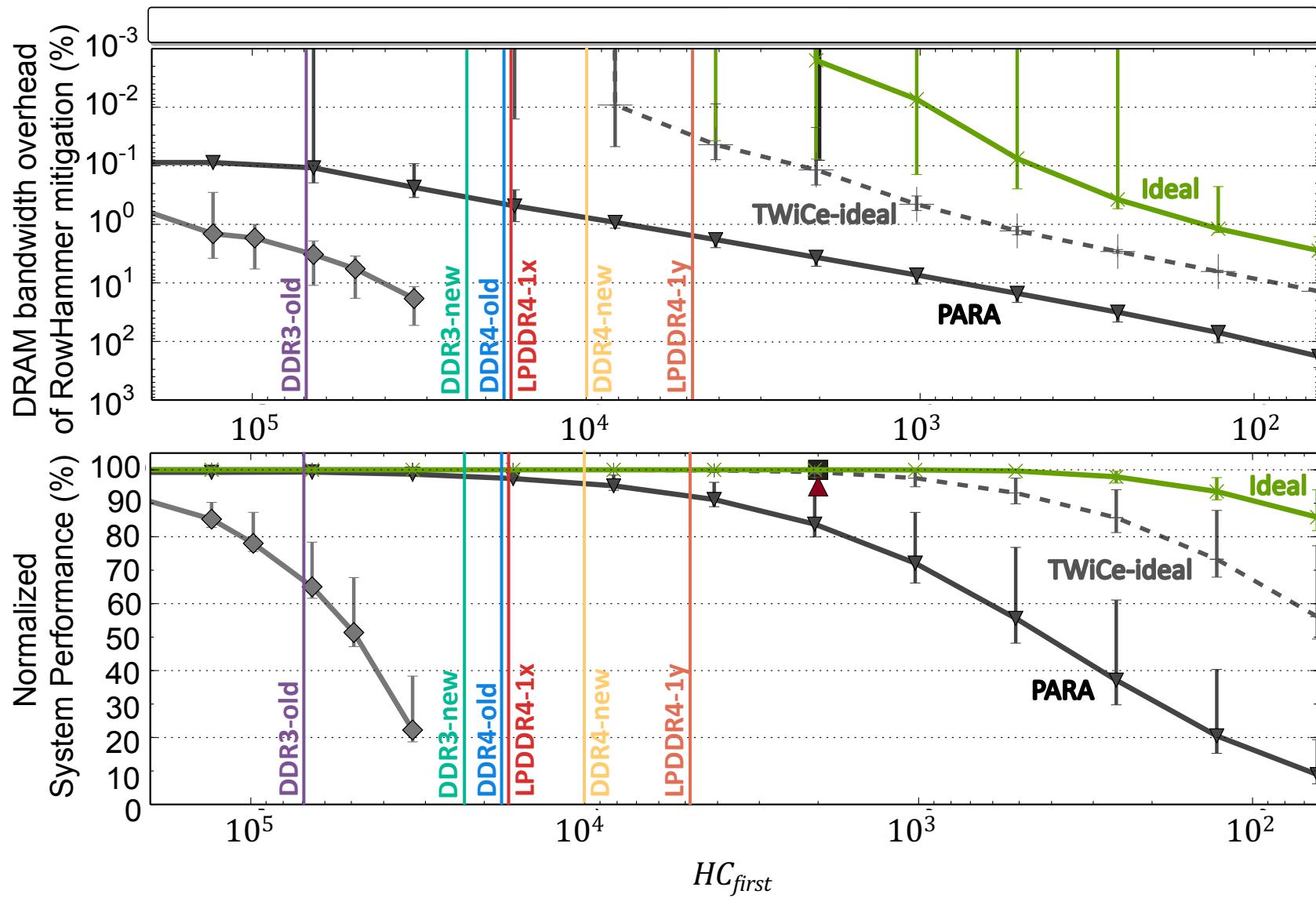## *An Experimental Analysis of Modern Devices and Mitigation Techniques*

