# Computer Architecture

## Lecture 2b: Data Retention and Memory Refresh

Prof. Onur Mutlu
ETH Zürich

Fall 2020

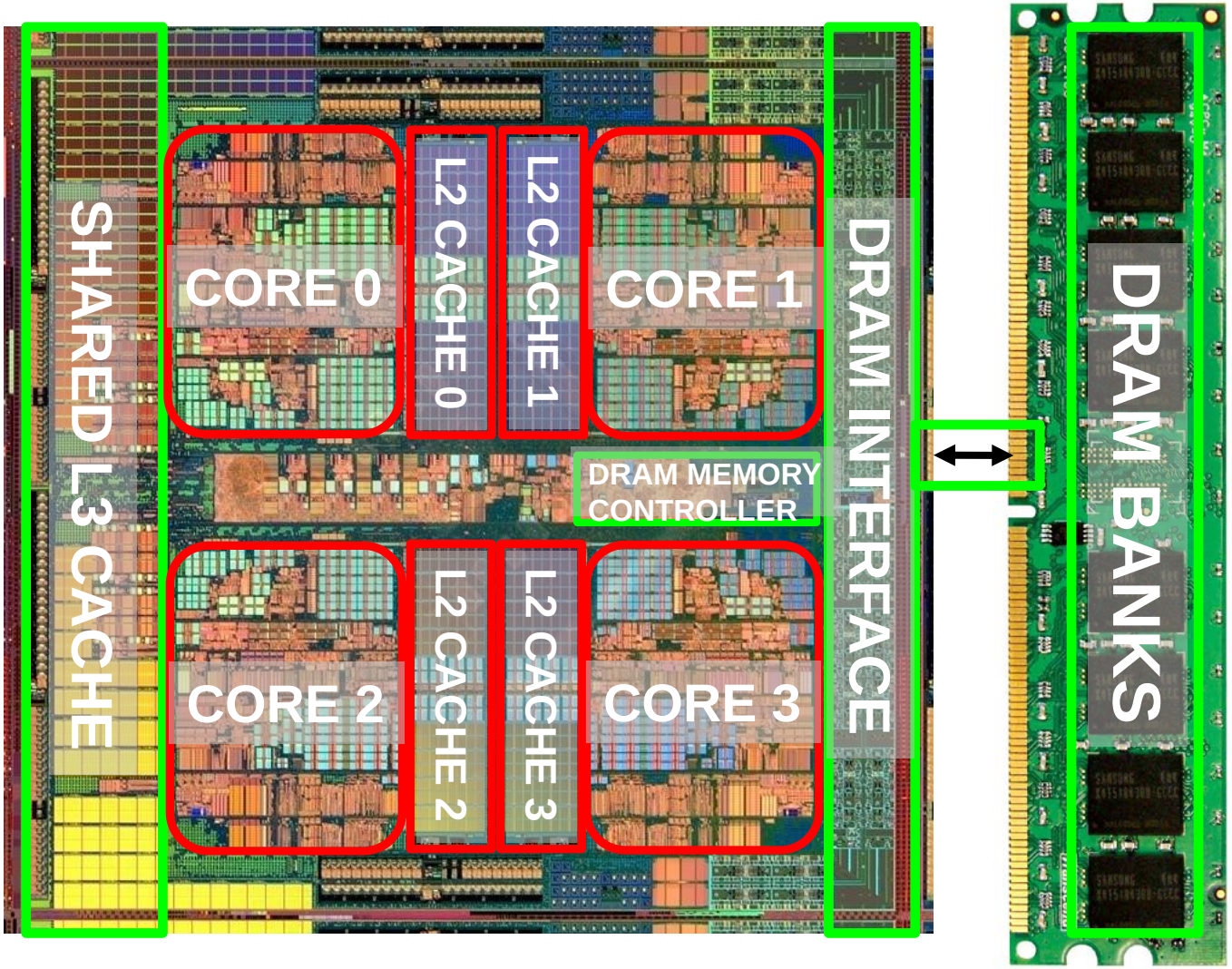
18 September 2020

# Another Example

- DRAM Refresh

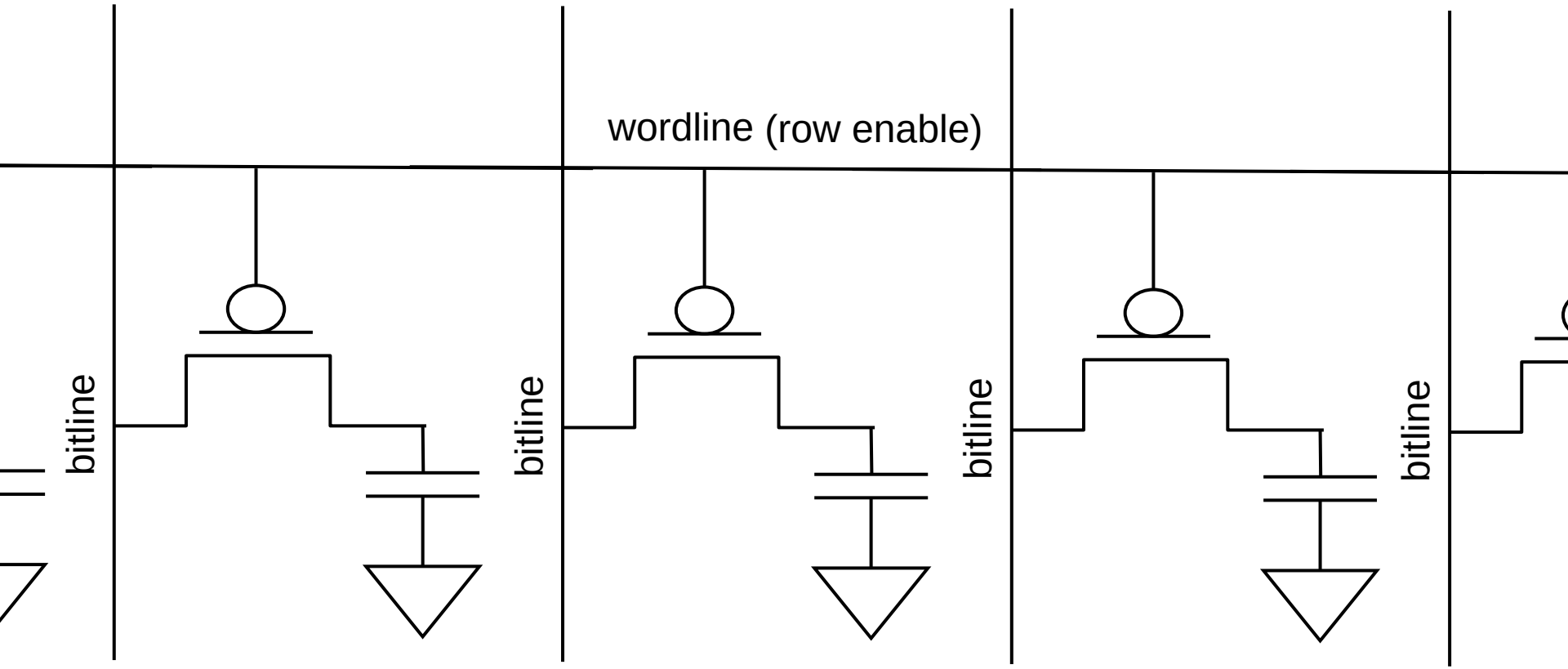
# DRAM in the System

Multi-Core Chip




*Die photo credit: AMD Barcelona

# A DRAM Cell




- A DRAM cell consists of a capacitor and an access transistor
- It stores data in terms of charge in the capacitor
- A DRAM chip consists of (10s of 1000s of) rows of such cells

# DRAM Refresh

- DRAM capacitor charge leaks over time

- The memory controller needs to refresh each row periodically to restore charge
  - Activate each row every N ms
  - Typical N = 64 ms

- Downsides of refresh

  -- Energy consumption: Each refresh consumes energy

  -- Performance degradation: DRAM rank/bank unavailable while refreshed

  -- QoS/predictability impact: (Long) pause times during refresh

  -- Refresh rate limits DRAM capacity scaling

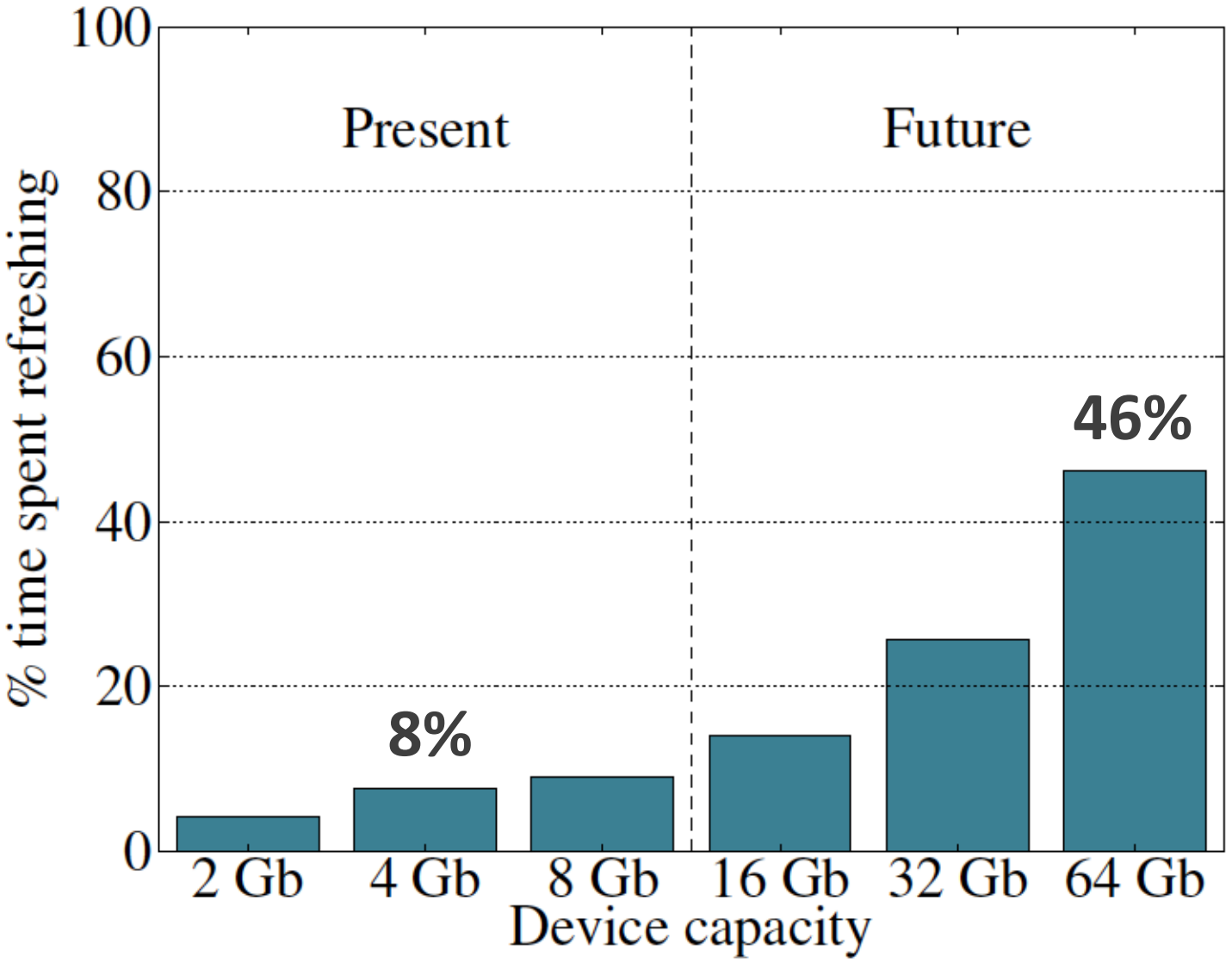
# First, Some Analysis

- Imagine a system with 8 ExaByte DRAM (2^63 bytes)
- Assume a row size of 8 KiloBytes (2^13 bytes)

- How many rows are there?
- How many refreshes happen in 64ms?
- What is the total power consumption of DRAM refresh?
- What is the total energy consumption of DRAM refresh during a day?

- A good exercise...
- Brownie points from me if you do it...

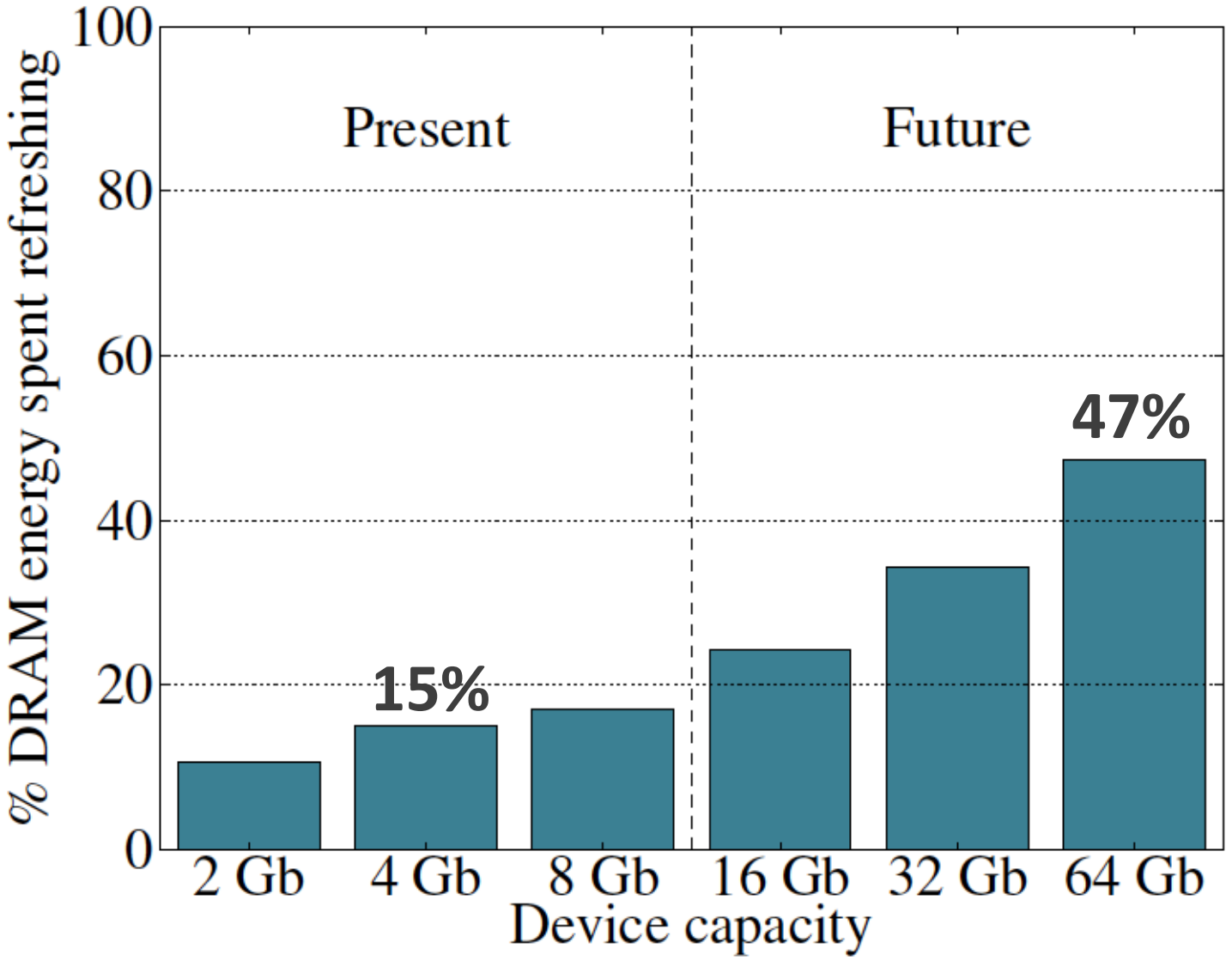
# Refresh Overhead: Performance






Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# Refresh Overhead: Energy



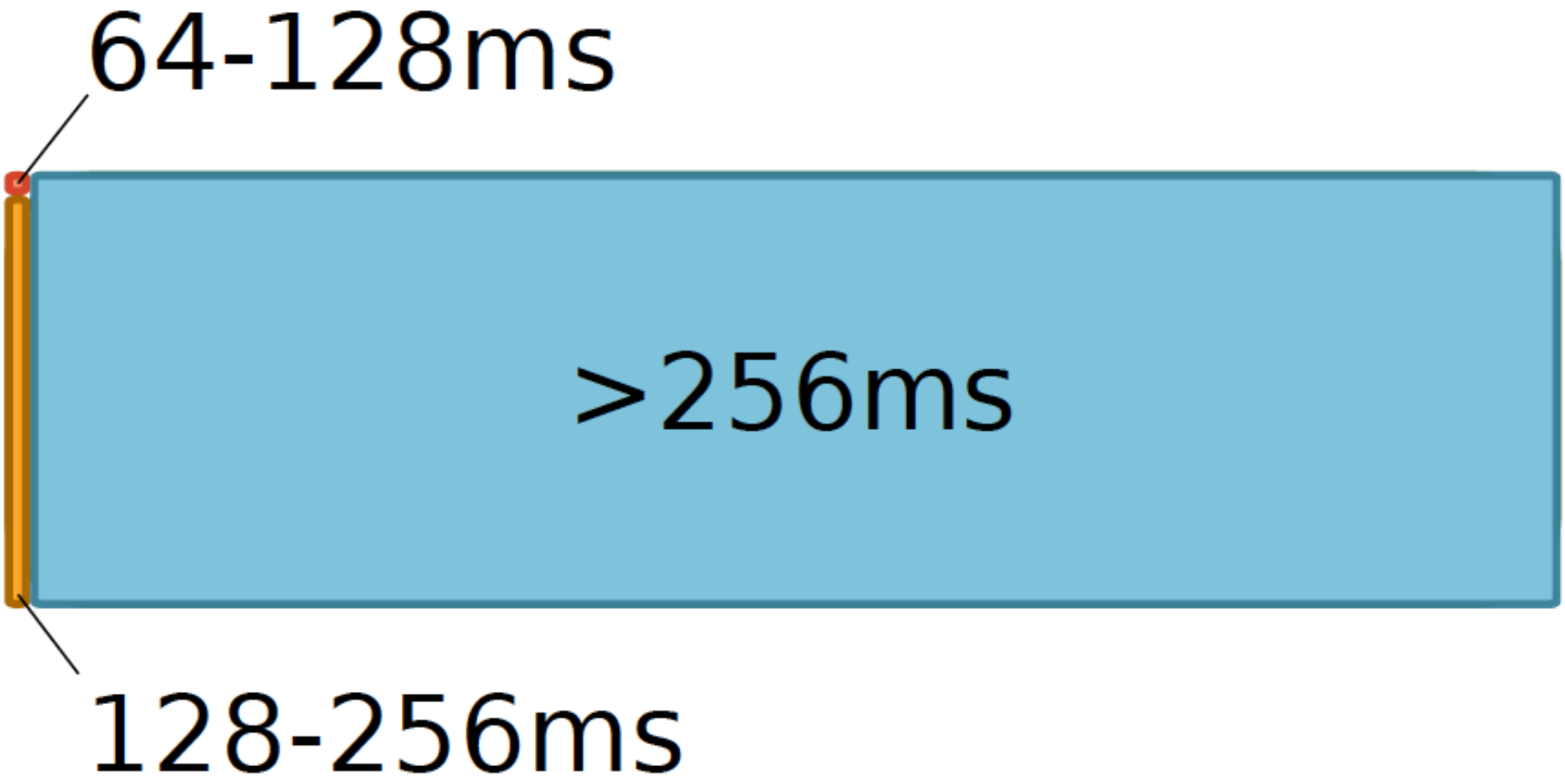



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# How Do We Solve the Problem?

- Observation: All DRAM rows are refreshed every 64ms.
- Critical thinking: Do we need to refresh all rows every 64ms?
- What if we knew what happened underneath (in DRAM cells) and exposed that information to upper layers?

# Underneath: Retention Time Profile of DRAM



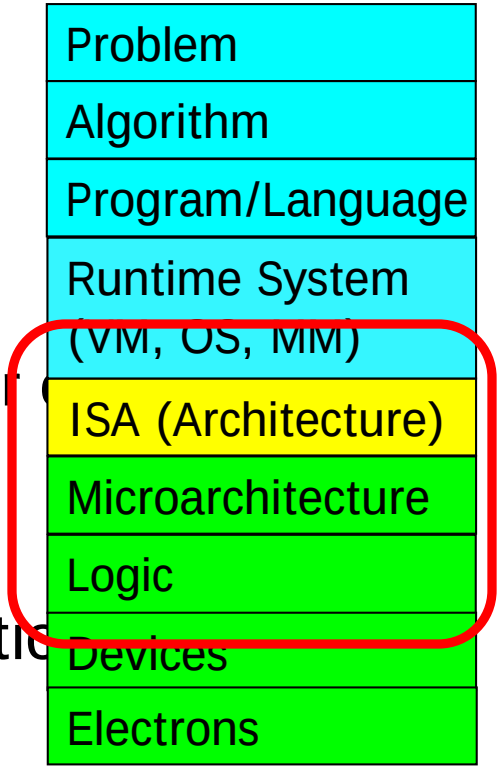

Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# Aside: Why Do We Have Such a Profile?