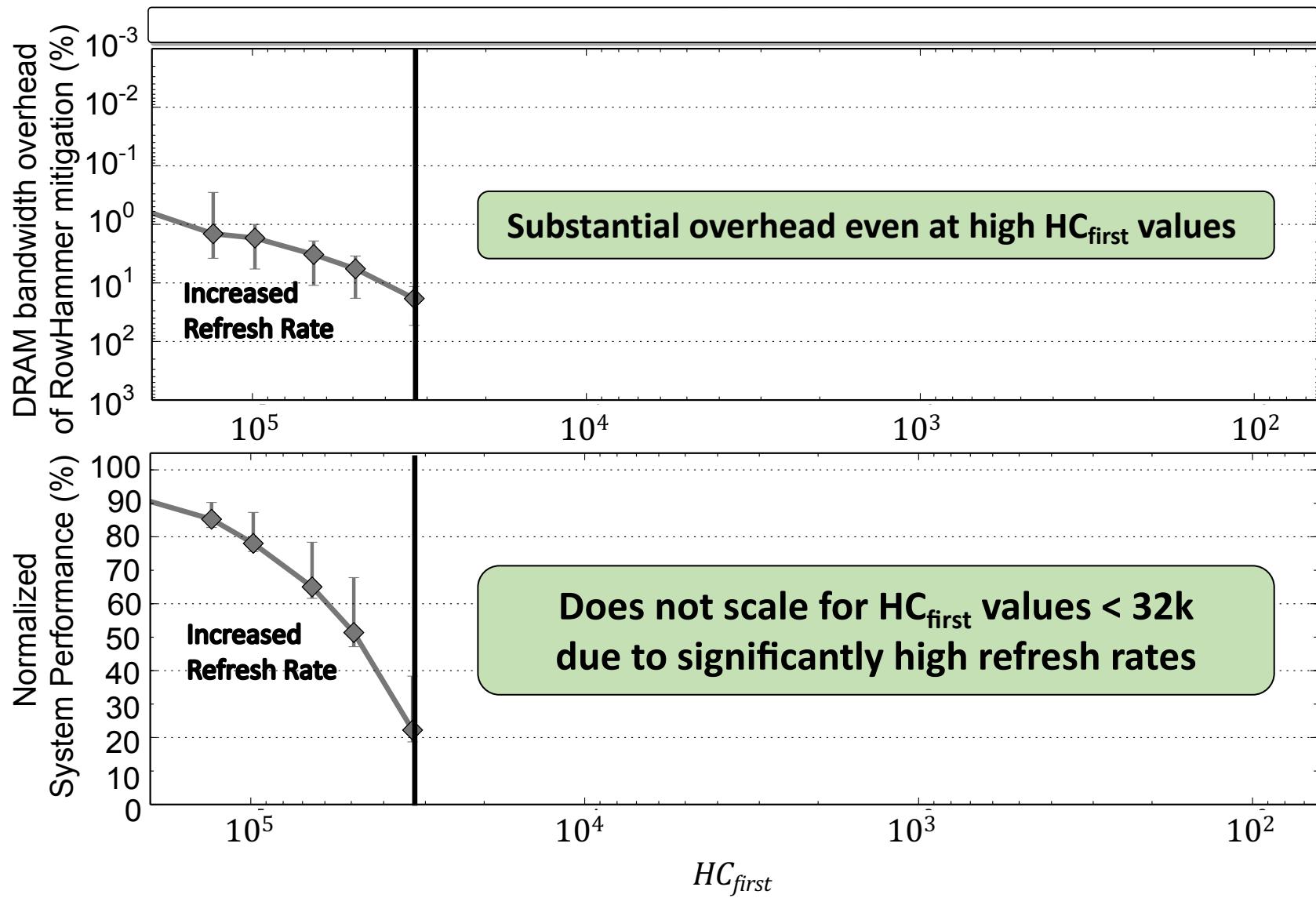**Jeremie S. Kim**   **Minesh Patel**

A. Giray Yağlıkçı   **Hasan Hassan**