- Answer: Manufacturing is not perfect
- Not all DRAM cells are exactly the same
- Some are more leaky than others
- This is called Manufacturing Process Variation

# Opportunity: Taking Advantage of This Profile

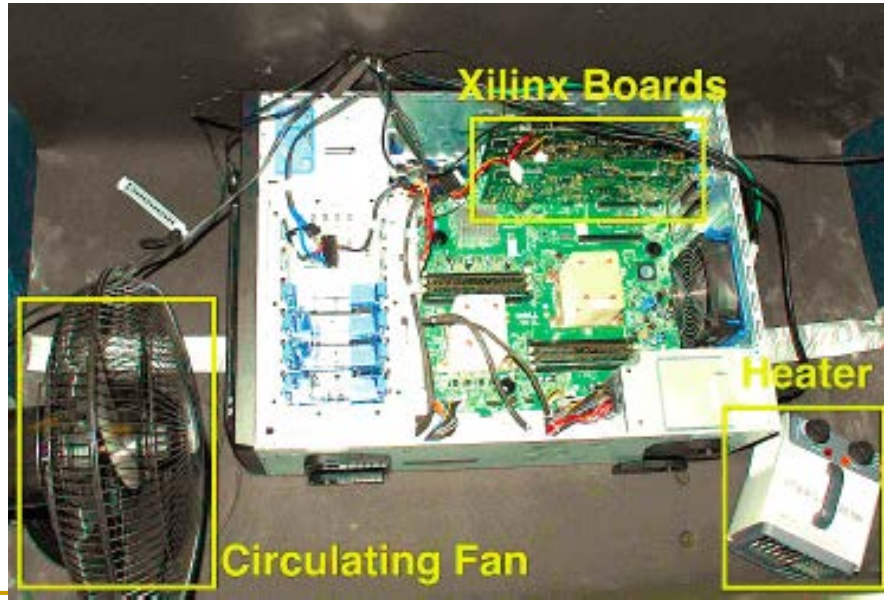
- Assume we know the retention time of each row exactly
- What can we do with this information?
- Who do we expose this information to?
- How much information do we expose?
  - Affects hardware/software overhead, power
    verification complexity, cost
- How do we determine this profile informatio
  - Also, who determines it?



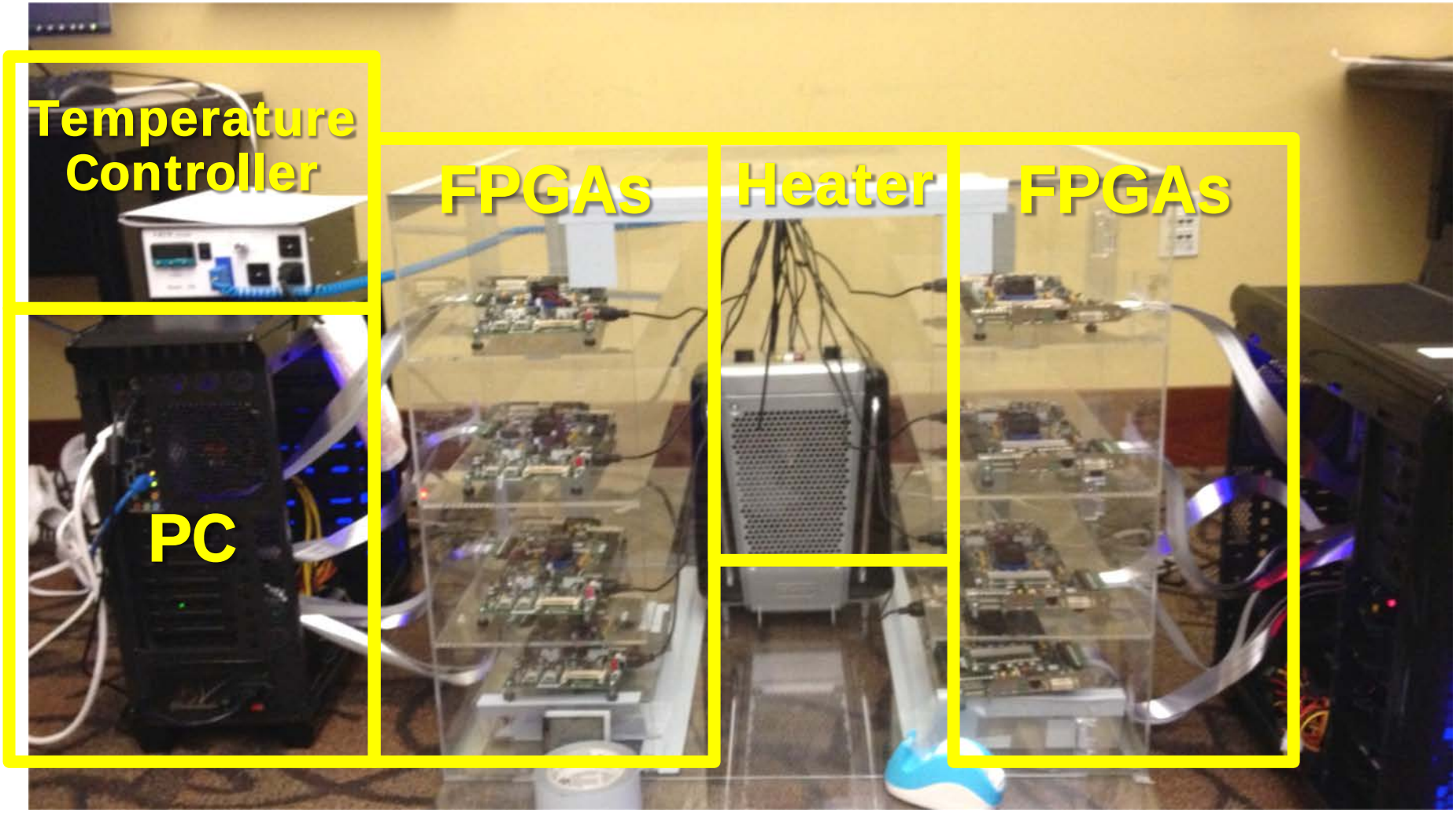
# Experimental Infrastructure (DRAM)


Liu+, "An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms", ISCA 2013.

Khan+, "The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study," SIGMETRICS 2014.



# Experimental Infrastructure (DRAM)





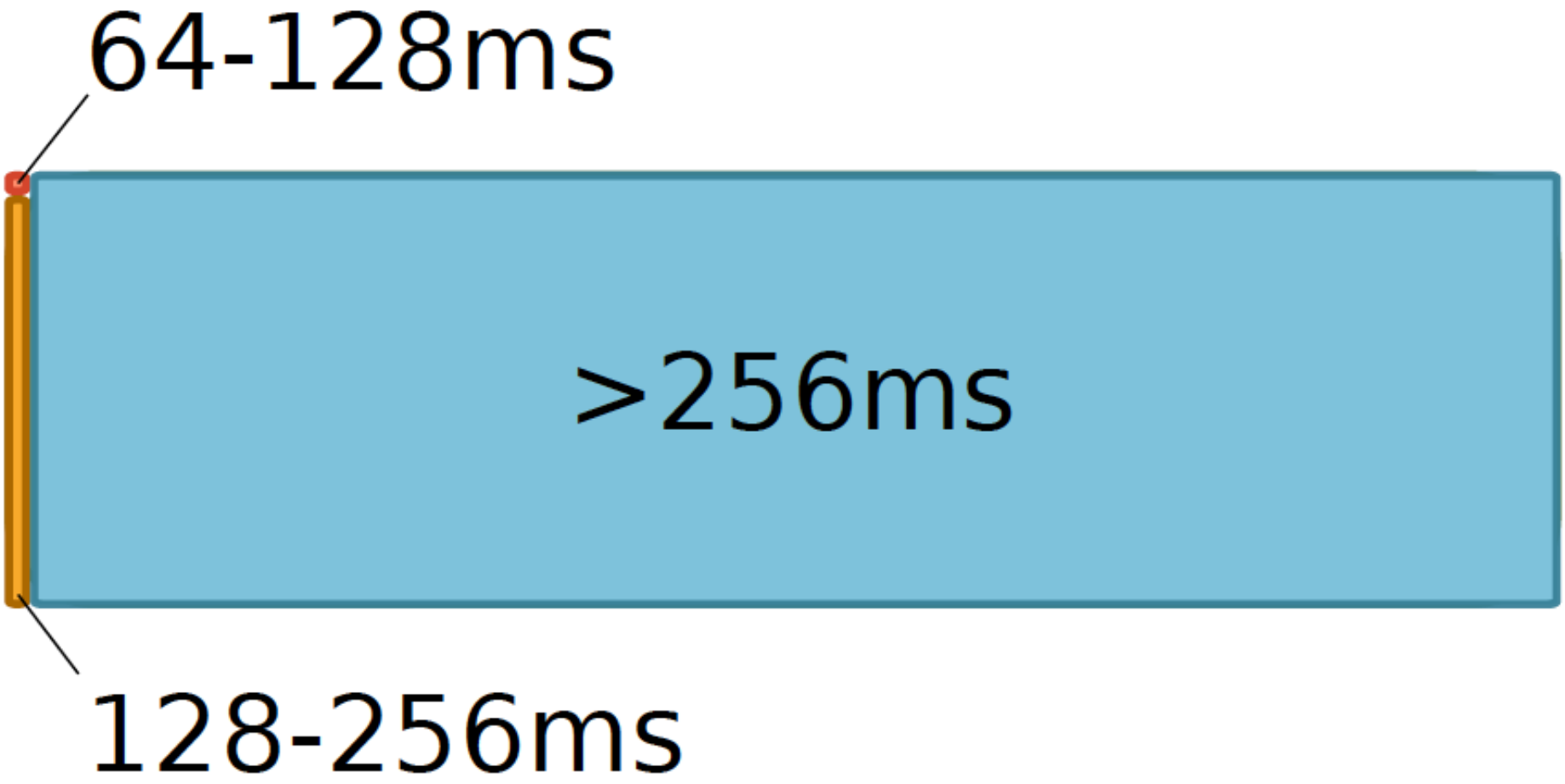

Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

# DRAM Testing Platform and Method

- Test platform: Developed a DDR3 DRAM testing platform using the Xilinx ML605 FPGA development board
  - Temperature controlled

- Tested DRAM chips: 248 commodity DRAM chips from five manufacturers (A,B,C,D,E)
- Seven families based on equal capacity per device:
  - A 1Gb, A 2Gb
  - B 2Gb
  - C 2Gb
  - D 1Gb, D 2Gb
  - E 2Gb

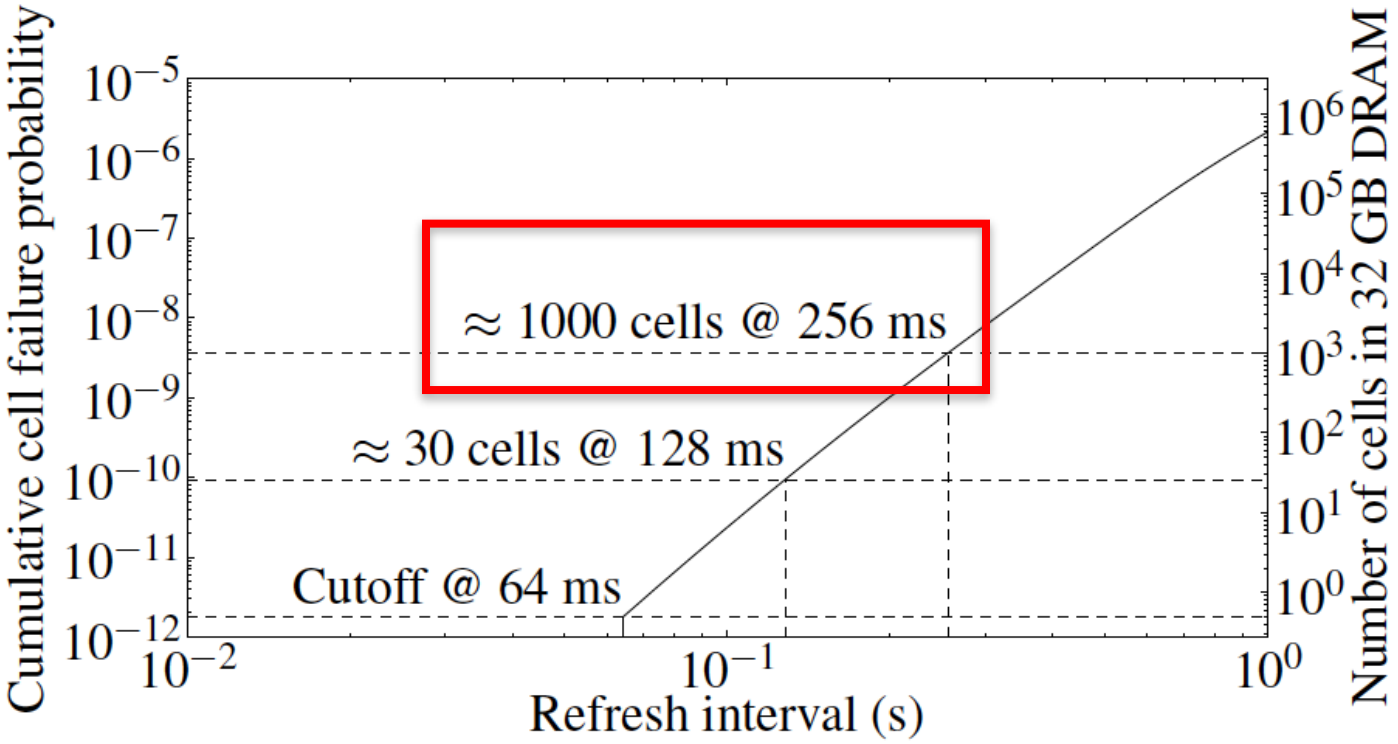
# Underneath: Retention Time Profile of DRAM




Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# Retention Time of DRAM Rows

- Observation: Overwhelming majority of DRAM rows can be refreshed much less often without losing data




Key Idea of RAIDR: Refresh weak rows more frequently, all other rows less frequently



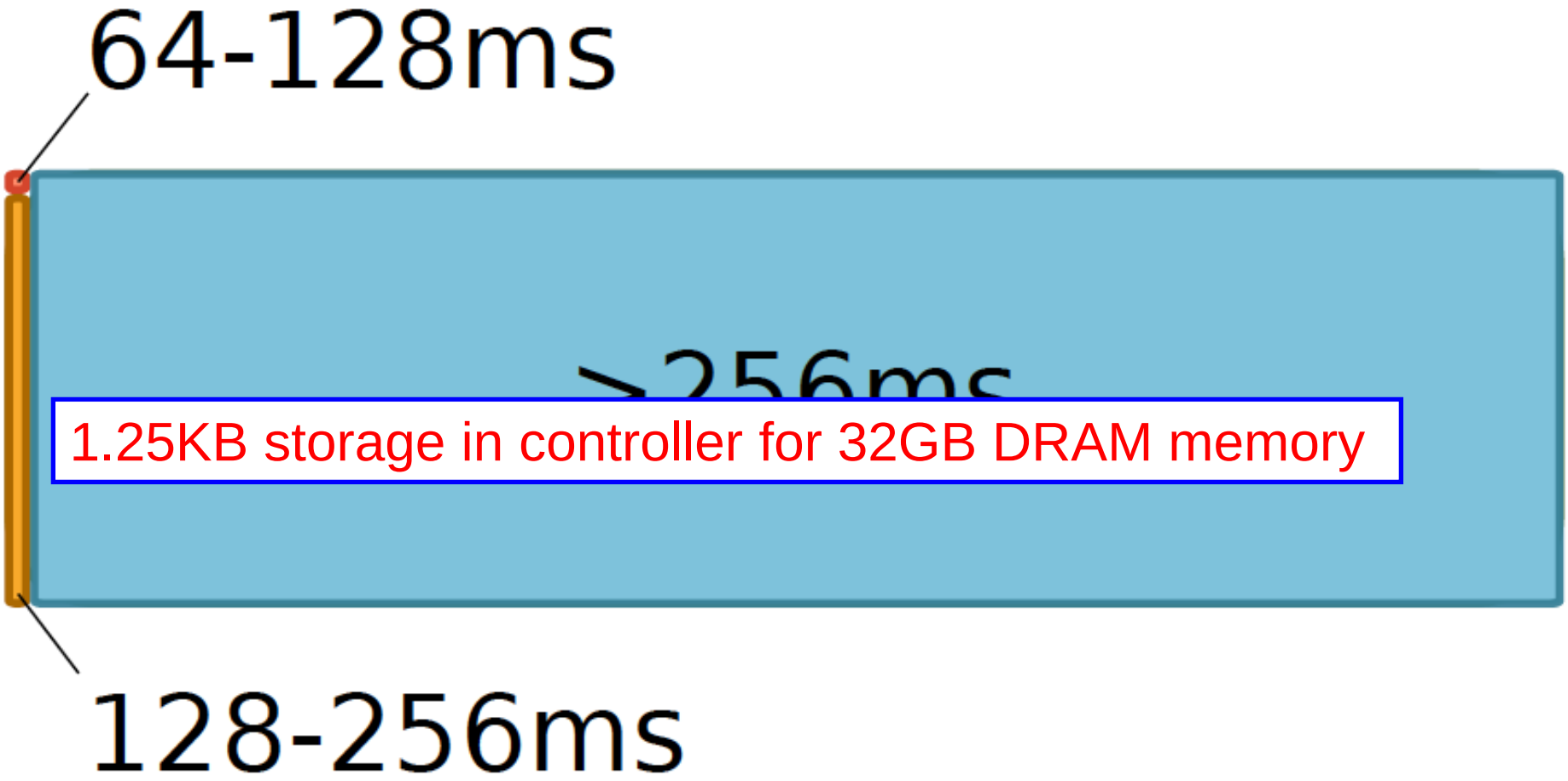
Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# RAIDR: Eliminating Unnecessary DRAM Refreshes

Liu, Jaiyen, Veras, Mutlu,
RAIDR: Retention-Aware Intelligent DRAM Refresh
ISCA 2012.

# RAIDR: Mechanism

1. Profiling: Identify the retention time of all DRAM rows



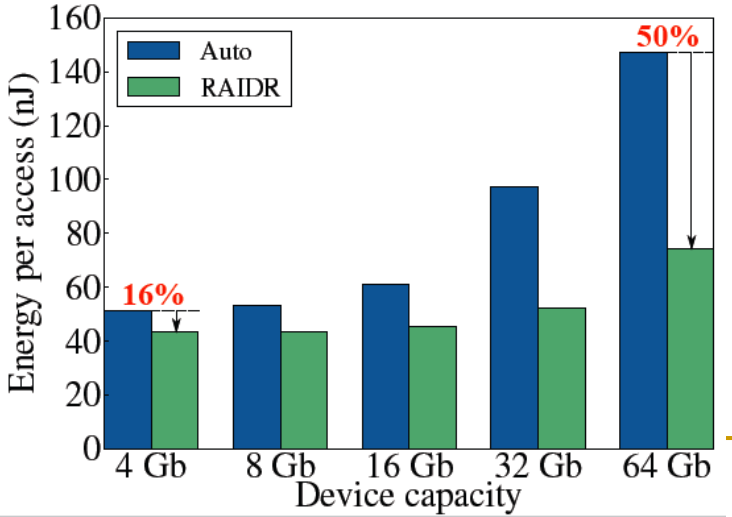

→ check the bins to determine refresh rate of a row

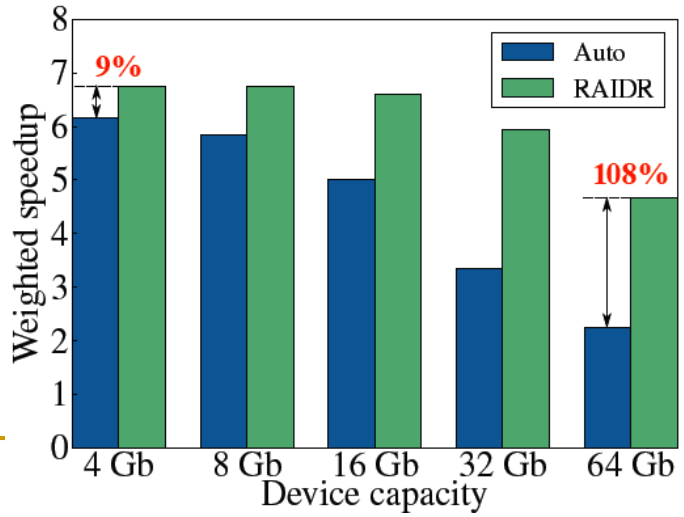


Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# RAIDR: Results and Takeaways

- System: 32GB DRAM, 8-core; Various workloads
- RAIDR hardware cost: 1.25 kB (2 Bloom filters)
- Refresh reduction: 74.6%
- Dynamic DRAM energy reduction: 16%
- Idle DRAM power reduction: 20%
- Performance improvement: 9%
- Benefits increase as DRAM scales in density



# Reading on RAIDR

- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu,
  **"RAIDR: Retention-Aware Intelligent DRAM Refresh"**
  *Proceedings of the 39th International Symposium on Computer Architecture (ISCA)*, Portland, OR, June 2012. Slides (pdf)

- One potential reading for your Homework 1 assignment

**RAIDR: Retention-Aware Intelligent DRAM Refresh**

Jamie Liu Ben Jaiyen Richard Veras Onur Mutlu
Carnegie Mellon University
{jamiel,bjaiyen,rveras,onur}@cmu.edu

# If You Are Interested … Further Readings

- Onur Mutlu,
  **"Memory Scaling: A Systems Architecture Perspective"**
  *Technical talk at MemCon 2013 (MEMCON)*, Santa Clara, CA, August 2013.
  Slides (pptx) (pdf) Video

- Kevin Chang, Donghyuk Lee, Zeshan Chishti, Alaa Alameldeen, Chris Wilkerson, Yoongu Kim, and Onur Mutlu,
  **"Improving DRAM Performance by Parallelizing Refreshes with Accesses"**
  *Proceedings of the 20th International Symposium on High-Performance Computer Architecture (HPCA)*, Orlando, FL, February 2014. Slides (pptx) (pdf)

# Takeaway 1

Breaking the abstraction layers
(between components and transformation hierarchy levels)

and knowing what is underneath

enables you to **understand** and **solve** problems

# Takeaway 2

Cooperation between
multiple components and layers
can enable
more effective
solutions and systems

# Digging Deeper: Making RAIDR Work

"Good ideas are a dime a dozen"

"Making them work is oftentimes the real contribution"

# Recall: RAIDR: Mechanism

1. Profiling: Identify the retention time of all DRAM rows

   → can be done at design time or during operation

2. Binning: Store rows into bins by retention time

   → use Bloom Filters for efficient and scalable storage

   1.25KB storage in controller for 32GB DRAM memory

3. Refreshing: Memory controller refreshes rows in different bins at different rates

   → check the bins to determine refresh rate of a row



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

# 1. Profiling

To profile a row:

1. Write data to the row
2. Prevent it from being refreshed
3. Measure time before data corruption

| | Row 1 | Row 2 | Row 3 |
|---|---|---|---|
| Initially | 11111111... | 11111111... | 11111111... |
| After 64 ms | 11111111... | 11111111... | 11111111... |
| After 128 ms | 11011111... (64–128ms) | 11111111... | 11111111... |
| After 256 ms | | 11111011... (128–256ms) | 11111111... (>256ms) |

# DRAM Retention Time Profiling

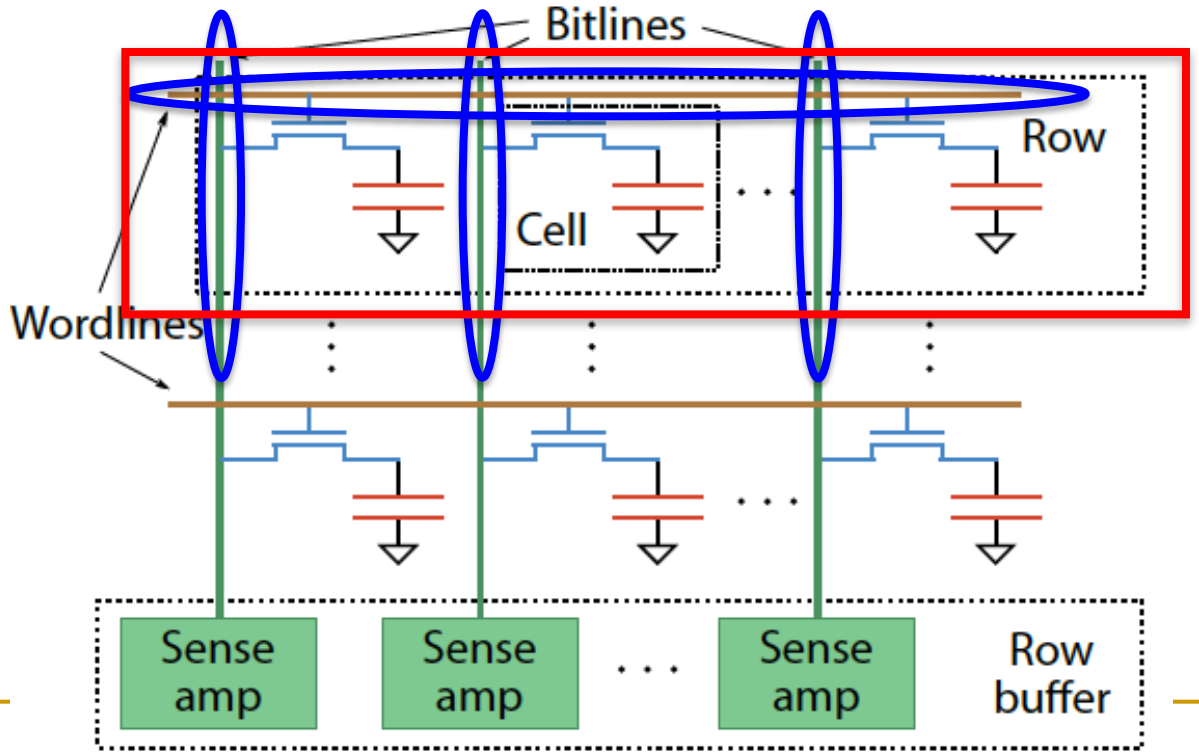
- Q: Is it really this easy?
- A: Ummm, not really...