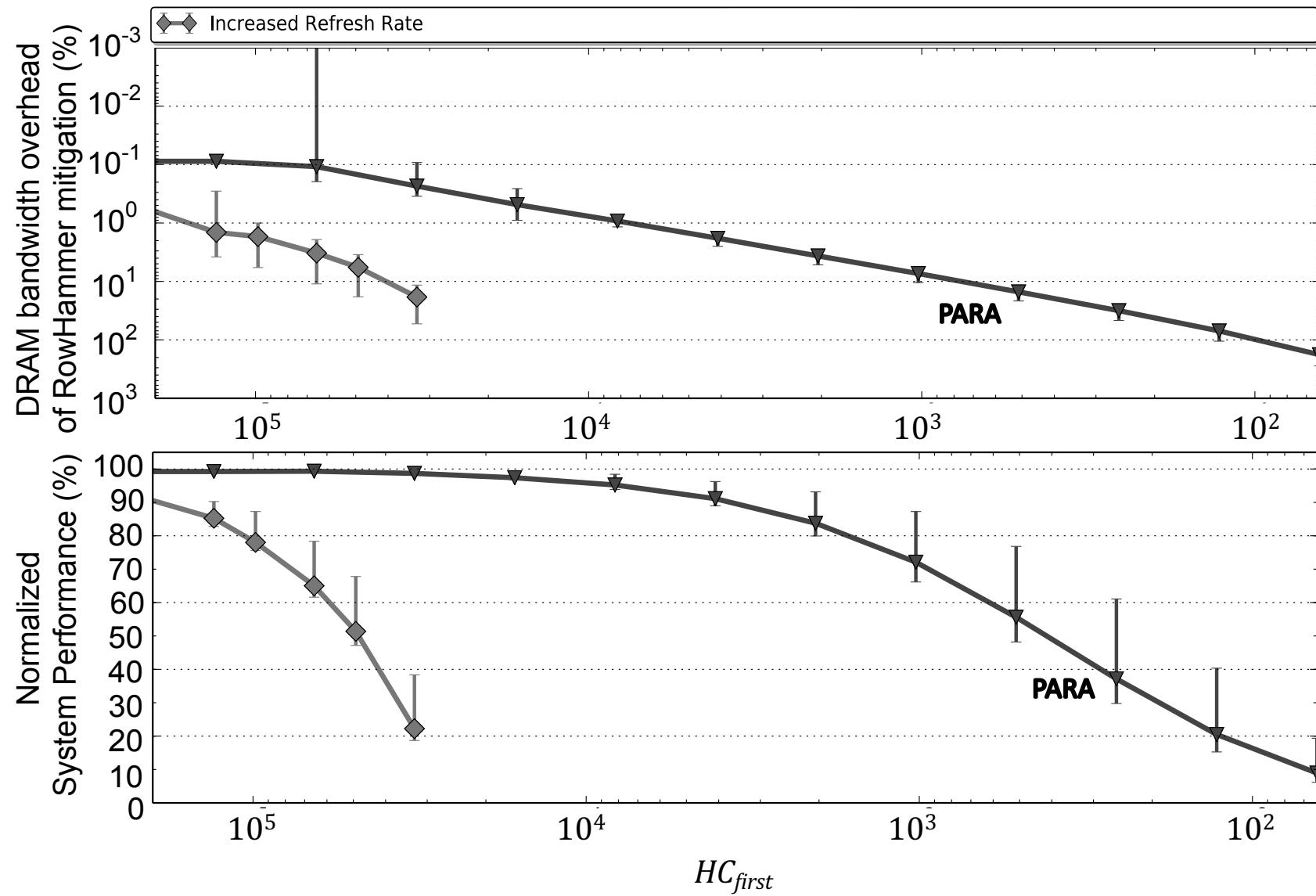
Roknoddin Azizi   **Lois Orosa**   **Onur Mutlu**