# Two Challenges to Retention Time Profiling

- Data Pattern Dependence (DPD) of retention time
- Variable Retention Time (VRT) phenomenon

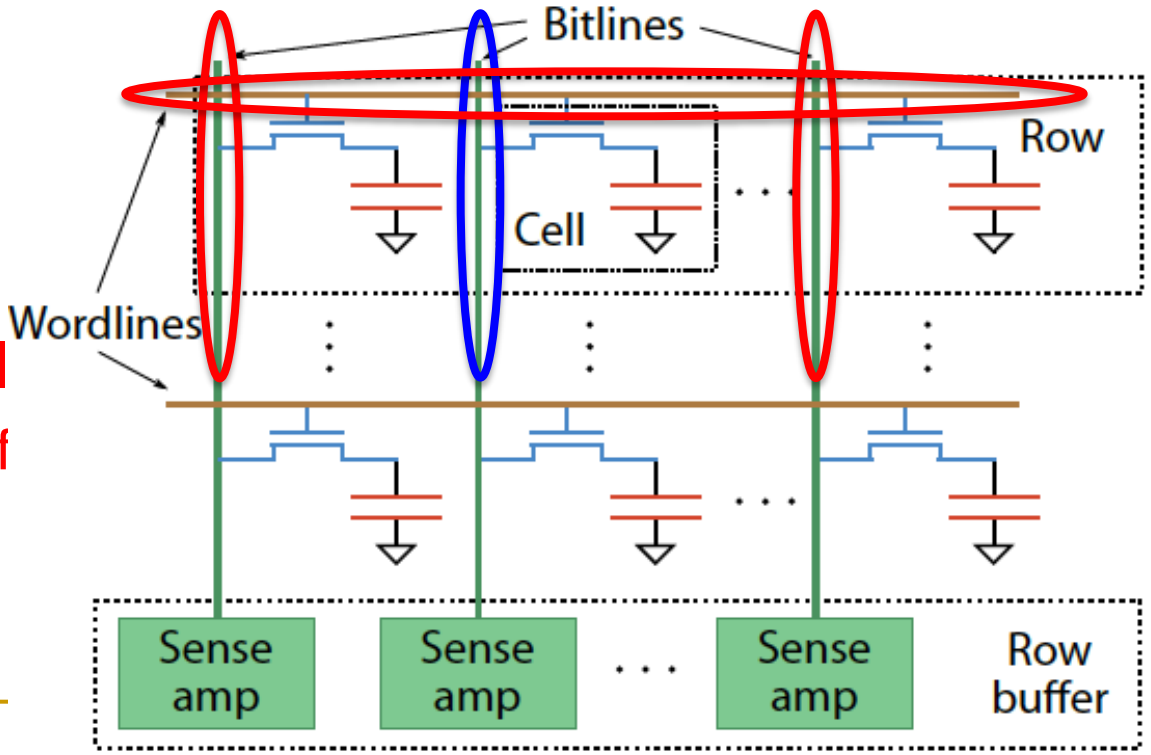
# Two Challenges to Retention Time Profiling

- Challenge 1: Data Pattern Dependence (DPD)
  - Retention time of a DRAM cell depends on its value and the values of cells nearby it
  - When a row is activated, all bitlines are perturbed simultaneously



# Data Pattern Dependence

- Electrical noise on the bitline affects reliable sensing of a DRAM cell
- The magnitude of this noise is affected by values of nearby cells via
  - Bitline-bitline coupling → electrical coupling between adjacent bitlines
  - Bitline-wordline coupling → electrical coupling between each bitline and the activated wordline




- Retention
  nearby cel
  → need to f

tored in

etention time

# DPD: Implications on Profiling Mechanisms

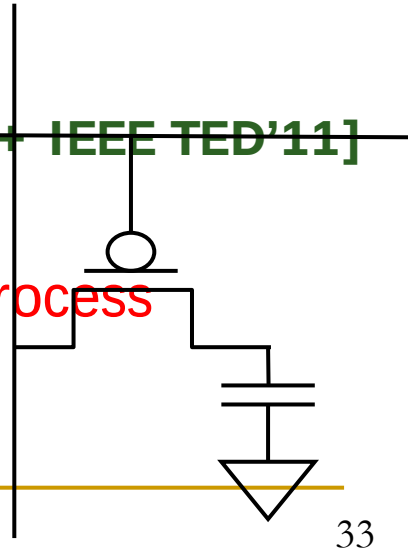
- Any retention time profiling mechanism must handle data pattern dependence of retention time
- Intuitive approach: Identify the data pattern that induces the worst-case retention time for a particular cell or device

- Problem 1: Very hard to know at the memory controller which bits actually interfere with each other due to
  - Opaque mapping of addresses to physical DRAM geometry → logically consecutive bits may not be physically consecutive
  - Remapping of faulty bitlines/wordlines to redundant ones internally within DRAM

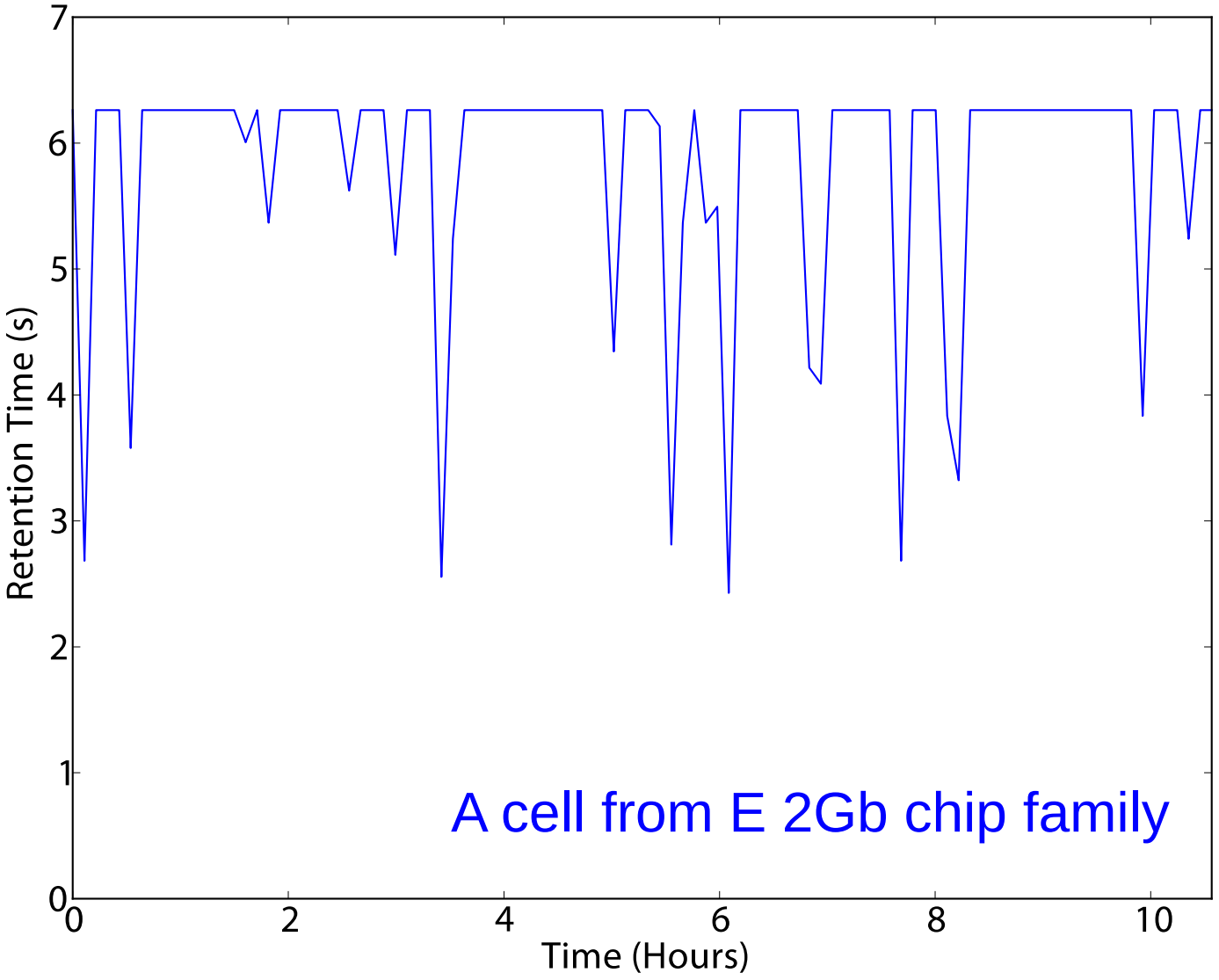
- Problem 2: Worst-case coupling noise is affected by non-obvious second order bitline coupling effects

# Two Challenges to Retention Time Profiling

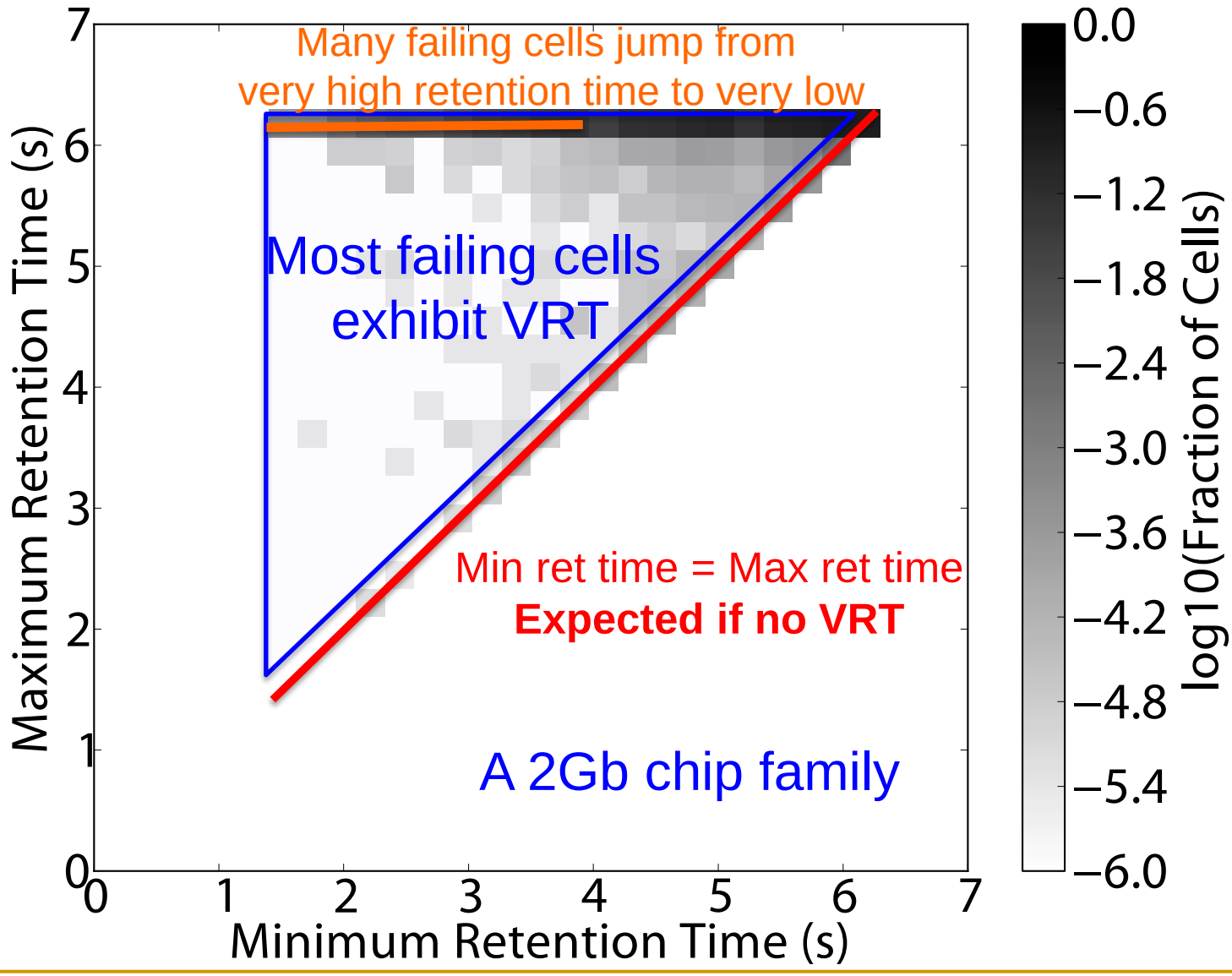
- Challenge 2: Variable Retention Time (VRT)
  - Retention time of a DRAM cell changes randomly over time
    - a cell alternates between multiple retention time states
  - Leakage current of a cell changes sporadically due to a charge trap in the gate oxide of the DRAM cell access transistor
  - When the trap becomes occupied, charge leaks more readily from the transistor's drain, leading to a short retention time
    - Called *Trap-Assisted Gate-Induced Drain Leakage*
  - This process appears to be a random process **[Kim + IEEE TED'11]**
  - Worst-case retention time depends on a random process
    → need to find the worst case despite this



# An Example VRT Cell



# Variable Retention Time



# VRT: Implications on Profiling Mechanisms

- Problem 1: There does not seem to be a way of determining if a cell exhibits VRT without actually observing a cell exhibiting VRT
  - VRT is a memoryless random process **[Kim+ JJAP 2010]**
- Problem 2: VRT complicates retention time profiling by DRAM manufacturers
  - Exposure to very high temperatures can induce VRT in cells that were not previously susceptible
    → can happen during soldering of DRAM chips
    → manufacturer's retention time profile may not be accurate
- One option for future work: Use ECC to continuously profile DRAM online while aggressively reducing refresh rate
  - Need to keep ECC overhead in check

# More on DRAM Retention Analysis

- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu, **"An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms"** *Proceedings of the 40th International Symposium on Computer Architecture (ISCA)*, Tel-Aviv, Israel, June 2013. Slides (ppt) Slides (pdf)

## An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms


Jamie Liu*
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jamiel@alumni.cmu.edu

Ben Jaiyen*
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
bjaiyen@alumni.cmu.edu

Yoongu Kim
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yoonguk@ece.cmu.edu

Chris Wilkerson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
chris.wilkerson@intel.com

Onur Mutlu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
onur@cmu.edu

# Finding DRAM Retention Failures

- How can we reliably find the retention time of all DRAM cells?

- Goals: so that we can
  - Make DRAM reliable and secure
  - Make techniques like RAIDR work
    → improve performance and energy

# Mitigation of Retention Issues [SIGMETRICS'14]

- Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa Alameldeen, Chris Wilkerson, and Onur Mutlu,
  **"The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study"**
  *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, Austin, TX, June 2014. [Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Full data sets]

## The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study


Samira Khan[†*]
samirakhan@cmu.edu

Donghyuk Lee[†]
donghyuk1@cmu.edu

Yoongu Kim[†]
yoongukim@cmu.edu

Alaa R. Alameldeen[*]
alaa.r.alameldeen@intel.com

Chris Wilkerson[*]
chris.wilkerson@intel.com

Onur Mutlu[†]
onur@cmu.edu

[†]Carnegie Mellon University [*]Intel Labs

# Handling Variable Retention Time [DSN'15]

- Moinuddin Qureshi, Dae Hyun Kim, Samira Khan, Prashant Nair, and Onur Mutlu, **"AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems"**
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Rio de Janeiro, Brazil, June 2015.
  [Slides (pptx) (pdf)]

## AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems

Moinuddin K. Qureshi[†] Dae-Hyun Kim[†] Samira Khan[‡] Prashant J. Nair[†] Onur Mutlu[‡]
[†]Georgia Institute of Technology
{moin, dhkim, pnair6}@ece.gatech.edu
[‡]Carnegie Mellon University
{samirakhan, onur}@cmu.edu

# Handling Data-Dependent Failures [DSN'16]

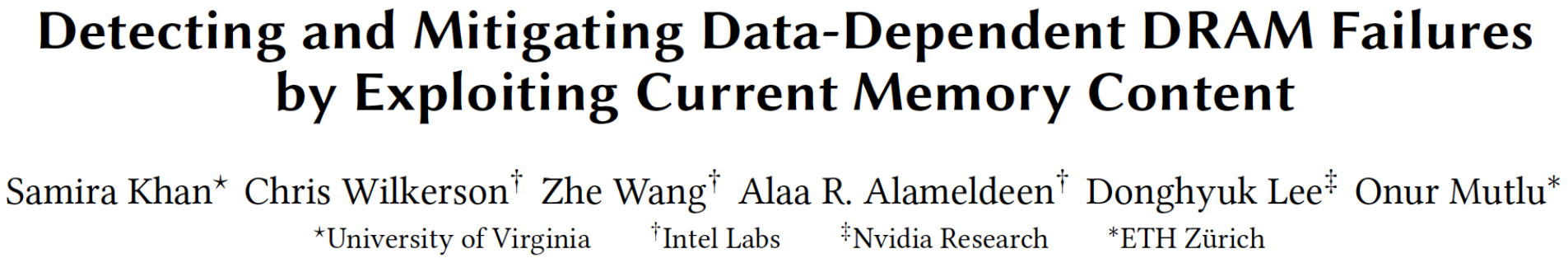
- Samira Khan, Donghyuk Lee, and Onur Mutlu,
  **"PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM"**
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Toulouse, France, June 2016.
  [Slides (pptx) (pdf)]

**PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM**

Samira Khan* Donghyuk Lee†‡ Onur Mutlu*†
*University of Virginia †Carnegie Mellon University ‡Nvidia *ETH Zürich

# Handling Data-Dependent Failures [MICRO'17]

- Samira Khan, Chris Wilkerson, Zhe Wang, Alaa R. Alameldeen, Donghyuk Lee, and Onur Mutlu,
  **"Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content"**
  *Proceedings of the 50th International Symposium on Microarchitecture (MICRO)*, Boston, MA, USA, October 2017.
  [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]



# Handling Both DPD and VRT [ISCA'17]

- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  **"The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"**
  *Proceedings of the 44th International Symposium on Computer Architecture (ISCA)*, Toronto, Canada, June 2017.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

- First experimental analysis of (mobile) LPDDR4 chips
- Analyzes the complex tradeoff space of retention time profiling
- Idea: enable fast and robust profiling at higher refresh intervals & temperatures

**The Reach Profiler (REAPER):**
**Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions**

Minesh Patel[§‡] Jeremie S. Kim[‡§] Onur Mutlu[§‡]
[§]ETH Zürich [‡]Carnegie Mellon University

# In-DRAM ECC Complicates Things [DSN'19]

- Minesh Patel, Jeremie S. Kim, Hasan Hassan, and Onur Mutlu,
  **"Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices"**
  *Proceedings of the 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Portland, OR, USA, June 2019.
  [Slides (pptx) (pdf)]
  [Talk Video (26 minutes)]
  [Full Talk Lecture (29 minutes)]
  [Source Code for EINSim, the Error Inference Simulator]
  ***Best paper award.***

**Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices**

Minesh Patel[†] Jeremie S. Kim[‡†] Hasan Hassan[†] Onur Mutlu[†‡]

[†]*ETH Zürich* [‡]*Carnegie Mellon University*

# More on In-DRAM ECC [MICRO'20]

- Minesh Patel, Jeremie S. Kim, Taha Shahroodi, Hasan Hassan, and Onur Mutlu,
  **"Bit-Exact ECC Recovery (BEER): Determining DRAM On-Die ECC Functions by Exploiting DRAM Data Retention Characteristics"**
  *Proceedings of the 53rd International Symposium on Microarchitecture (MICRO)*, Virtual, October 2020.

**Bit-Exact ECC Recovery (BEER): Determining DRAM On-Die ECC Functions by Exploiting DRAM Data Retention Characteristics**

Minesh Patel[†] Jeremie S. Kim[‡†] Taha Shahroodi[†] Hasan Hassan[†] Onur Mutlu[†‡]

[†]*ETH Zürich* [‡]*Carnegie Mellon University*

# 2. Binning

- How to efficiently and scalably store rows into retention time bins?
- Use Hardware Bloom Filters [Bloom, CACM 1970]

Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

# Bloom Filter

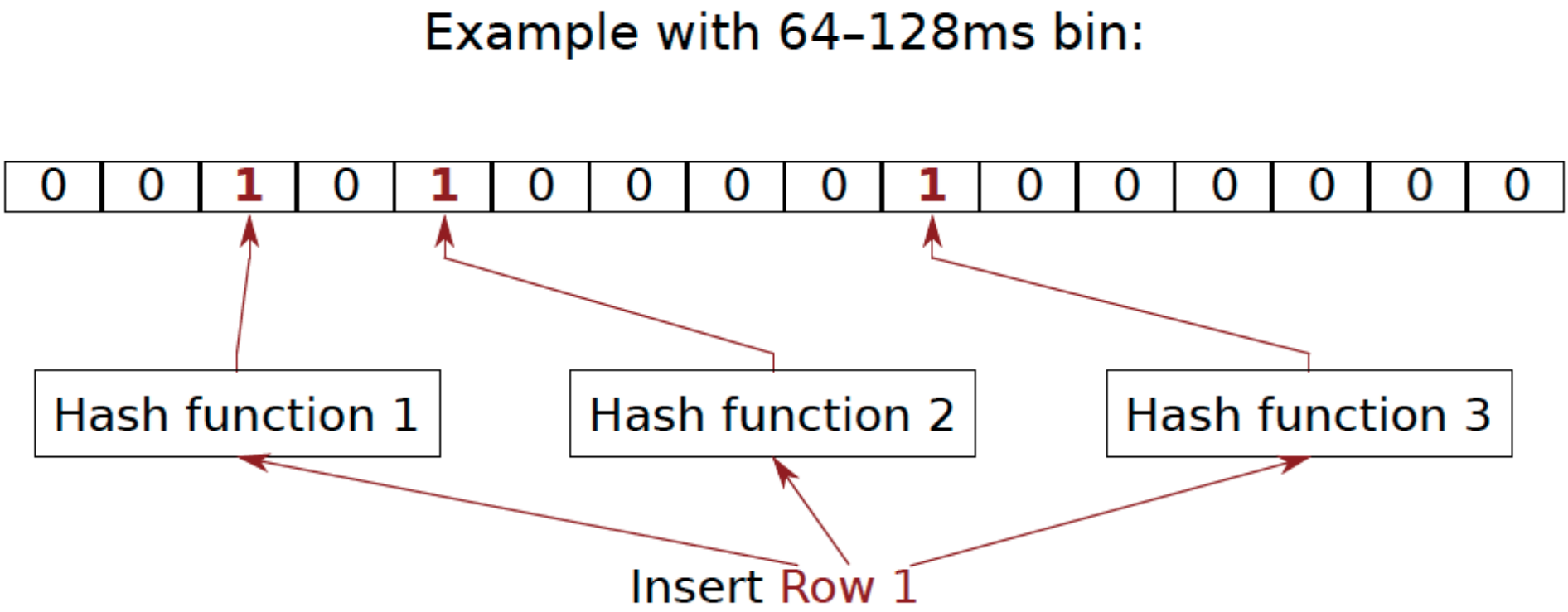
- [Bloom, CACM 1970]
- Probabilistic data structure that compactly represents set membership (presence or absence of element in a set)

- Non-approximate set membership: Use 1 bit per element to indicate absence/presence of each element from an element space of N elements
- Approximate set membership: use a much smaller number of bits and indicate each element's presence/absence with a subset of those bits
  - Some elements map to the bits other elements also map to

- Operations: 1) insert, 2) test, 3) remove all elements

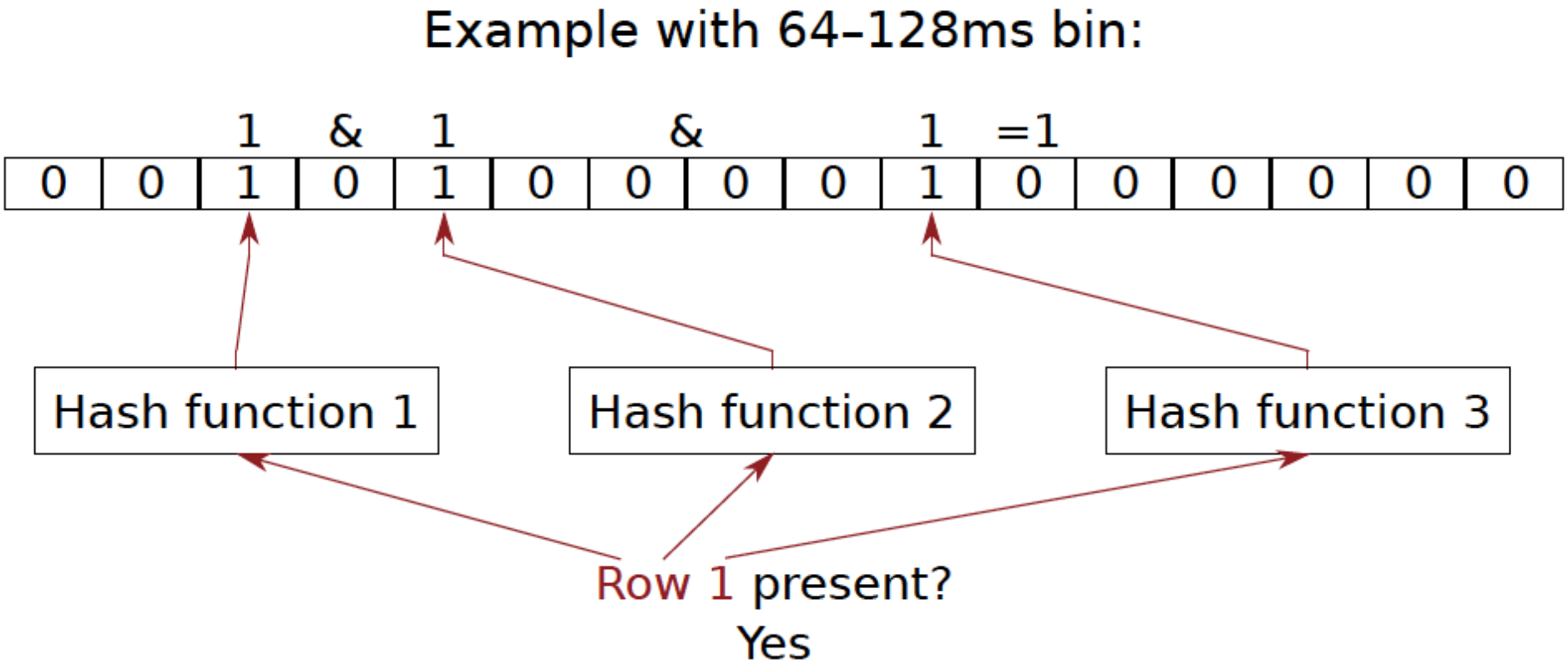
Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

# Bloom Filter Operation Example

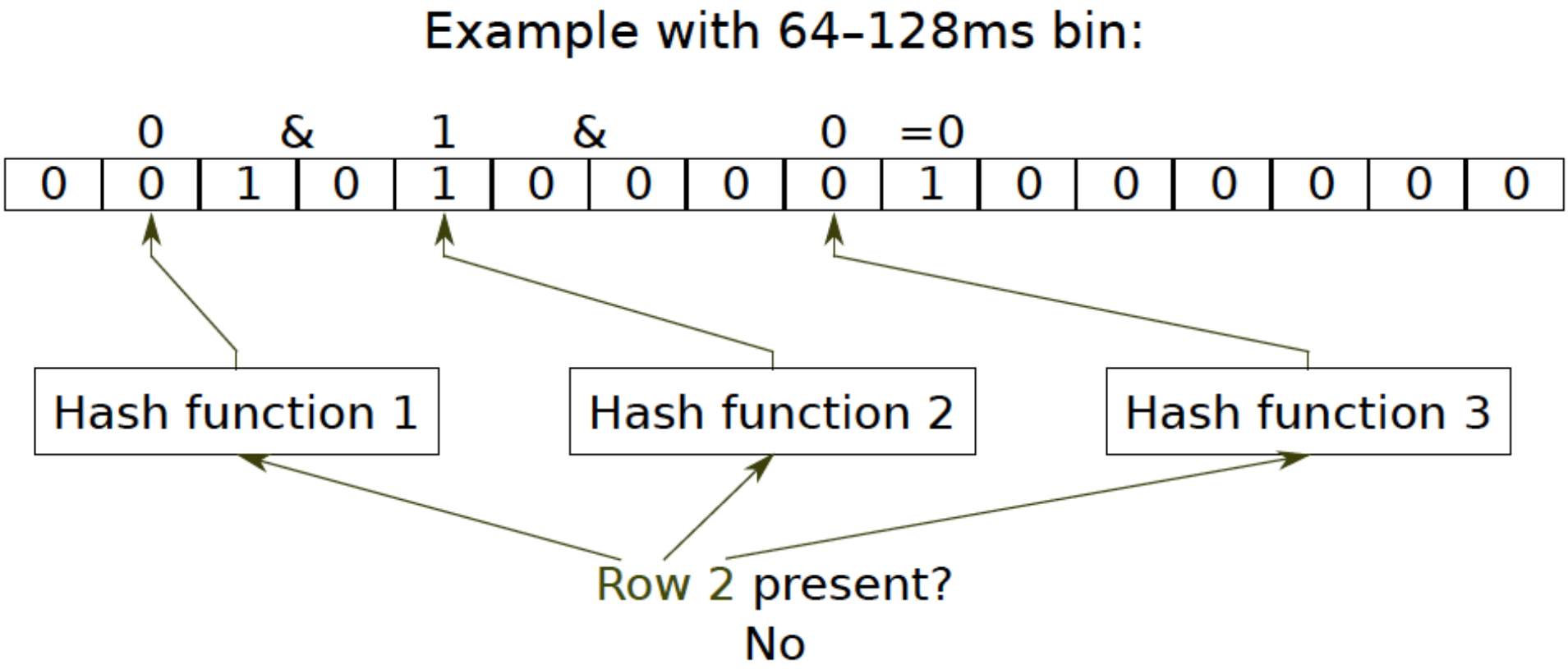



Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

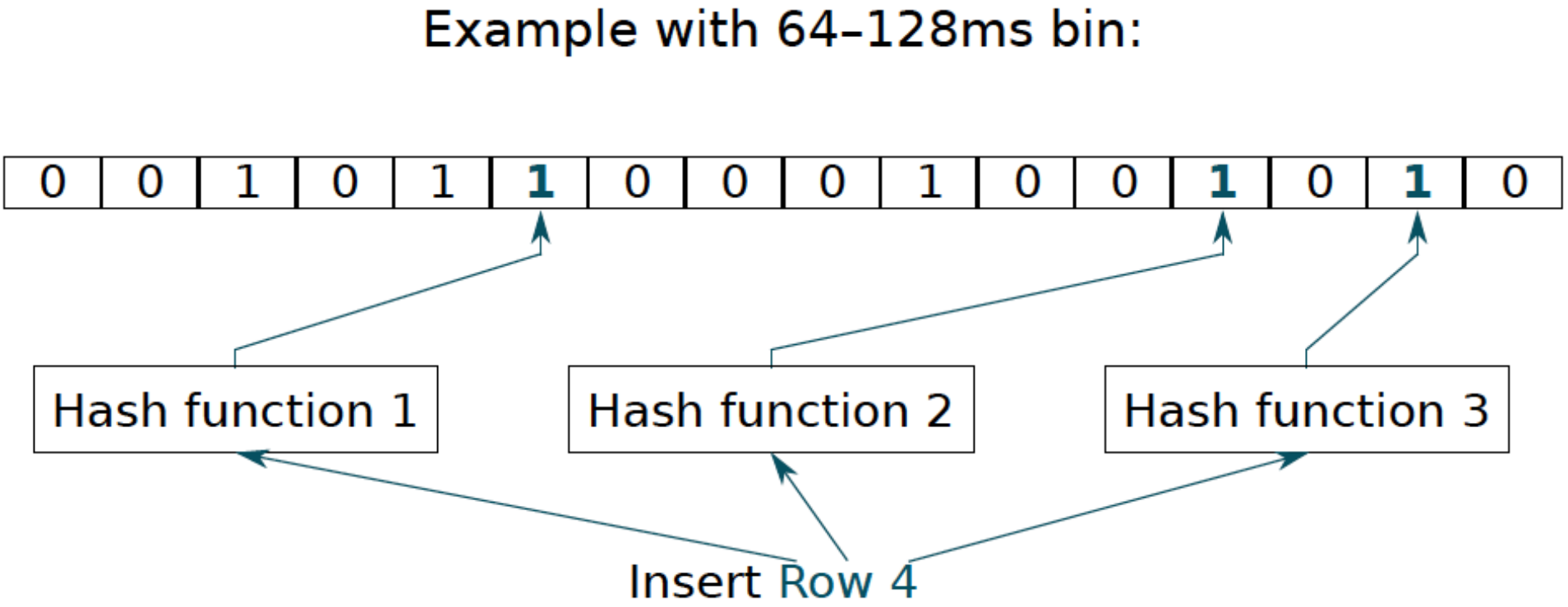
# Bloom Filter Operation Example



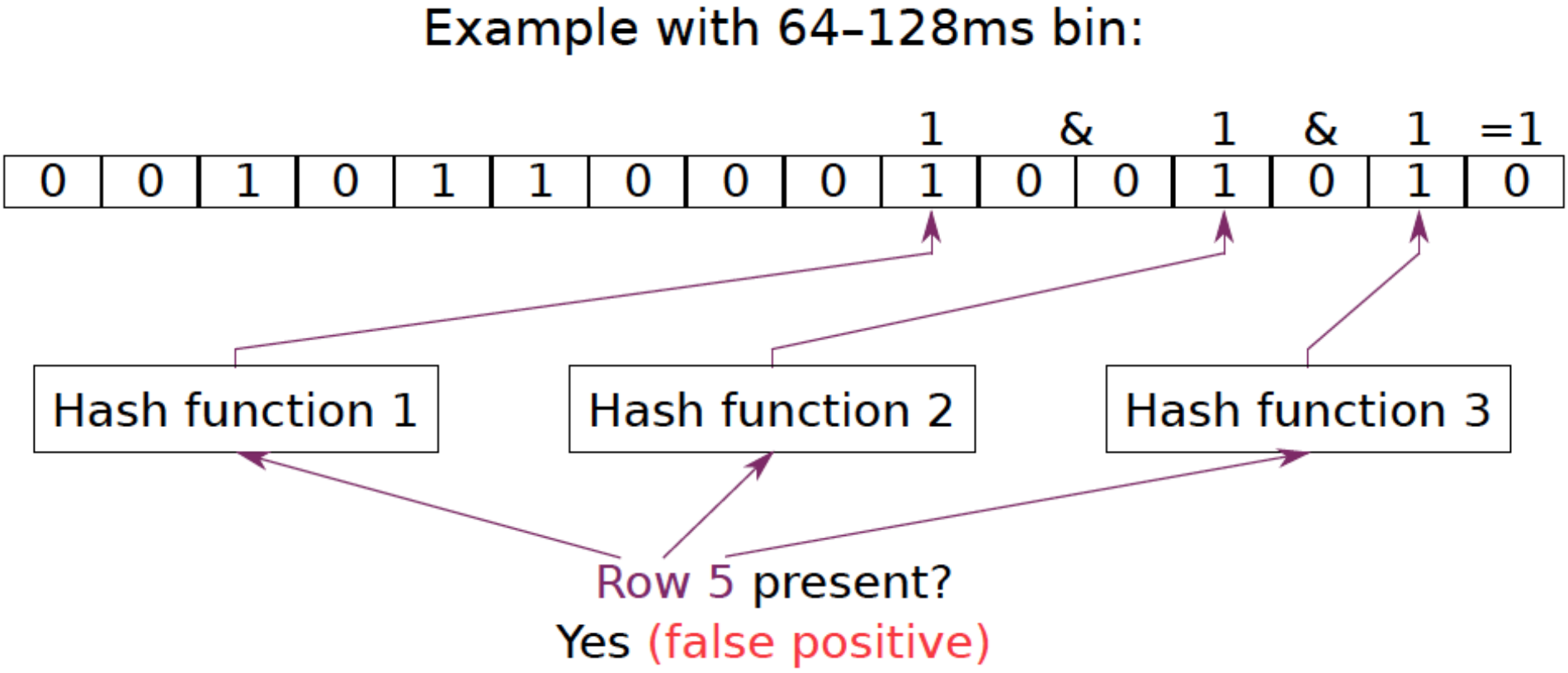
# Bloom Filter Operation Example



# Bloom Filter Operation Example



# Bloom Filter Operation Example



# Bloom Filters

## Space/Time Trade-offs in Hash Coding with Allowable Errors

In such applications, it is envisaged that overall performance could be improved by using a smaller core resident hash area in conjunction with the new methods and, when necessary, by using some secondary and perhaps time-consuming test to "catch" the small fraction of errors associated with the new methods. An example is discussed which illustrates possible areas of application for the new methods.

BURTON H. BLOOM
*Computer Usage Company, Newton Upper Falls, Mass.*

In this paper trade-offs among certain computational factors in hash coding are analyzed. The paradigm problem considered is that of testing a series of messages one-by-one for membership in a given set of messages. Two new hash-coding methods are examined and compared with a particular conventional hash-coding method. The computational factors considered are the size of the hash area (space), the time required to identify a message as a nonmember of the given set (reject time), and an allowable error frequency.

Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

# Bloom Filters: Pros and Cons

- Advantages

  + Enables storage-efficient representation of set membership

  + Insertion and testing for set membership (presence) are fast

  + No false negatives: If Bloom Filter says an element is not present in the set, the element must not have been inserted

  + Enables tradeoffs between time & storage efficiency & false positive rate (via sizing and hashing)

- Disadvantages

  -- False positives: An element may be deemed to be present in the set by the Bloom Filter but it may never have been inserted

  Not the right data structure when you cannot tolerate false positives

Bloom, "Space/Time Trade-offs in Hash Coding with Allowable Errors", CACM 1970.

# Benefits of Bloom Filters as Refresh Rate Bins

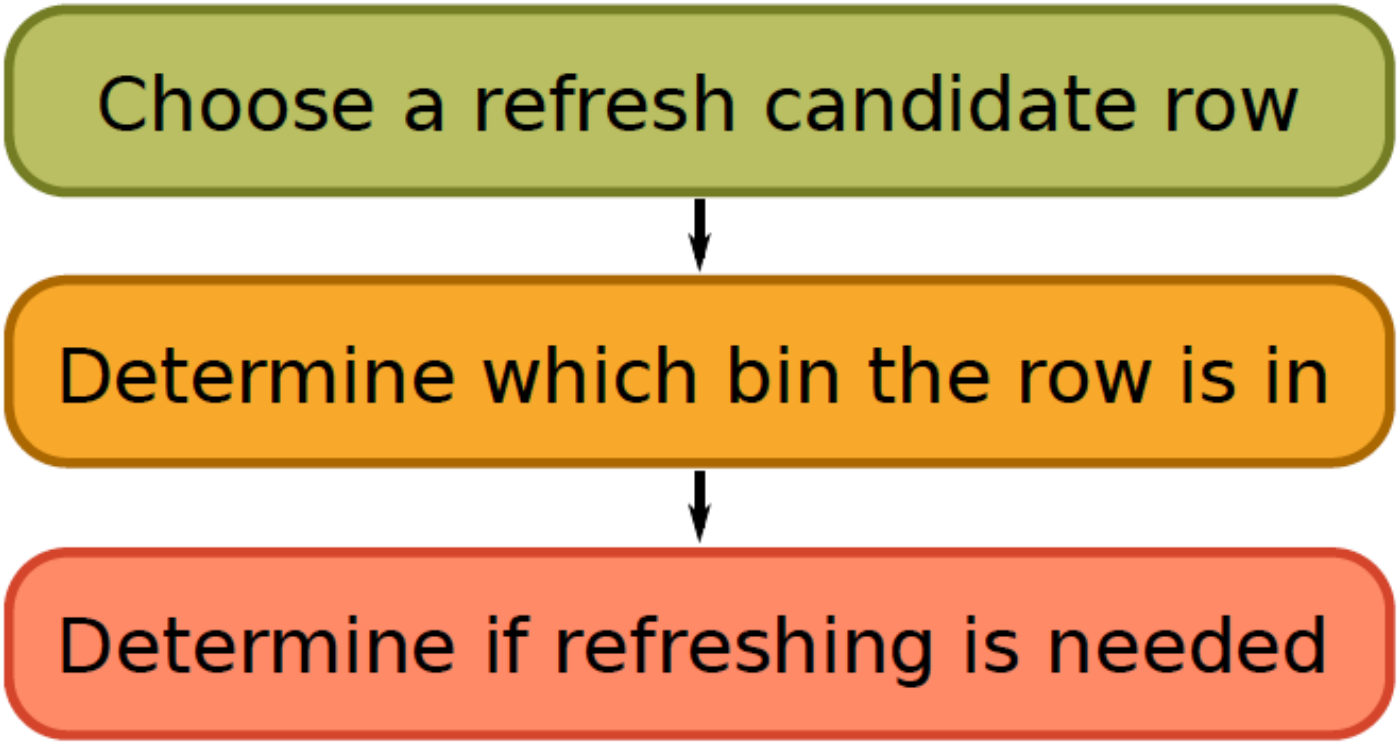
- False positives: a row may be declared present in the Bloom filter even if it was never inserted
  - Not a problem: Refresh some rows more frequently than needed
- No false negatives: rows are never refreshed less frequently than needed (no correctness problems)
- Scalable: a Bloom filter never overflows (unlike a fixed-size table)
- Efficient: No need to store info on a per-row basis; simple hardware → 1.25 KB for 2 filters for 32 GB DRAM system