SAFARI

ETH Zürich

Carnegie Mellon

Evaluation

SAFARI

Mitigation Mechanism Evaluation

Substantial overhead even at high HC_{first} values

Does not scale for HC_{first} values < 32k
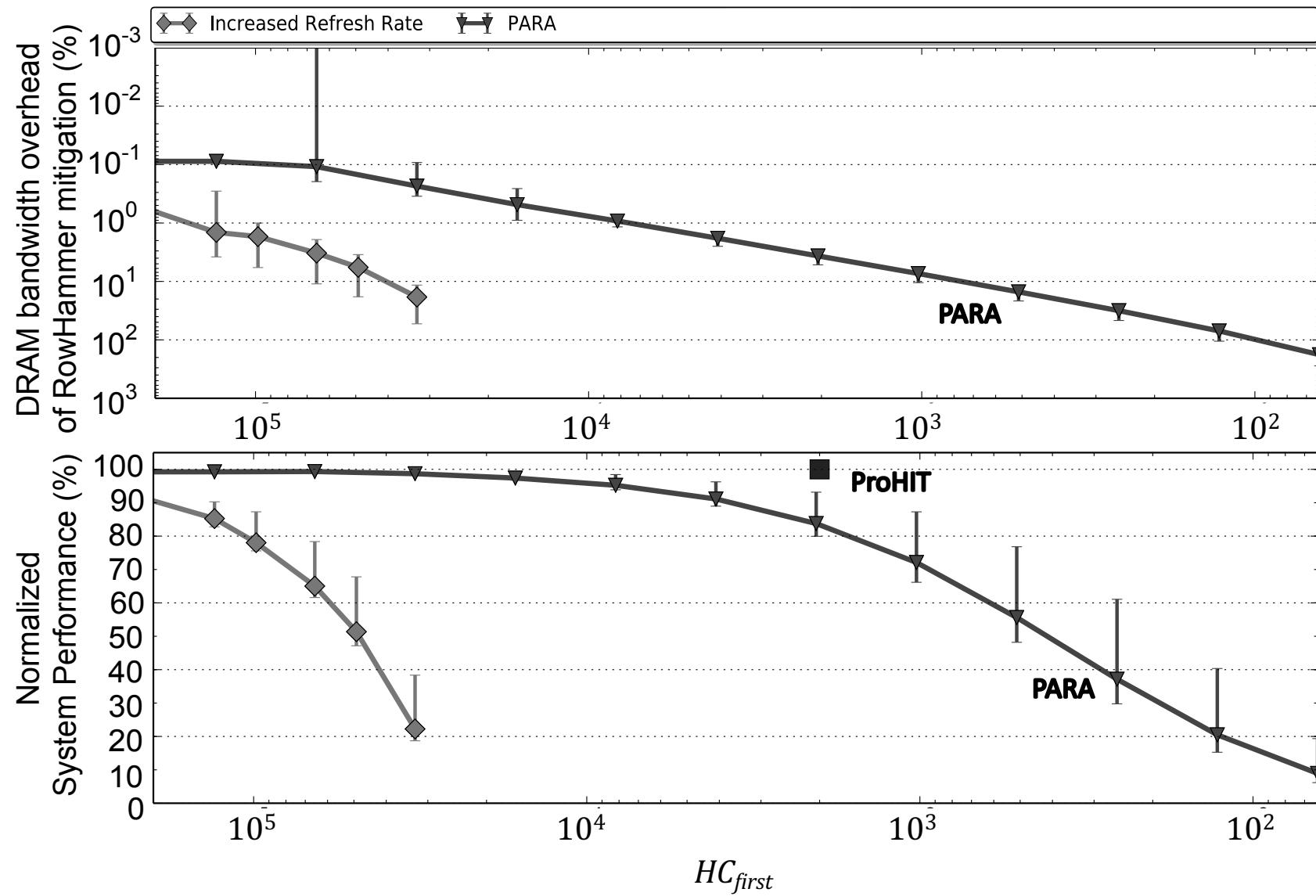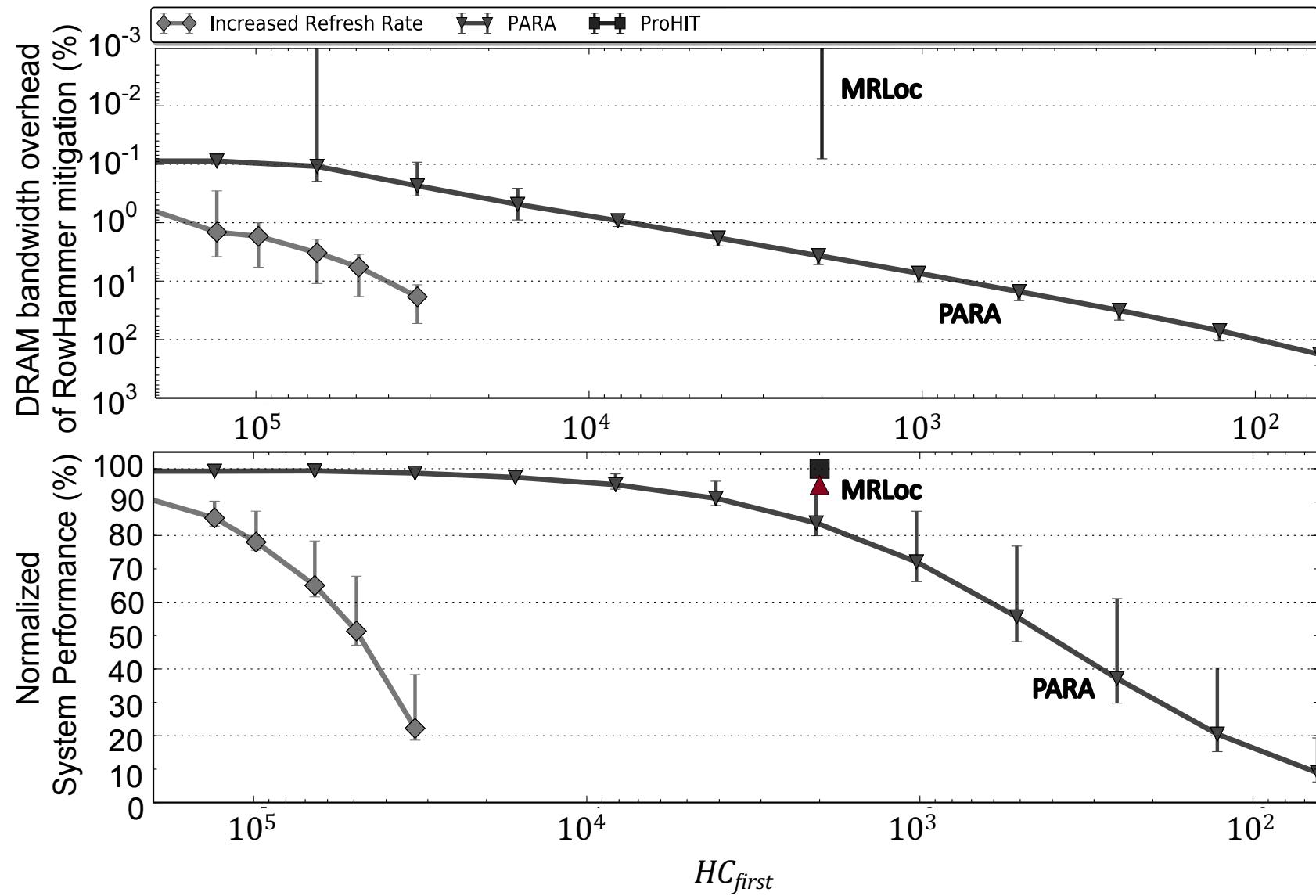due to significantly high refresh rates