# Use of Bloom Filters in Hardware

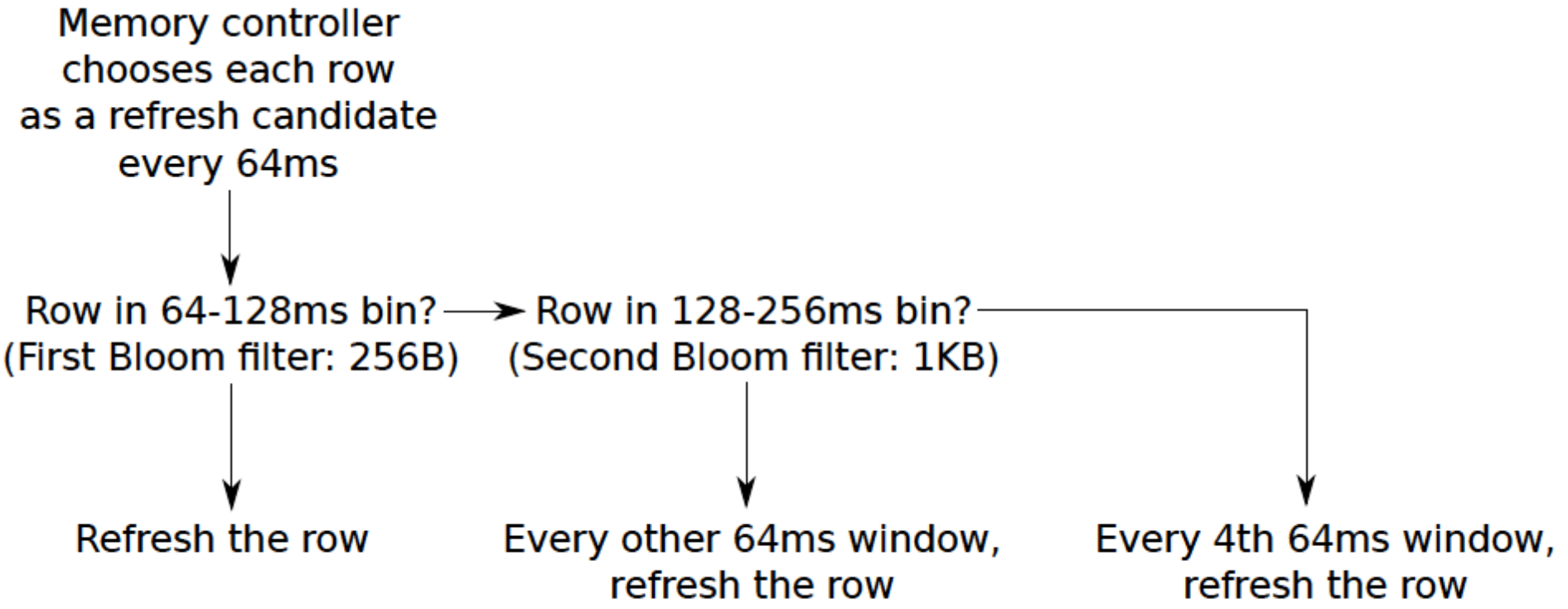
- Useful when you can tolerate false positives in set membership tests

- See the following recent examples for clear descriptions of how Bloom Filters are used
  - Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.
  - Seshadri et al., "The Evicted-Address Filter: A Unified Mechanism to Address Both Cache Pollution and Thrashing," PACT 2012.

# 3. Refreshing (RAIDR Refresh Controller)

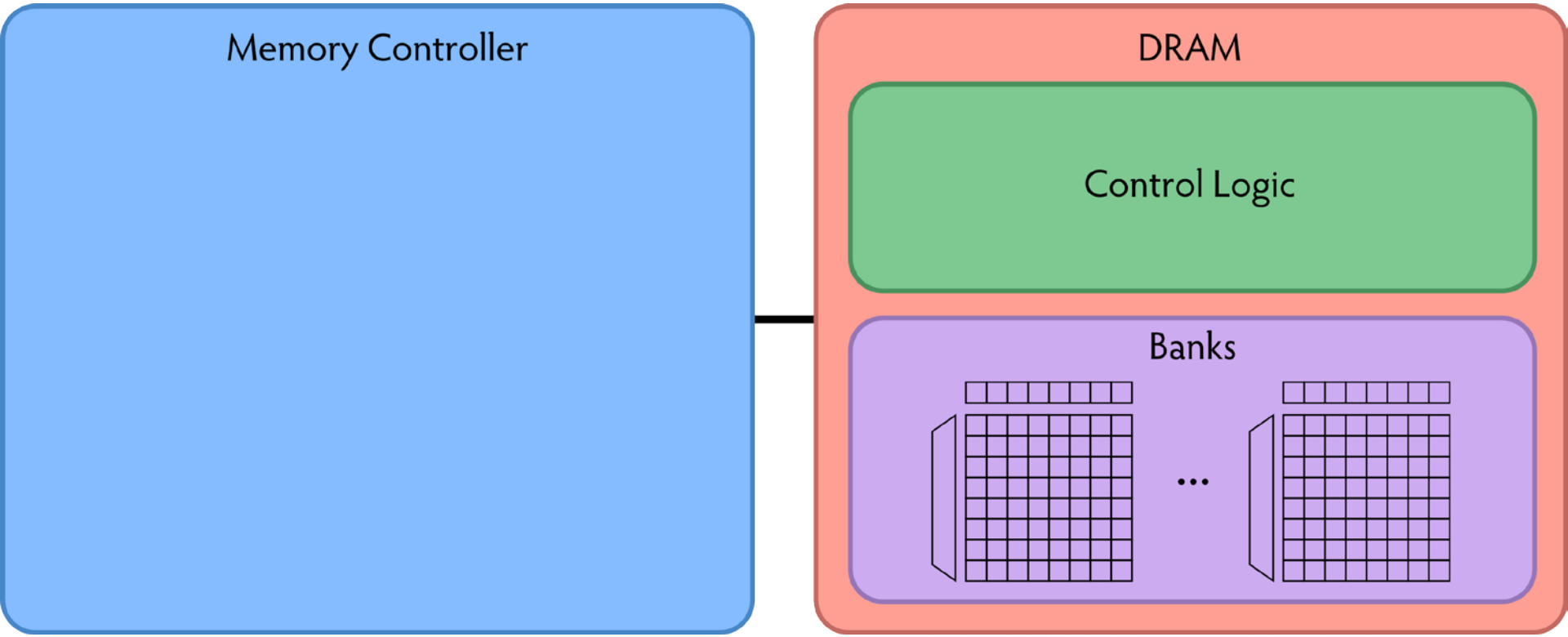


# 3. Refreshing (RAIDR Refresh Controller)




Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

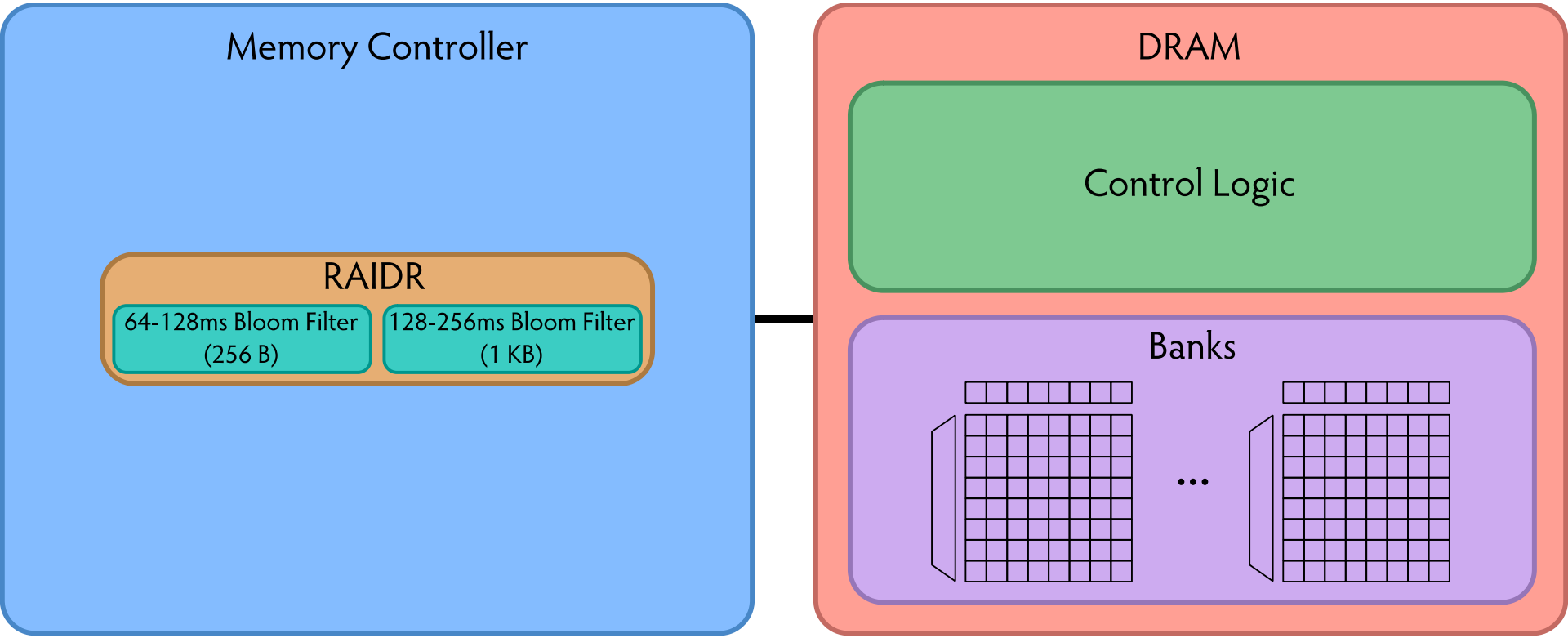
# RAIDR: Baseline Design




Refresh control is in DRAM in today's auto-refresh systems

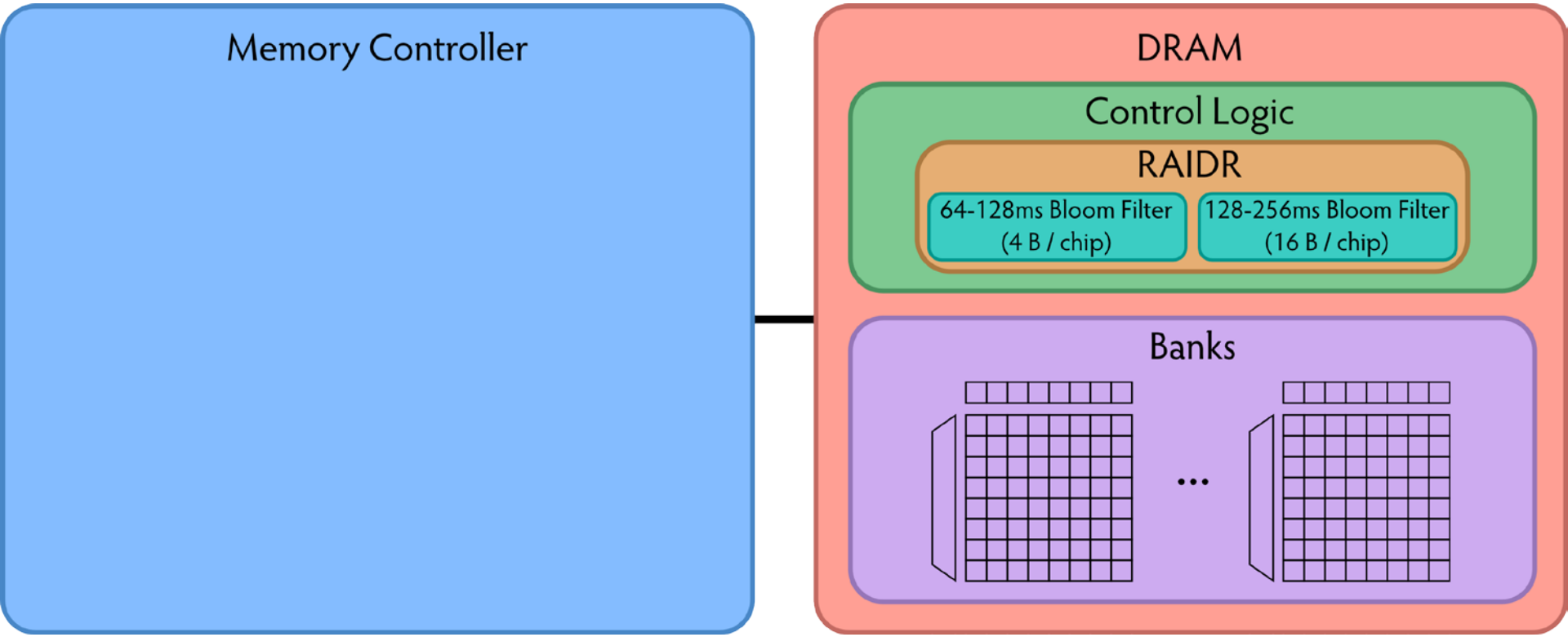
RAIDR can be implemented in either the controller or DRAM

# RAIDR in Memory Controller: Option 1




Overhead of RAIDR in DRAM controller:
1.25 KB Bloom Filters, 3 counters, additional commands issued for per-row refresh (all accounted for in evaluations)

# RAIDR in DRAM Chip: Option 2



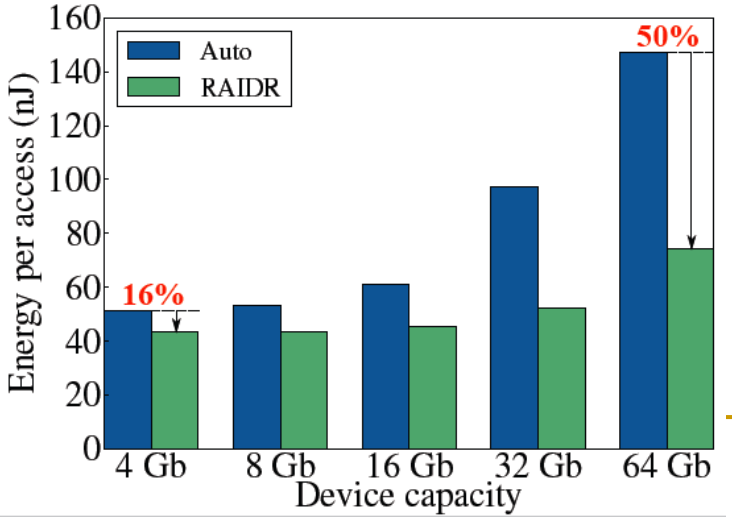

Overhead of RAIDR in DRAM chip:

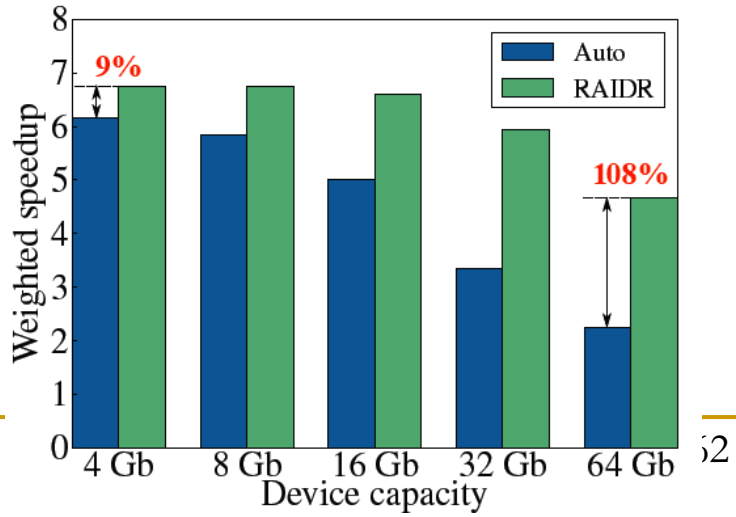
Per-chip overhead: 20B Bloom Filters, 1 counter (4 Gbit chip)

Total overhead: 1.25KB Bloom Filters, 64 counters (32 GB DRAM)

# RAIDR: Results and Takeaways

- System: 32GB DRAM, 8-core; SPEC, TPC-C, TPC-H workloads
- RAIDR hardware cost: 1.25 kB (2 Bloom filters)
- Refresh reduction: 74.6%
- Dynamic DRAM energy reduction: 16%
- Idle DRAM power reduction: 20%
- Performance improvement: 9%
- Benefits increase as DRAM scales in density



# DRAM Refresh: More Questions

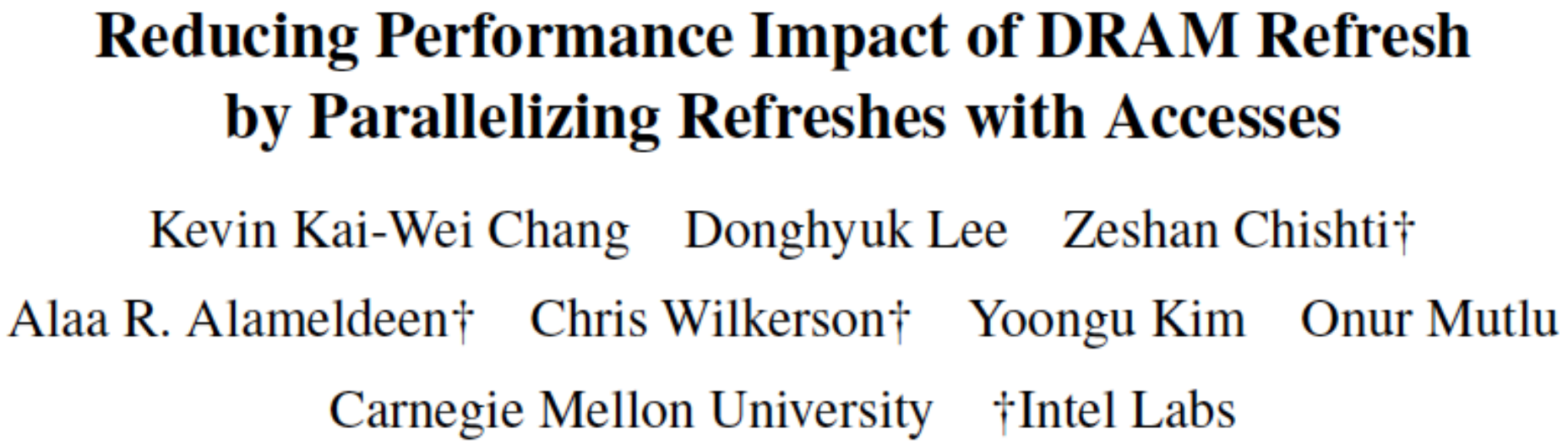
- What else can you do to reduce the impact of refresh?
- What else can you do if you know the retention times of rows?
- How can you accurately measure the retention time of DRAM rows?
- Recommended reading:
  - Liu et al., "An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms," ISCA 2013.

# DRAM Refresh: Summary and Conclusions

- DRAM refresh is a critical challenge
  - in scaling DRAM technology efficiently to higher capacities
- Several promising solution directions
  - Eliminate unnecessary refreshes [Liu+ ISCA'12]
  - Reduce refresh rate w/ online profiling and detect/correct any errors [Khan+ SIGMETRICS'14, Qureshi+ DSN'15, Patel+ ISCA'17]
  - Parallelize refreshes with accesses [Chang+ HPCA'14]
- Examined properties of retention time behavior [Liu+ ISCA'13]
  - Enable realistic VRT-Aware refresh techniques [Qureshi+ DSN'15]
- Many avenues for overcoming DRAM refresh challenges
  - Handling DPD/VRT phenomena
  - Enabling online retention time profiling and error mitigation
  - Exploiting application behavior

# More Information on Refresh-Access Parallelization

- Kevin Chang, Donghyuk Lee, Zeshan Chishti, Alaa Alameldeen, Chris Wilkerson, Yoongu Kim, and Onur Mutlu,
  **"Improving DRAM Performance by Parallelizing Refreshes with Accesses"**
  *Proceedings of the 20th International Symposium on High-Performance Computer Architecture (**HPCA**)*, Orlando, FL, February 2014.
  [Summary] [Slides (pptx) (pdf)]



**Reducing Performance Impact of DRAM Refresh by Parallelizing Refreshes with Accesses**

Kevin Kai-Wei Chang  Donghyuk Lee  Zeshan Chishti†

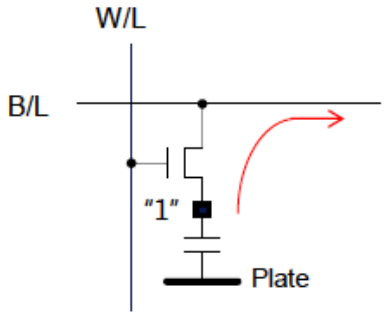
Alaa R. Alameldeen†  Chris Wilkerson†  Yoongu Kim  Onur Mutlu

Carnegie Mellon University  †Intel Labs

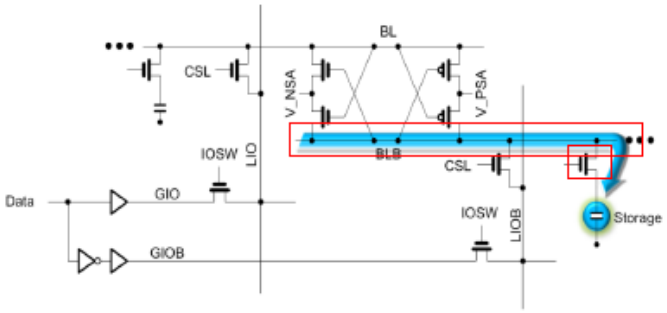
# Industry Is Writing Papers About It, Too

## DRAM Process Scaling Challenges

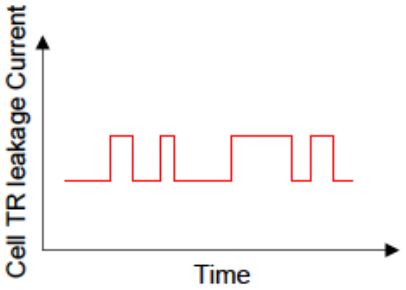
- **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance
  - Leakage current of cell access transistors increasing
- **tWR**
  - Contact resistance between the cell capacitor and access transistor increasing
  - On-current of the cell access transistor decreasing
  - Bit-line resistance increasing
- **VRT**
  - Occurring more frequently with cell capacitance decreasing