SAFARI

Mitigation Mechanism Evaluation

Mitigation Mechanism Evaluation

SAFARI

Mitigation Mechanism Evaluation

TWiCe's current form does not support $HC_{first} < 32k$, but we evaluate an ideal version (TWiCe-ideal) assuming it solves two critical design issues
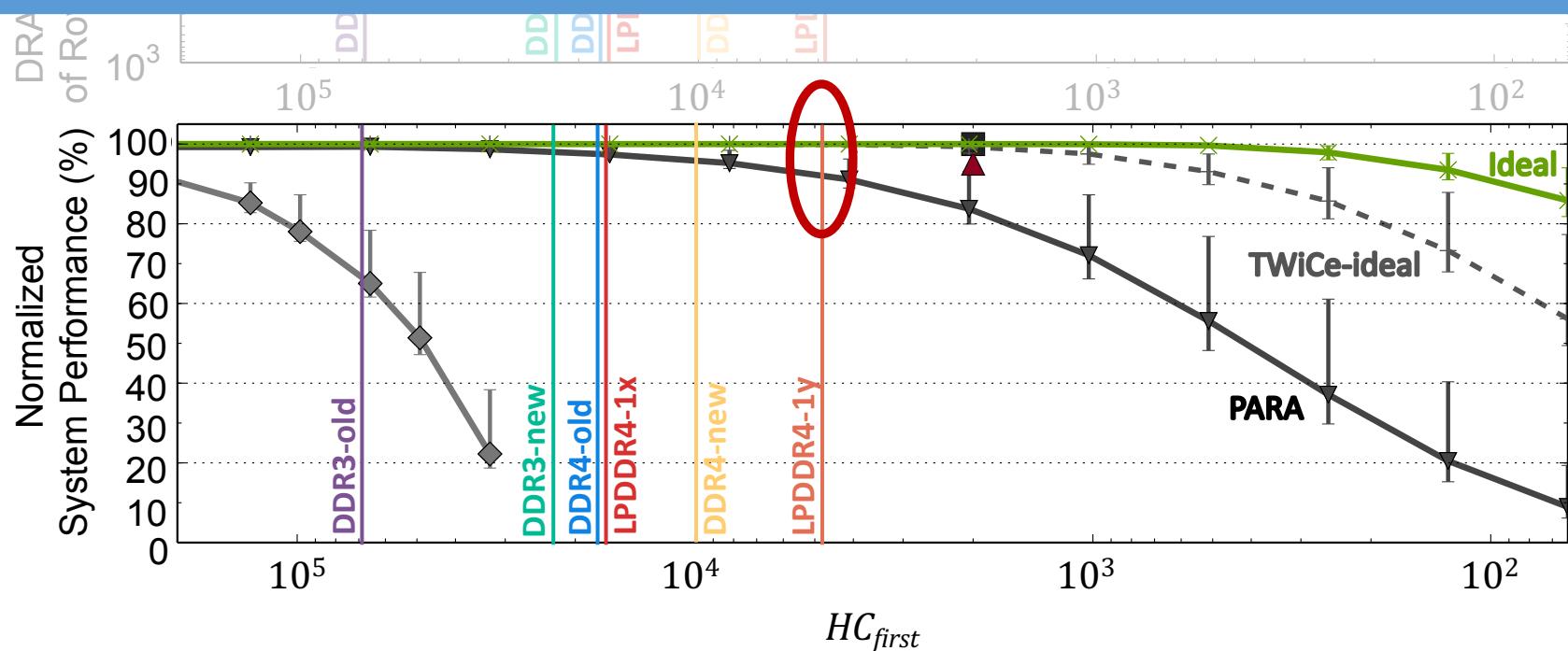
SAFARI