Refresh




tWR




VRT

# Call for Intelligent Memory Controllers

## DRAM Process Scaling Challenges

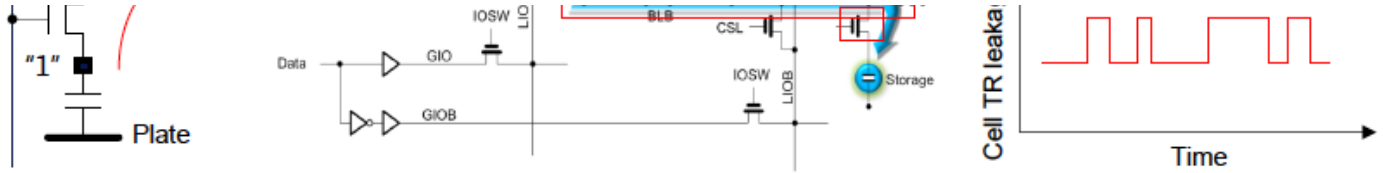
- **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance

THE MEMORY FORUM 2014

# Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

Uksong Kang, Hak-soo Yu, Churoo Park, *Hongzhong Zheng,
**John Halbert, **Kuljit Bains, SeongJin Jang, and Joo Sun Choi

*Samsung Electronics, Hwasung, Korea / *Samsung Electronics, San Jose / **Intel*




**Refresh** **tWR** **VRT**

# We Will Dig Deeper More In This Course

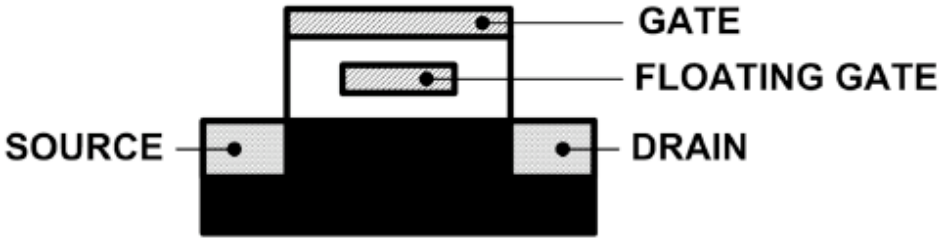
"Good ideas are a dime a dozen"

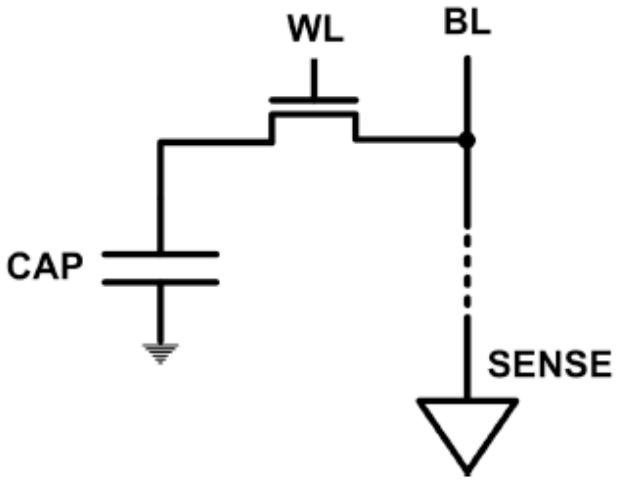
"Making them work is oftentimes the real contribution"

# Foreshadowing: Limits of Charge Memory

- Difficult charge placement and control
  - Flash: floating gate charge
  - DRAM: capacitor charge, transistor leakage

- Data retention and reliable sensing become difficult as charge storage unit size reduces



TECHSPOT

TRENDING ▾ REVIEWS ▾ FEATURES ▾ DOWNLOADS ▾ PRODUCT FINDER ▾ FORUMS ▾ TEC

# *An unfortunate tale about Samsung's SSD 840*

# read performance degradation

An avalanche of reports emerged last September, when owners of the usually speedy Samsung SSD 840 and SSD 840 EVO detected the drives were no longer performing as they used to.

The issue has to do with older blocks of data: reading **old files** nsistently slower than norma **as slow as 30MB/s** whereas **newly-written files** ones used in benchmarks, perform as fast as new – aro **500 MB/s** for the well regarded SSD 840 EVO. The reason no one had noticed (we reviewed the drive back in September 2013) is that data has to be several weeks old to show the problem. Samsung promptly admitted the issue and proposed a fix.

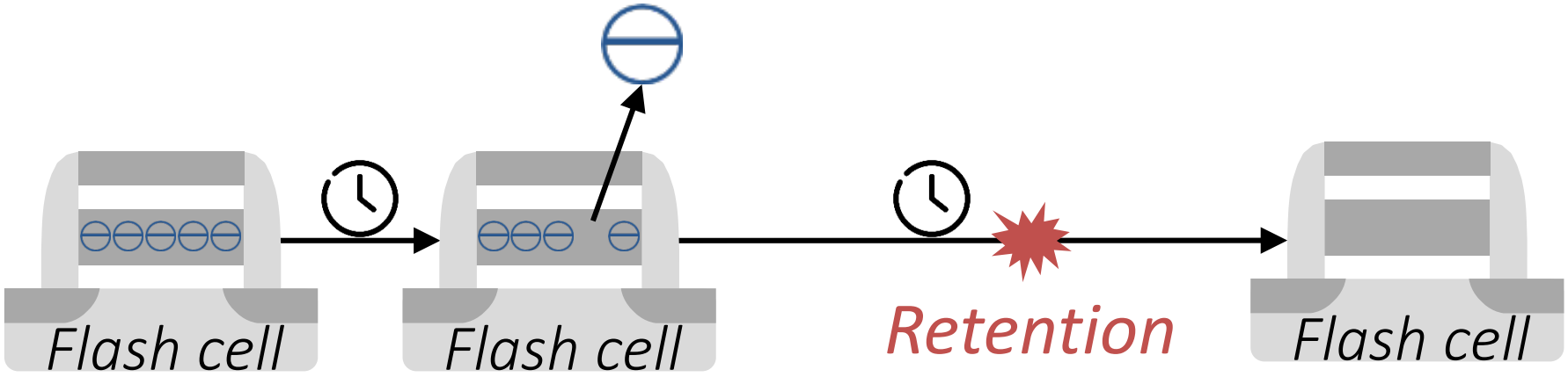
Reference: (May 5, 2015) Per Hansson, "*When SSD Performance Goes Awry*"
http://www.techspot.com/article/997-samsung-ssd-read-performance-degradation/

# Why is old data slower?


*Retention loss!*

# Retention loss

*Charge leakage over time*




*One dominant source of flash memory errors [DATE '12, ICCD '12]*

***Side effect: Longer read latency***

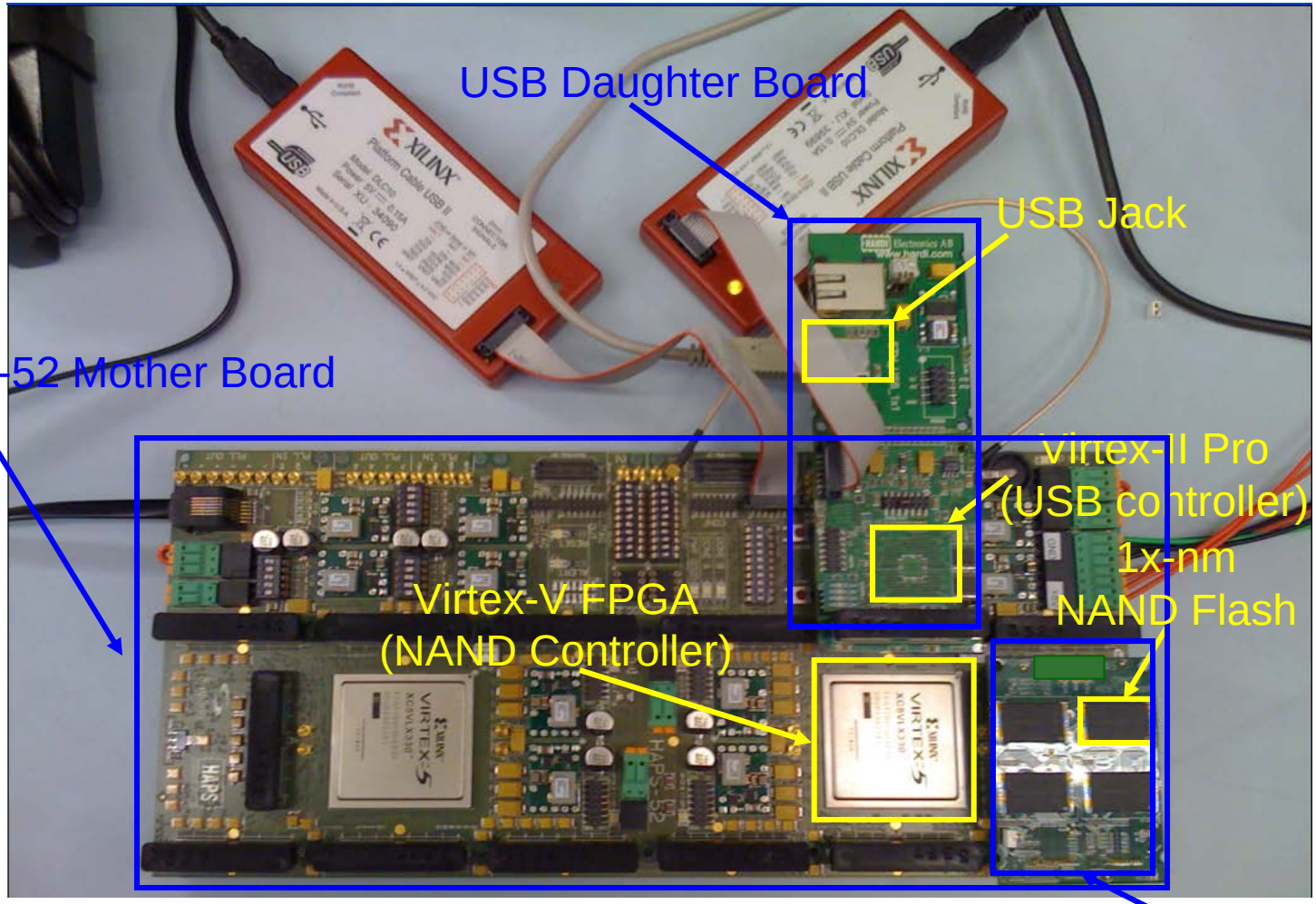
# NAND Flash Error Types

- Four types of errors [Cai+, DATE 2012]

- Caused by common flash operations
  - Read errors
  - Erase errors
  - Program (interference) errors

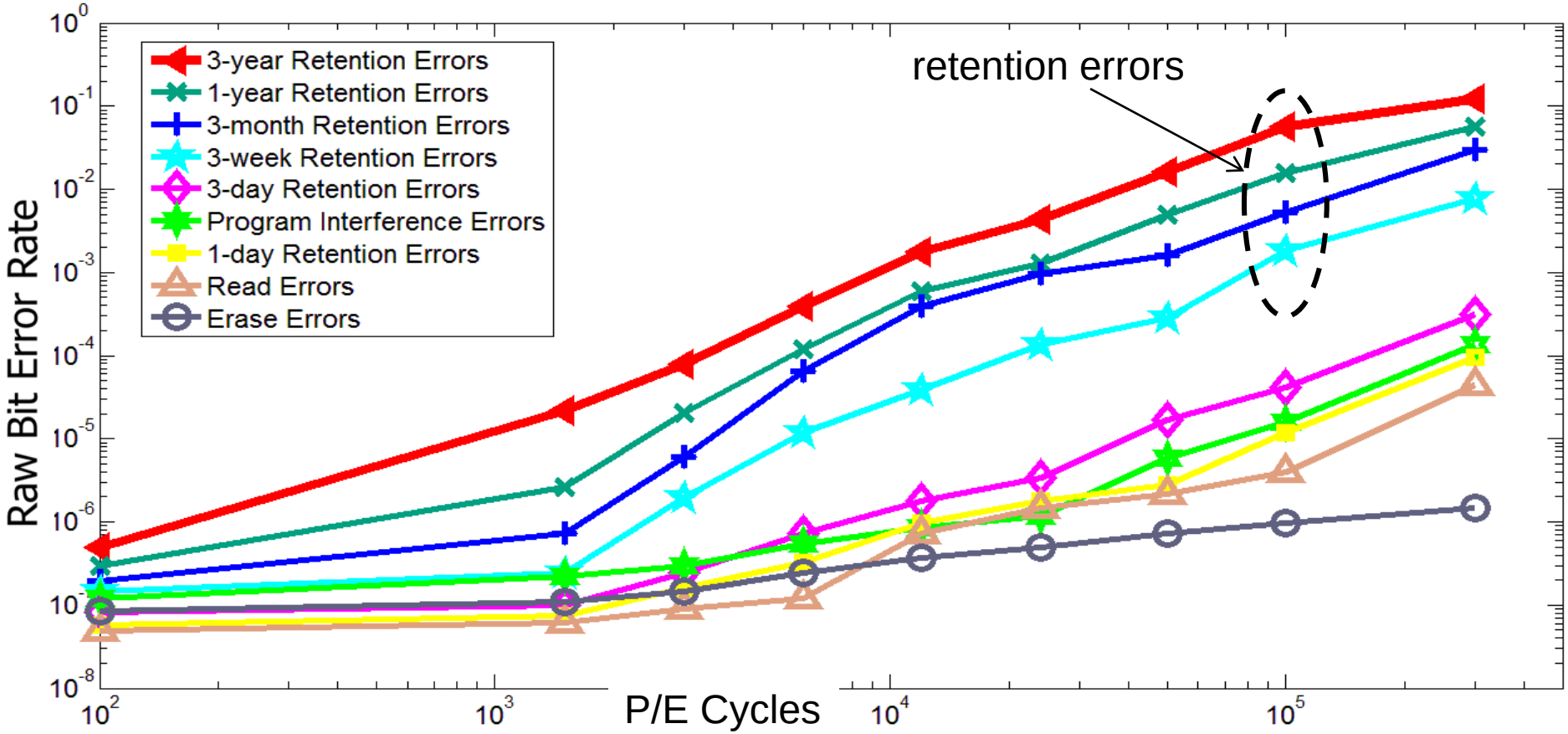
- Caused by flash cell losing charge over time
  - Retention errors
    - Whether an error happens depends on required retention time
    - Especially problematic in MLC flash because threshold voltage window to determine stored value is smaller

# Flash Experimental Testing Platform




[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

# Observations: Flash Error Analysis

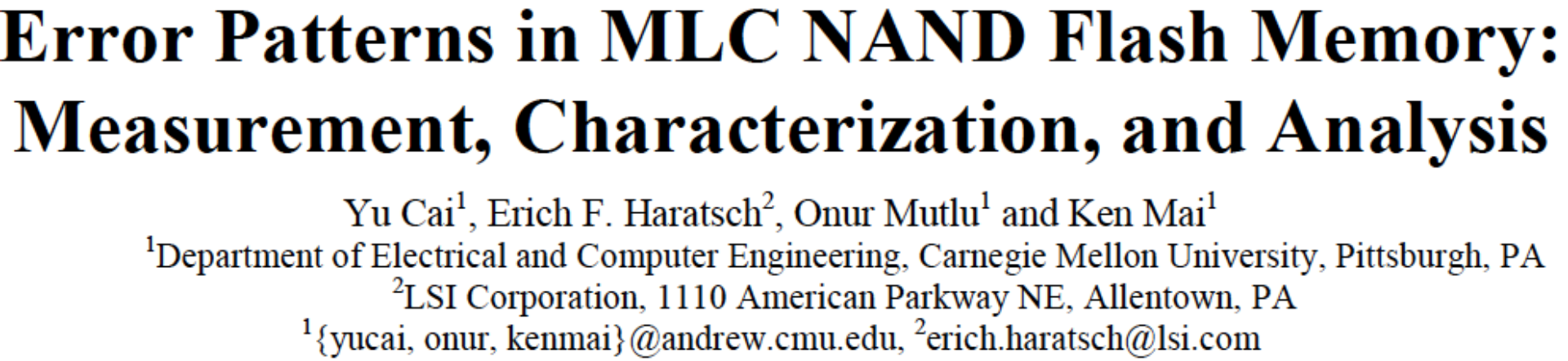



- Raw bit error rate increases exponentially with P/E cycles
- **Retention errors are dominant** (>99% for 1-year ret. time)
- Retention errors increase with retention time requirement


Cai et al., Error Patterns in MLC NAND Flash Memory, DATE 2012.

# More on Flash Error Analysis

- Yu Cai, Erich F. Haratsch, Onur Mutlu, and Ken Mai, **"Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis"** *Proceedings of the Design, Automation, and Test in Europe Conference (DATE)*, Dresden, Germany, March 2012. Slides (ppt)



**Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis**

Yu Cai[1], Erich F. Haratsch[2], Onur Mutlu[1] and Ken Mai[1]
[1]Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA
[2]LSI Corporation, 1110 American Parkway NE, Allentown, PA
[1]{yucai, onur, kenmai}@andrew.cmu.edu, [2]erich.haratsch@lsi.com

# Solution to Retention Errors

- Refresh periodically
- Change the period based on P/E cycle wearout
  - Refresh more often at higher P/E cycles
- Use a combination of in-place and remapping-based refresh
- Cai et al. "Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime", ICCD 2012.

# Flash Correct-and-Refresh [ICCD'12]

- Yu Cai, Gulay Yalcin, Onur Mutlu, Erich F. Haratsch, Adrian Cristal, Osman Unsal, and Ken Mai,
  **"Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime"**
  *Proceedings of the 30th IEEE International Conference on Computer Design (**ICCD**)*, Montreal, Quebec, Canada, September 2012. Slides (ppt)(pdf)

Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime

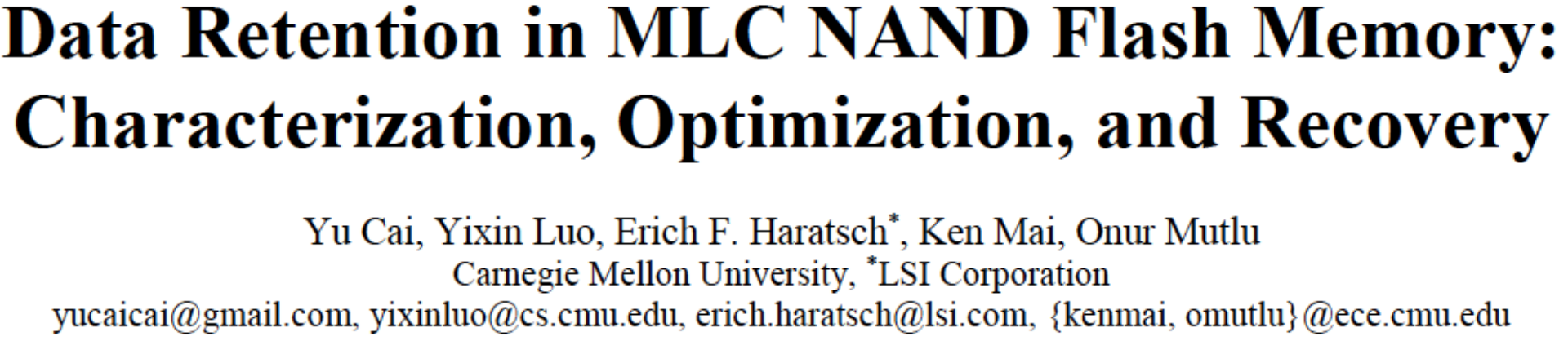
Yu Cai[1], Gulay Yalcin[2], Onur Mutlu[1], Erich F. Haratsch[3], Adrian Cristal[2], Osman S. Unsal[2] and Ken Mai[1]
[1]DSSC, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA
[2]Barcelona Supercomputing Center, C/Jordi Girona 29, Barcelona, Spain
[3]LSI Corporation, 1110 American Parkway NE, Allentown, PA

# More on Flash Error Analysis [Intel Tech J'13]

- Yu Cai, Gulay Yalcin, Onur Mutlu, Erich F. Haratsch, Adrian Cristal, Osman Unsal, and Ken Mai,
  **"Error Analysis and Retention-Aware Error Management for NAND Flash Memory"**
  *Intel Technology Journal (ITJ) Special Issue on Memory Resiliency*, Vol. 17, No. 1, May 2013.

# Flash Memory Data Retention Analysis

- Yu Cai, Yixin Luo, Erich F. Haratsch, Ken Mai, and Onur Mutlu,
  **"Data Retention in MLC NAND Flash Memory: Characterization, Optimization and Recovery"**
  *Proceedings of the 21st International Symposium on High-Performance Computer Architecture (HPCA)*, Bay Area, CA, February 2015.
  [Slides (pptx) (pdf)] [Poster (pdf)]
  ***Best paper session.***

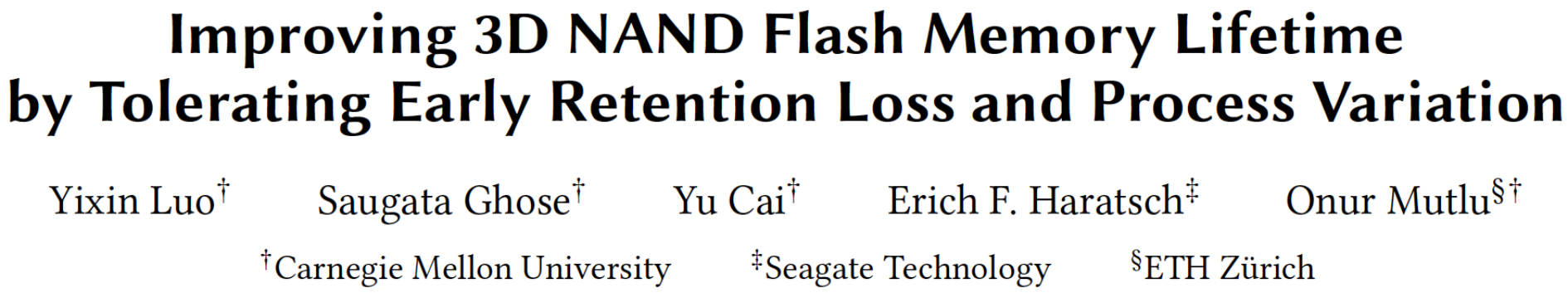


**Data Retention in MLC NAND Flash Memory: Characterization, Optimization, and Recovery**

Yu Cai, Yixin Luo, Erich F. Haratsch*, Ken Mai, Onur Mutlu
Carnegie Mellon University, *LSI Corporation
yucaicai@gmail.com, yixinluo@cs.cmu.edu, erich.haratsch@lsi.com, {kenmai, omutlu}@ece.cmu.edu

# 3D Flash Data Retention [SIGMETRICS'18]

- Yixin Luo, Saugata Ghose, Yu Cai, Erich F. Haratsch, and Onur Mutlu, **"Improving 3D NAND Flash Memory Lifetime by Tolerating Early Retention Loss and Process Variation"**
  *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Irvine, CA, USA, June 2018.
  [Abstract]
  [POMACS Journal Version (same content, different format)]
  [Slides (pptx) (pdf)]



**Improving 3D NAND Flash Memory Lifetime by Tolerating Early Retention Loss and Process Variation**

Yixin Luo[†] Saugata Ghose[†] Yu Cai[†] Erich F. Haratsch[‡] Onur Mutlu[§†]

[†]Carnegie Mellon University [‡]Seagate Technology [§]ETH Zürich

# Many Errors and Their Mitigation [PIEEE'17]


*Proceedings of the IEEE, Sept. 2017*

## Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

*This paper reviews the most recent advances in solid-state drive (SSD) error characterization, mitigation, and data recovery techniques to improve both SSD's reliability and lifetime.*

By Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu

https://arxiv.org/pdf/1706.08642

# More Up-to-date Version

- Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu, **"Errors in Flash-Memory-Based Solid-State Drives: Analysis, Mitigation, and Recovery"**
  *Invited Book Chapter in Inside Solid State Drives*, 2018.
  [Preliminary arxiv.org version]

**Errors in Flash-Memory-Based Solid-State Drives: Analysis, Mitigation, and Recovery**

YU CAI, SAUGATA GHOSE
Carnegie Mellon University

ERICH F. HARATSCH
Seagate Technology

YIXIN LUO
Carnegie Mellon University

ONUR MUTLU
ETH Zürich and Carnegie Mellon University

# Lectures on Flash Memory and SSDs

- L14b: Flash Memory and Solid-State Drives
  (PDF) (PPT)
  Video
  https://www.youtube.com/watch?v=XbKOmOPjLtc

- L15: Flash Memory and Solid-State Drives II
  (PDF) (PPT)
  Video
  https://www.youtube.com/watch?v=-OPtz7Ne9og

# Computer Architecture

## Lecture 2b: Data Retention and Memory Refresh

Prof. Onur Mutlu
ETH Zürich

Fall 2020

18 September 2020

# Profiling for DRAM Data Retention Failures

# Finding DRAM Retention Failures

- How can we reliably find the retention time of all DRAM cells?

- Goals: so that we can
  - Make DRAM reliable and secure
  - Make techniques like RAIDR work
    → improve performance and energy

# Mitigation of Retention Issues [SIGMETRICS'14]

- Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa Alameldeen, Chris Wilkerson, and Onur Mutlu,
  **"The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study"**
  *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS)*, Austin, TX, June 2014. [Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Full data sets]

## The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study


Samira Khan[†*]
samirakhan@cmu.edu

Donghyuk Lee[†]
donghyuk1@cmu.edu

Yoongu Kim[†]
yoongukim@cmu.edu

Alaa R. Alameldeen[*]
alaa.r.alameldeen@intel.com

Chris Wilkerson[*]
chris.wilkerson@intel.com

Onur Mutlu[†]
onur@cmu.edu

[†]Carnegie Mellon University [*]Intel Labs

# Towards an Online Profiling System

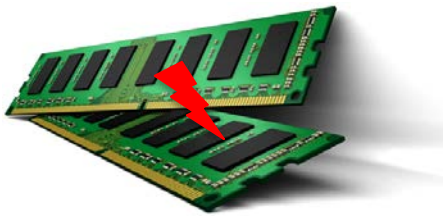
## *Key Observations:*

- **Testing alone cannot detect all possible failures**
- **Combination of ECC and other mitigation techniques is much more effective**
  - **But degrades performance**
- **Testing can help to reduce the ECC strength**
  - **Even when starting with a higher strength ECC**

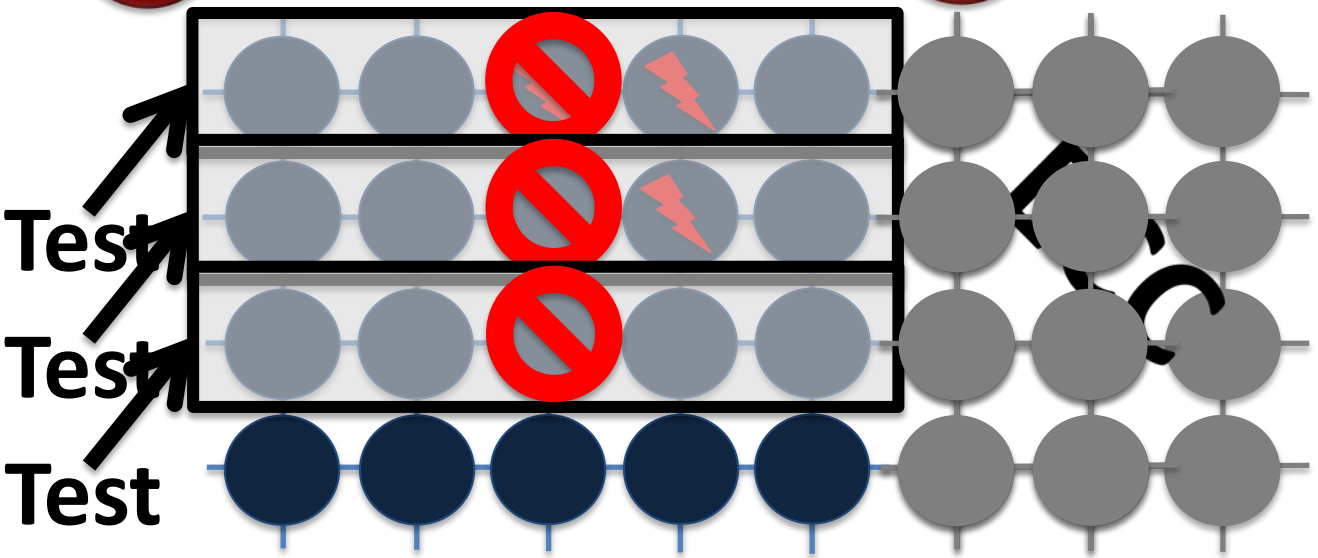
Khan+, "The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study," SIGMETRICS 2014.

# Towards an Online Profiling System

**Initially Protect DRAM with Strong ECC** 1



**Periodically Test Parts of DRAM** 2




**Mitigate errors and reduce ECC** 3

**Run tests periodically after a short interval at smaller regions of memory**

# Handling Variable Retention Time [DSN'15]

- Moinuddin Qureshi, Dae Hyun Kim, Samira Khan, Prashant Nair, and Onur Mutlu, **"AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems"**
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Rio de Janeiro, Brazil, June 2015.
  [Slides (pptx) (pdf)]

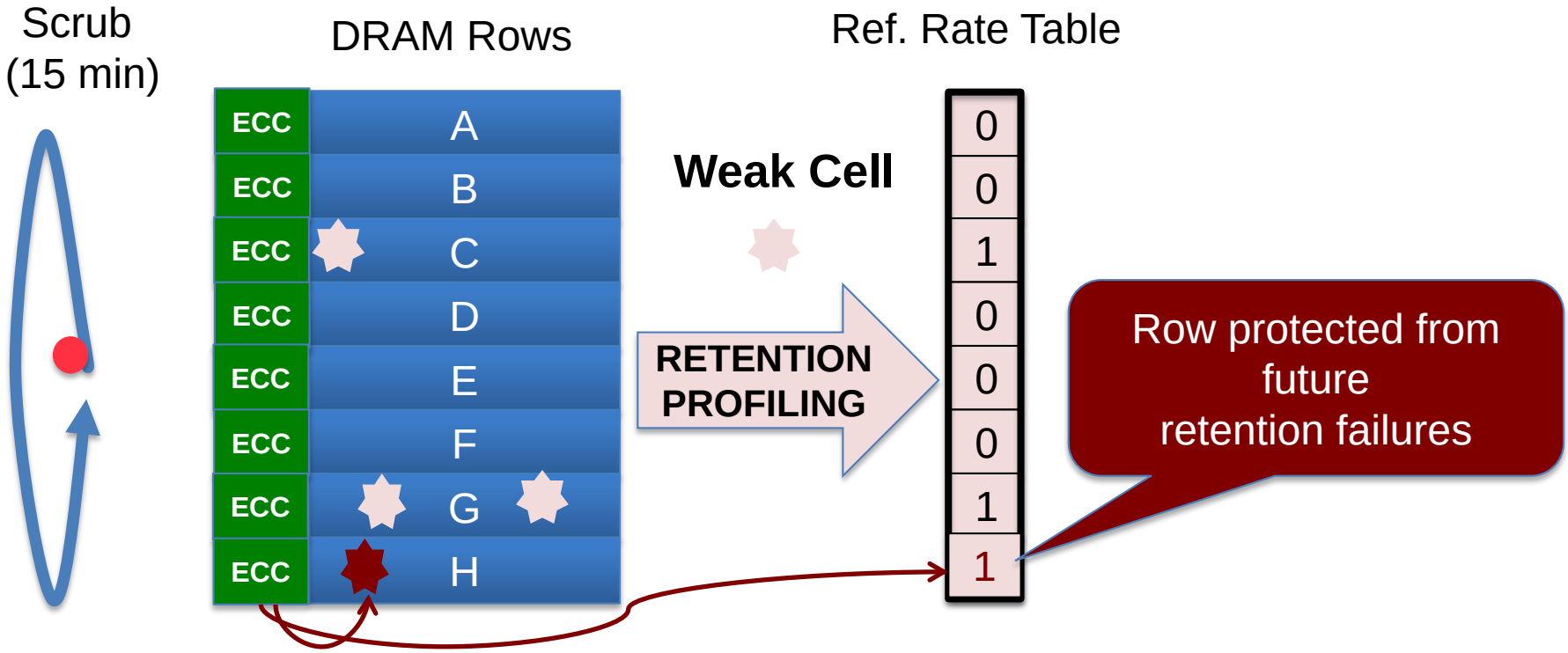
## AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems

Moinuddin K. Qureshi[†] Dae-Hyun Kim[†] Samira Khan[‡] Prashant J. Nair[†] Onur Mutlu[‡]
[†]Georgia Institute of Technology
{moin, dhkim, pnair6}@ece.gatech.edu
[‡]Carnegie Mellon University
{samirakhan, onur}@cmu.edu

# AVATAR

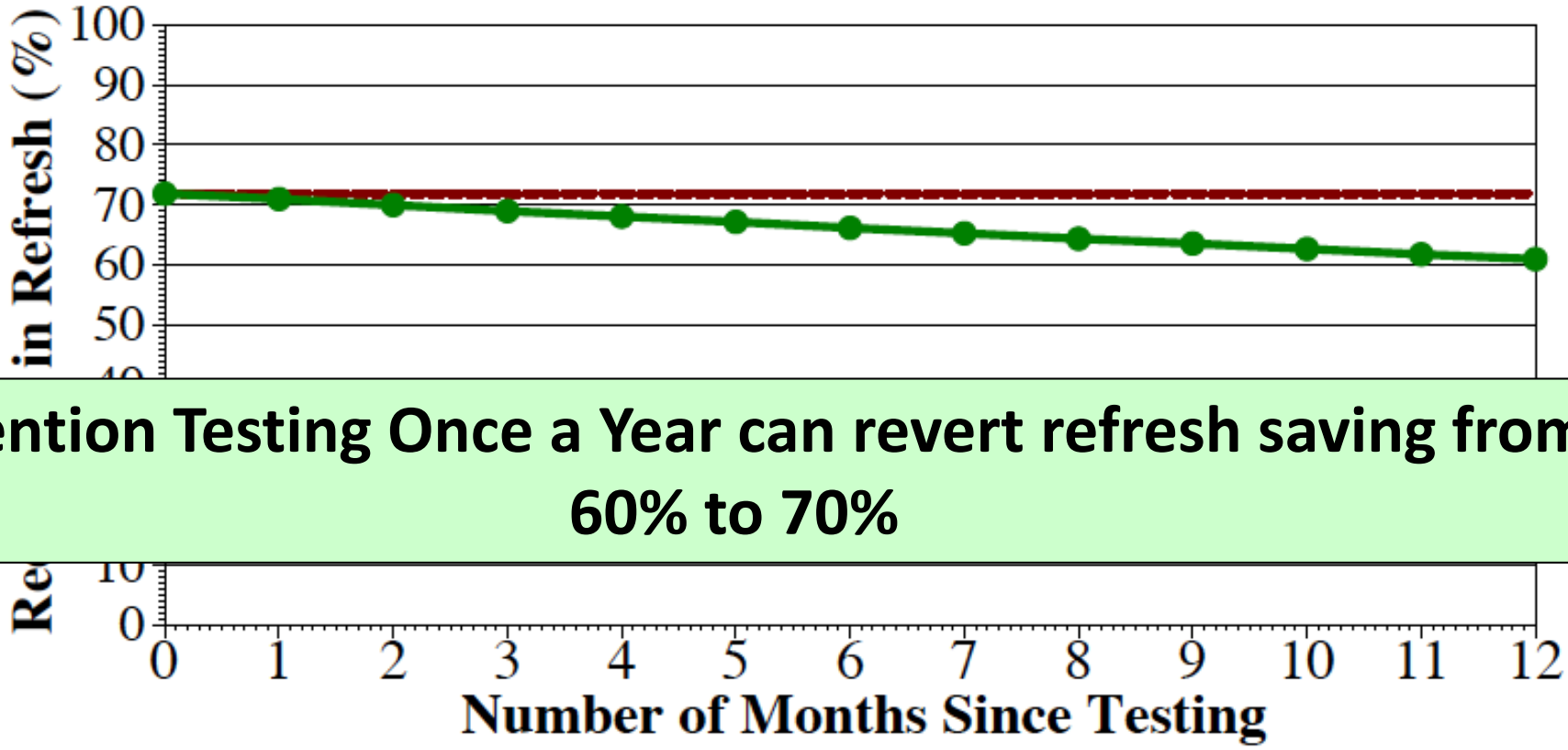
Insight: Avoid retention failures ➔ Upgrade row on ECC error

Observation: Rate of VRT >> Rate of soft error (50x-2500x)




**AVATAR mitigates VRT by increasing refresh rate on error**

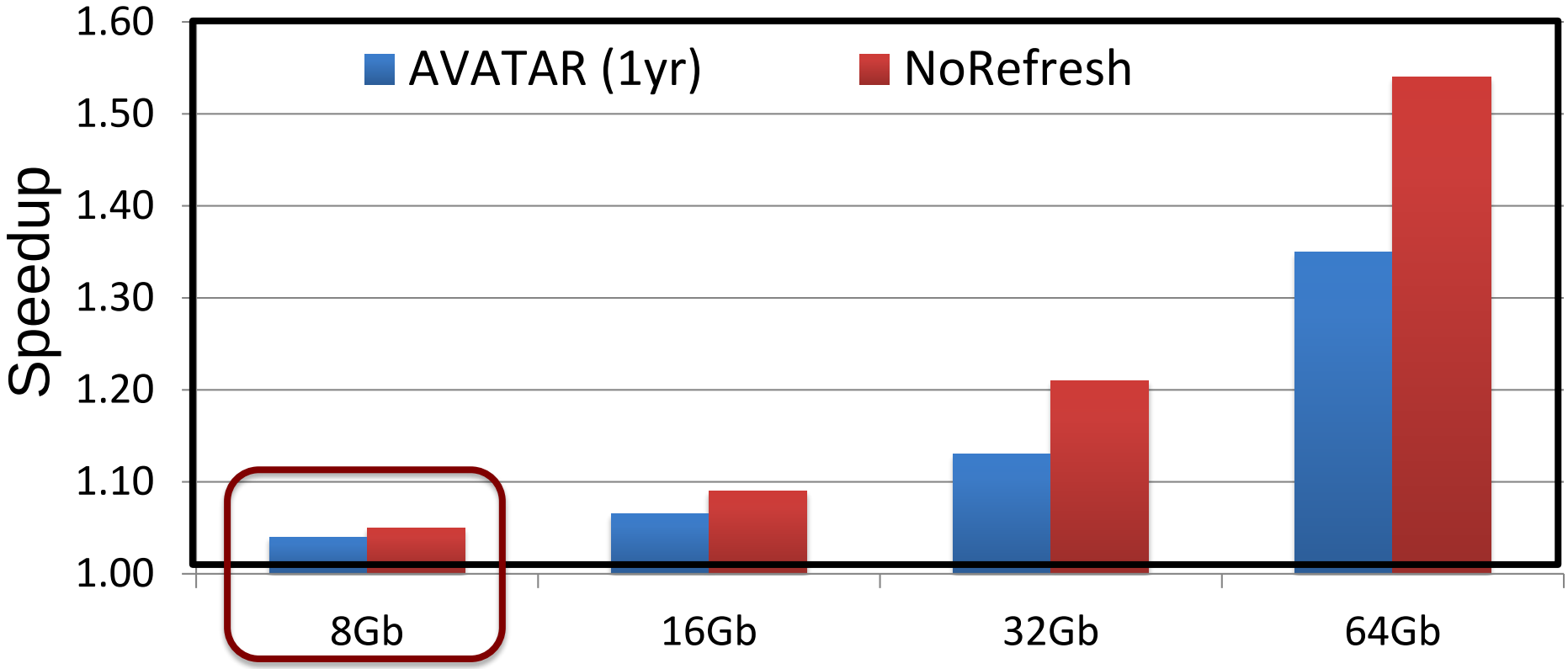
# RESULTS: REFRESH SAVINGS




**Retention Testing Once a Year can revert refresh saving from 60% to 70%**

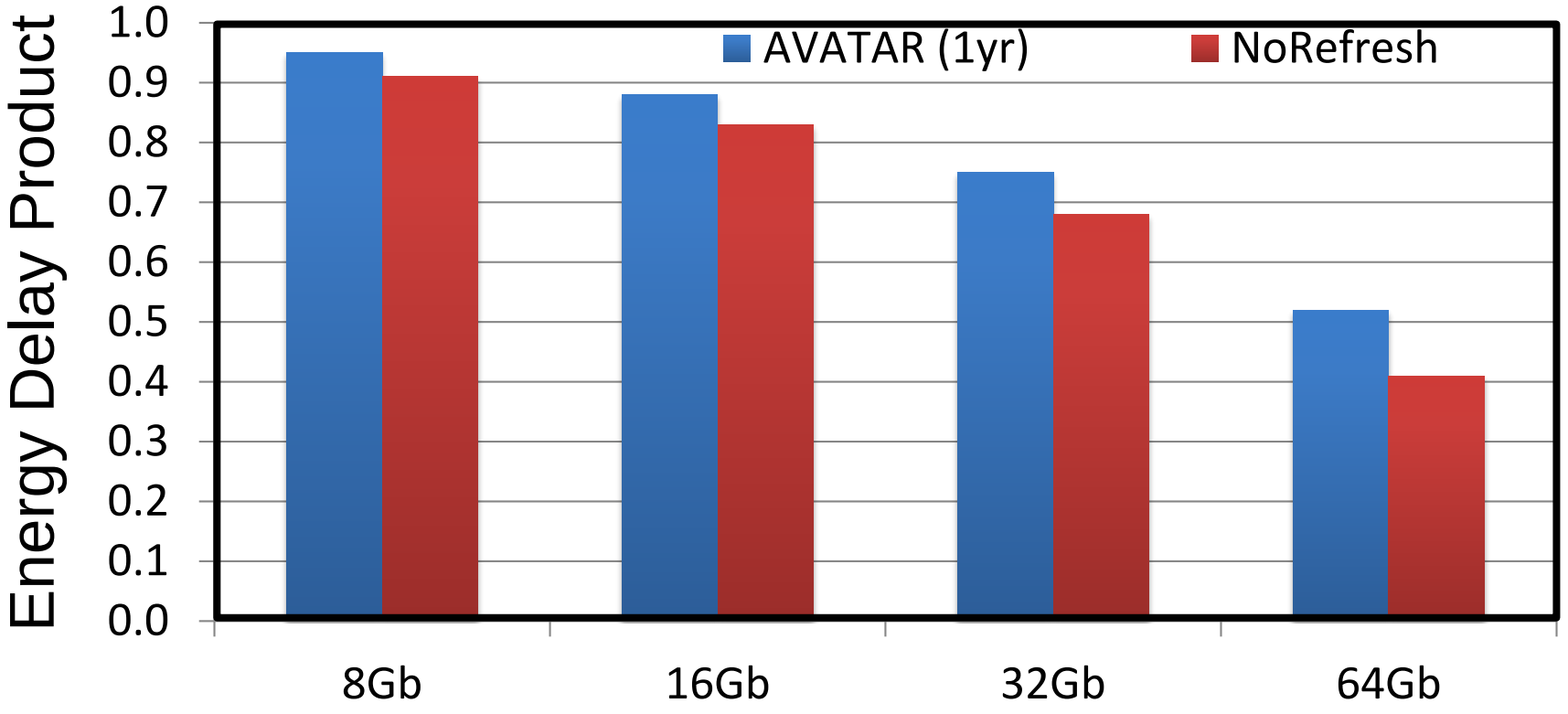
**AVATAR reduces refresh by 60%-70%, similar to multi rate refresh but with VRT tolerance**