Mitigation Mechanism Evaluation

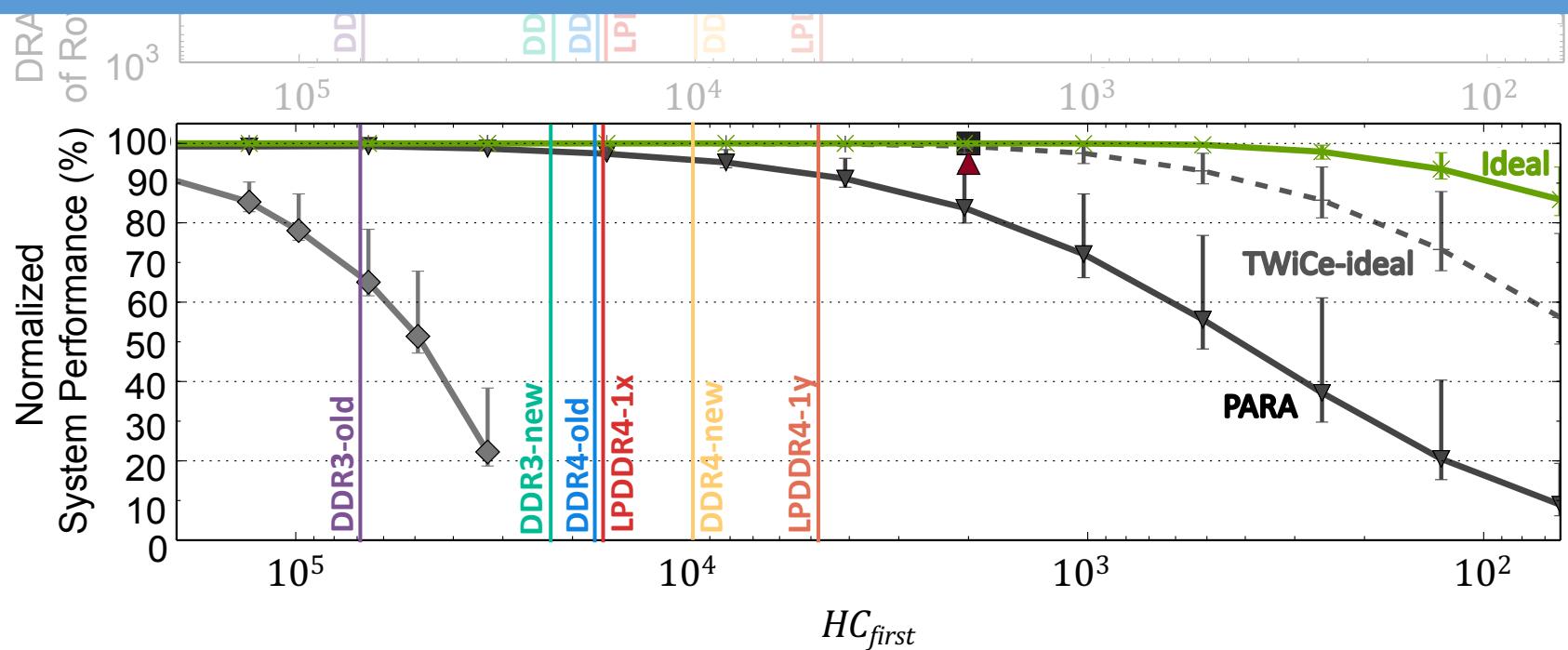**SAFARI**

Mitigation Mechanism Evaluation

PARA, ProHIT, and MRLoc are viable options
for mitigating RowHammer bit flips in **worst chips** today
with reasonable system performance **(92%, 100%, 100%)**

Legend: Increased Refresh Rate, PARA, ProHIT, MRLoc, TWiCe, TWiCe-ideal, Ideal

Y-axis: Normalized System Performance (%)
X-axis: HC_{first}

SAFARI

Mitigation Mechanism Evaluation

Only PARA's design scales to low HC$_{first}$ values
that we may see in future DRAM chips but
has very low normalized system performance

Mitigation Mechanism Evaluation

The ideal refresh-based mitigation mechanism
is significantly better than any existing mechanism
as HC_{first} reduces below 1024

This indicates significant opportunity for developing
a RowHammer solution with low performance overhead
that also scales to low HC_{first} values

SAFARI

Effective RowHammer Characterization

To characterize our DRAM chips at **worst-case** conditions, we:

1. **Prevent sources of interference during core test loop**
   - **We disable**: DRAM refresh, DRAM calibration events, RowHammer mitigation mechanisms
   - Ensure **test shorter than refresh window** (i.e., 32ms) to prevent retention failures
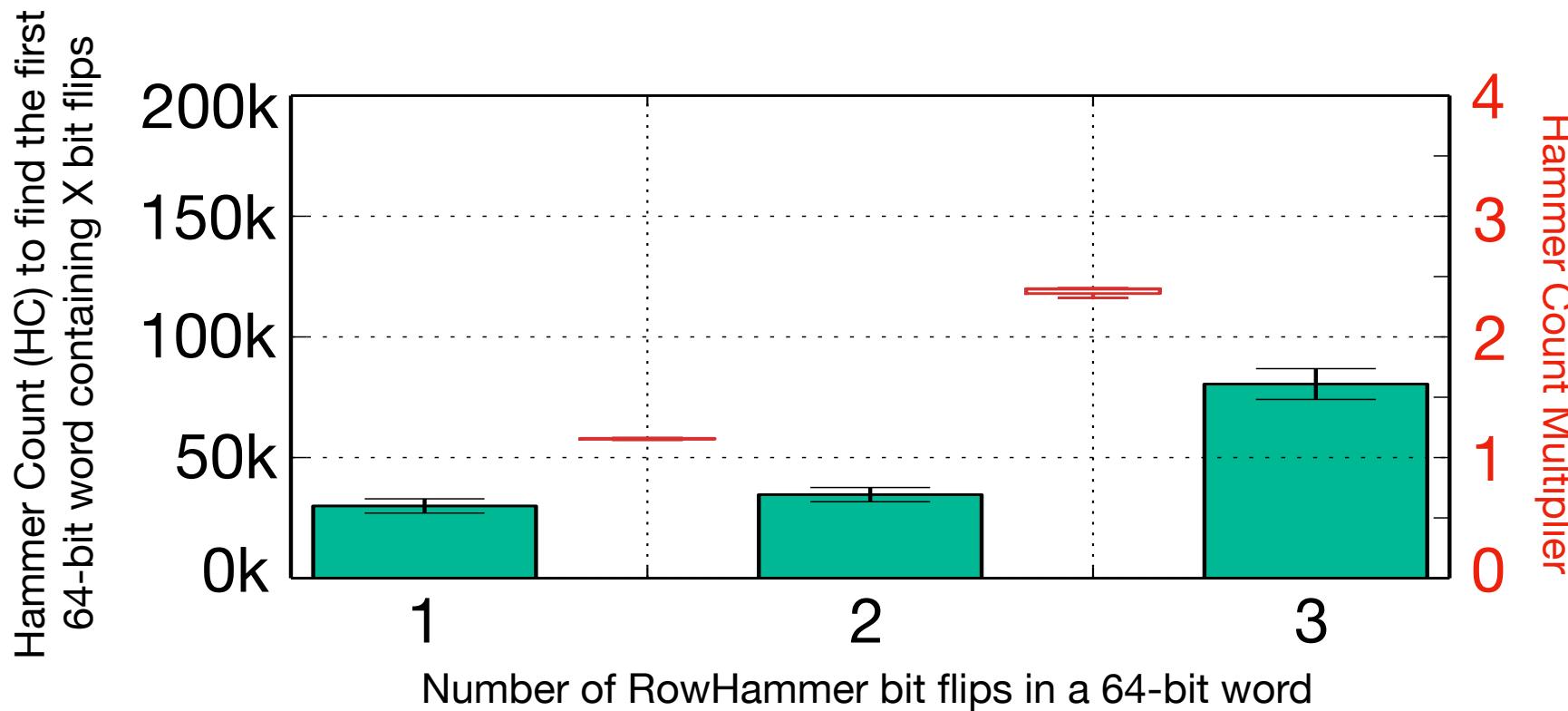
2. **Worst-case access sequence**

   We construct based on three observations from prior work:

   1. An aggressor row causes the most RowHammer bit flips in immediately **adjacent** rows
   2. A **double-sided hammer** targeting victim row N (i.e., repeatedly accessing rows N+1 and N-1) causes the most bit flips in row N compared to other access patterns
   3. **Increasing the rate of DRAM activations** results in more RowHammer bit flips
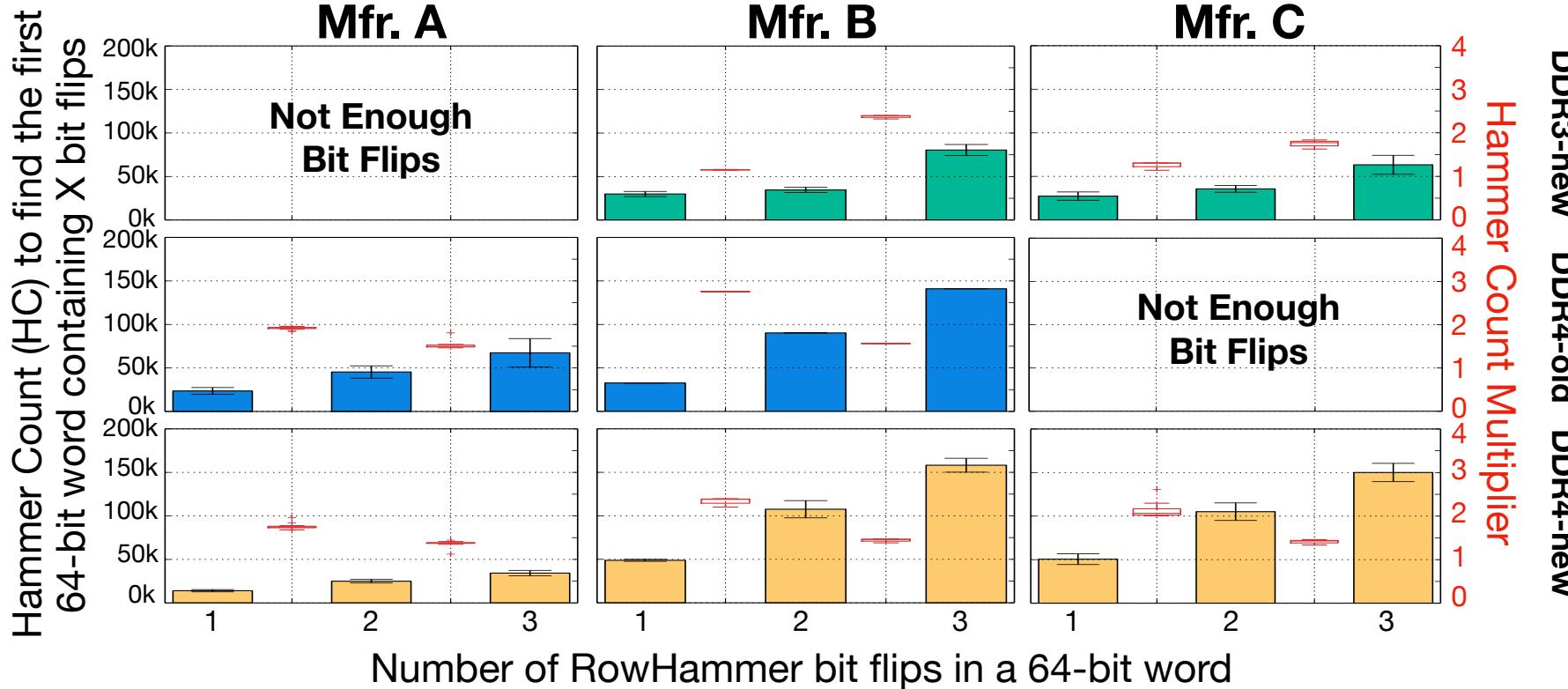
   Using these observations, we test each row's worst-case vulnerability to RowHammer by **repeatedly accessing the two directly physically-adjacent rows as fast as possible**

6. Error-Correcting Code (ECC) Effects

Q. How would different Error Correction Codes (ECC) change the Hammer Count required to cause RowHammer bit flips?

6. Error-Correcting Code (ECC) Effects

Single-error correcting code can improve HC$_{first}$ by up to 2.78× in DDR4 DRAM chips, and 1.65× in DDR3-new DRAM chips.

RowHammer Solutions Going Forward

Two promising directions for new RowHammer solutions:

1. DRAM-system cooperation

- DRAM-based or system-level mechanism **alone** ignores potential benefits of addressing the RowHammer vulnerability **holistically**
- We believe a **holistic** solution can prevent RowHammer at **low cost**

2. Profile-guided

- Accurate **profile of RowHammer-susceptible cells** in DRAM provides a powerful substrate for building **targeted** RowHammer solutions, e.g.:
  - Only increase the refresh rate for rows containing RowHammer-susceptible cells

- We believe a **fast and accurate** profiling mechanism is a key research challenge for developing low-overhead and scalable RowHammer solutions