# SPEEDUP




**AVATAR gets 2/3rd the performance of NoRefresh. More gains at higher capacity nodes**

# ENERGY DELAY PRODUCT




**AVATAR reduces EDP,**
**Significant reduction at higher capacity nodes**

# Handling Data-Dependent Failures [DSN'16]

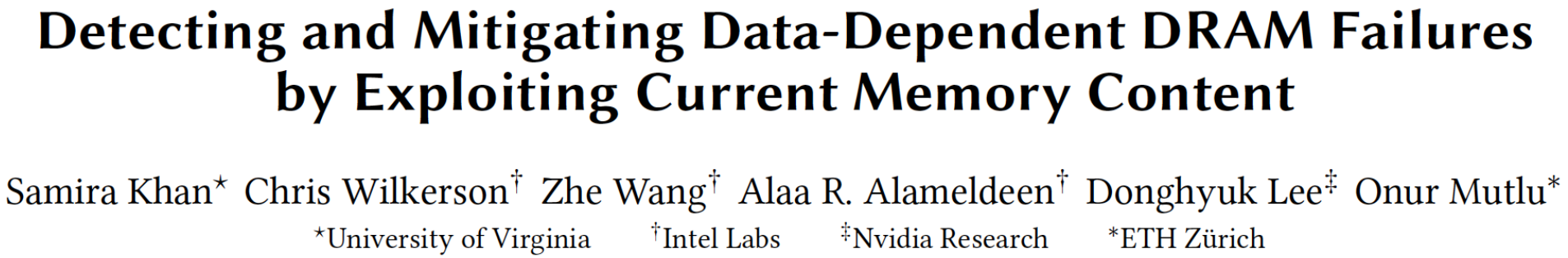
- Samira Khan, Donghyuk Lee, and Onur Mutlu,
  **"PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM"**
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Toulouse, France, June 2016.
  [Slides (pptx) (pdf)]

**PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM**

Samira Khan* Donghyuk Lee†‡ Onur Mutlu*†
*University of Virginia †Carnegie Mellon University ‡Nvidia *ETH Zürich

# Handling Data-Dependent Failures [MICRO'17]

- Samira Khan, Chris Wilkerson, Zhe Wang, Alaa R. Alameldeen, Donghyuk Lee, and Onur Mutlu,
  **"Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content"**
  *Proceedings of the 50th International Symposium on Microarchitecture (MICRO)*, Boston, MA, USA, October 2017.
  [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]



# Handling Both DPD and VRT [ISCA'17]

- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  **"The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"**
  *Proceedings of the 44th International Symposium on Computer Architecture (ISCA)*, Toronto, Canada, June 2017.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

- First experimental analysis of (mobile) LPDDR4 chips
- Analyzes the complex tradeoff space of retention time profiling
- Idea: enable fast and robust profiling at higher refresh intervals & temperatures

**The Reach Profiler (REAPER):**
**Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions**

Minesh Patel[§‡] Jeremie S. Kim[‡§] Onur Mutlu[§‡]
[§]ETH Zürich [‡]Carnegie Mellon University

# *The Reach Profiler (REAPER):*

## Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions

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

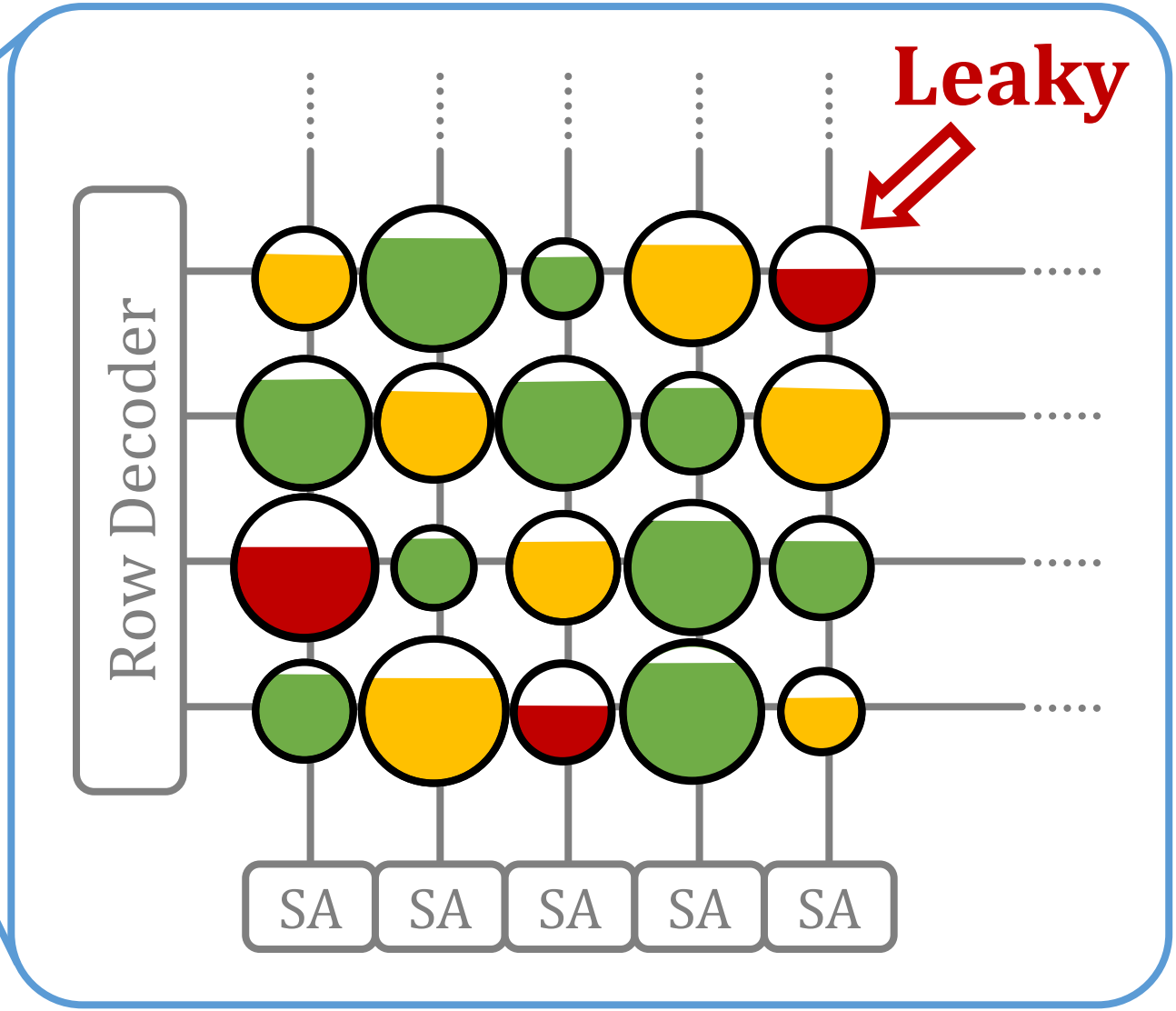
**Onur Mutlu**

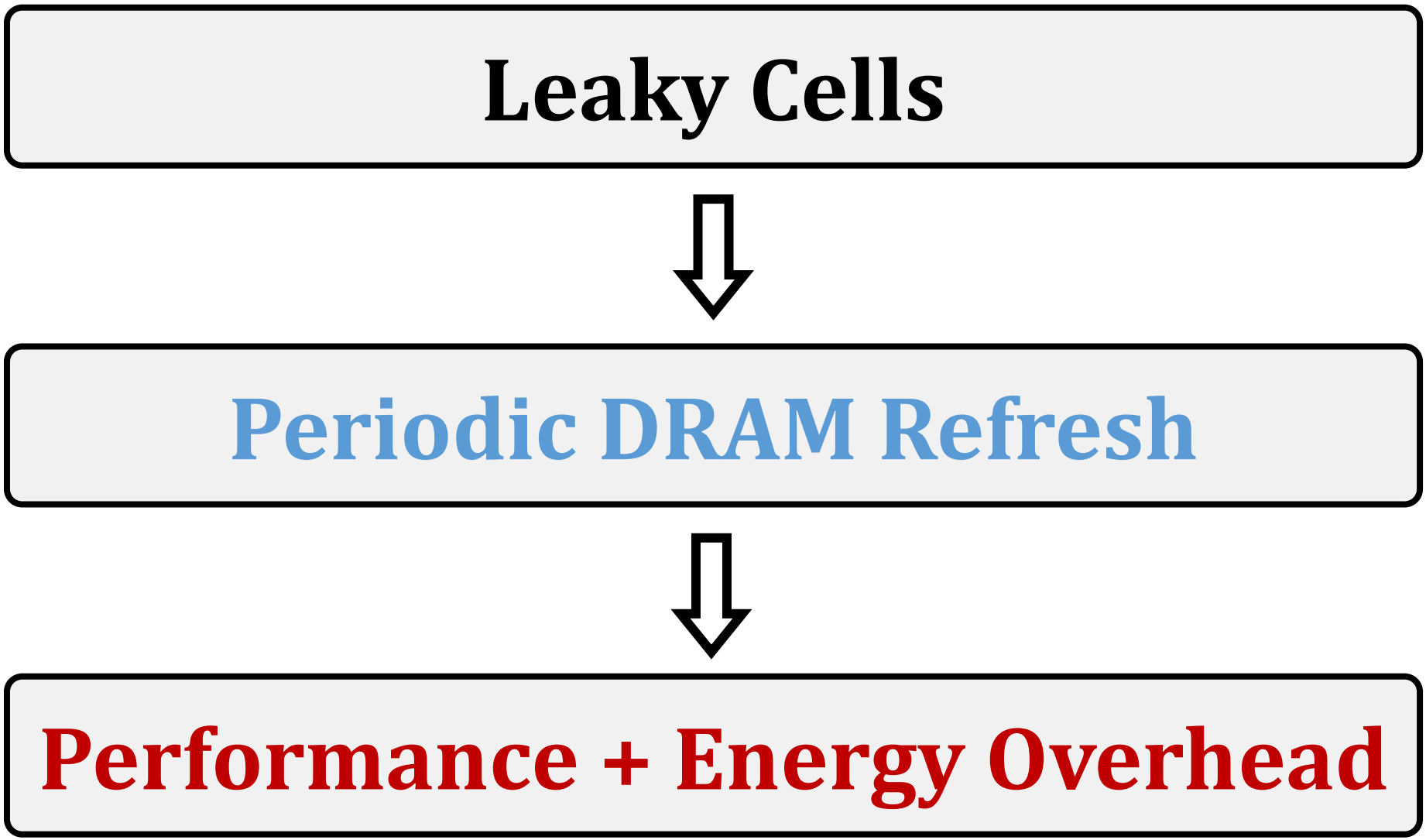

SAFARI

ETH Zürich

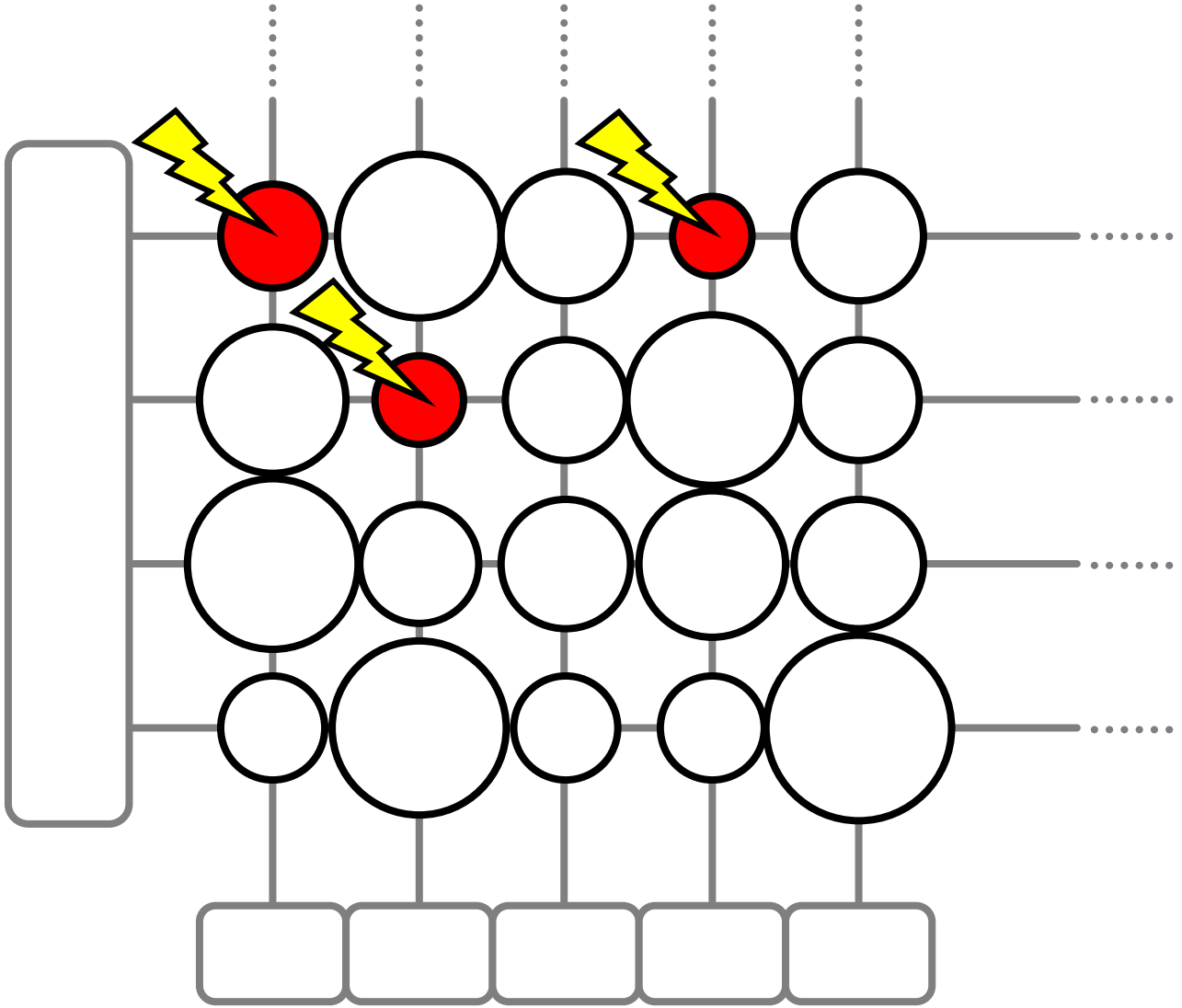
Carnegie Mellon

# DRAM




Leaky Cells
Periodic DRAM Refresh
Performance + Energy Overhead

# Goal: find *all* retention failures for a refresh interval $T$ > default (64ms)



**Process, voltage, temperature**

**Variable retention time**

**Data pattern dependence**

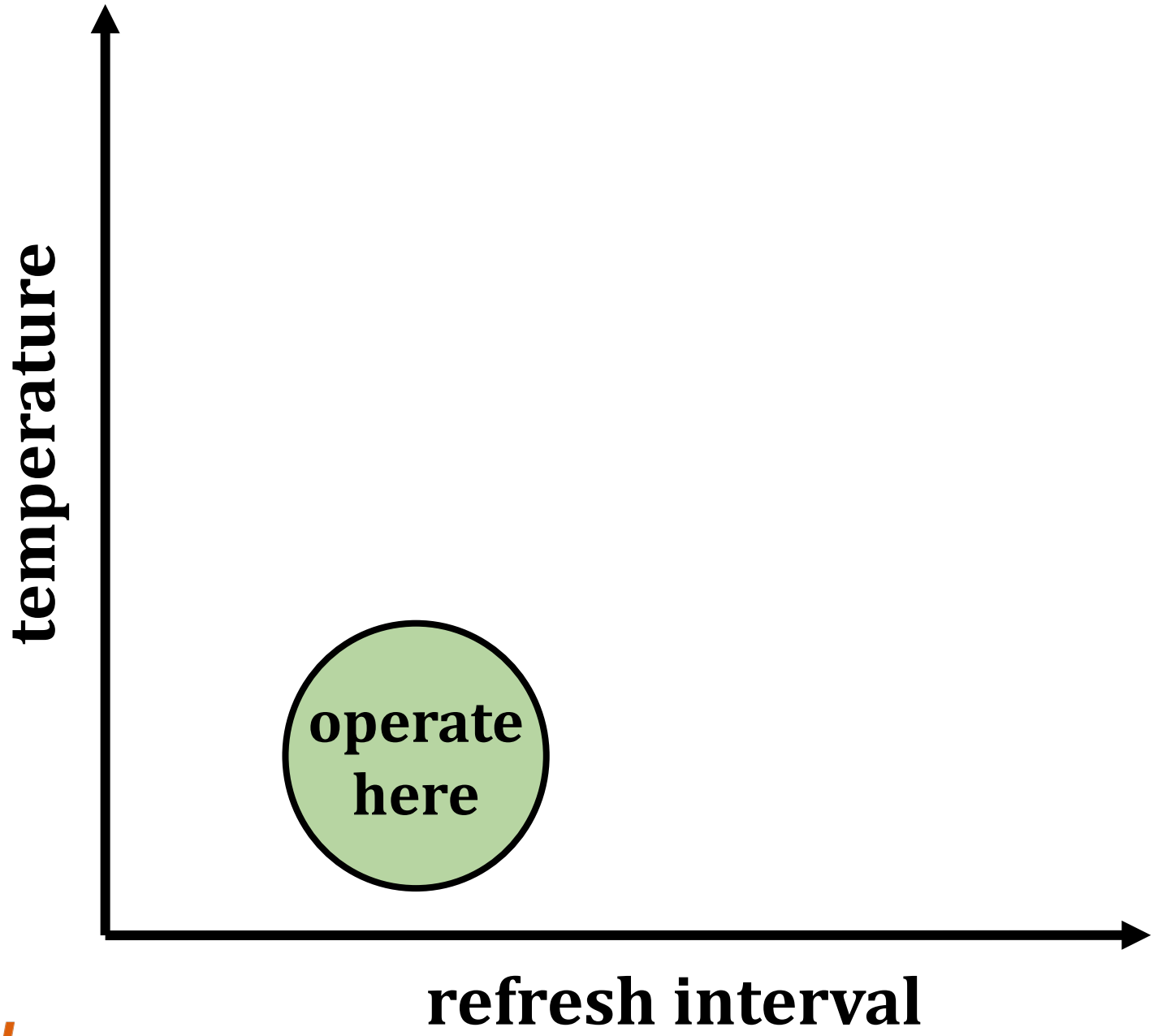
# Characterization of 368 LPDDR4 DRAM Chips

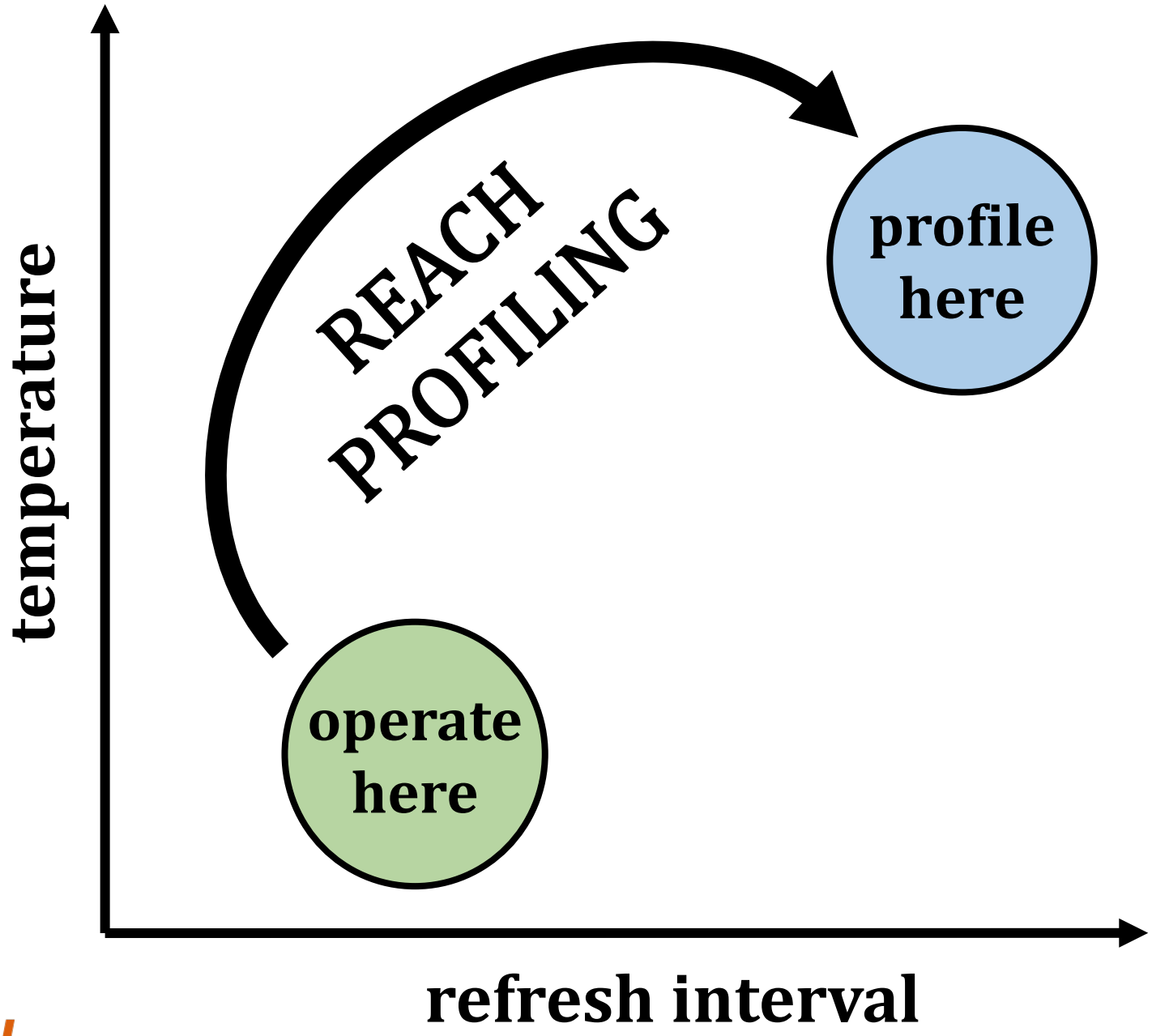
①

Cells are **more likely to fail** at an **increased (refresh interval | temperature)**

②

**Complex tradeoff space** between profiling **(speed & coverage & false positives)**


temperature
operate here
refresh interval


temperature
REACH
PROFILING
profile
here
operate
here
refresh interval

# Reach Profiling

**A new DRAM retention failure profiling methodology**

+ **Faster** and **more reliable** than current approaches

+ Enables **longer refresh intervals**

# REAPER Outline

1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. **LPDDR4 Characterization**
5. Reach Profiling
6. End-to-end Evaluation

# Experimental Infrastructure

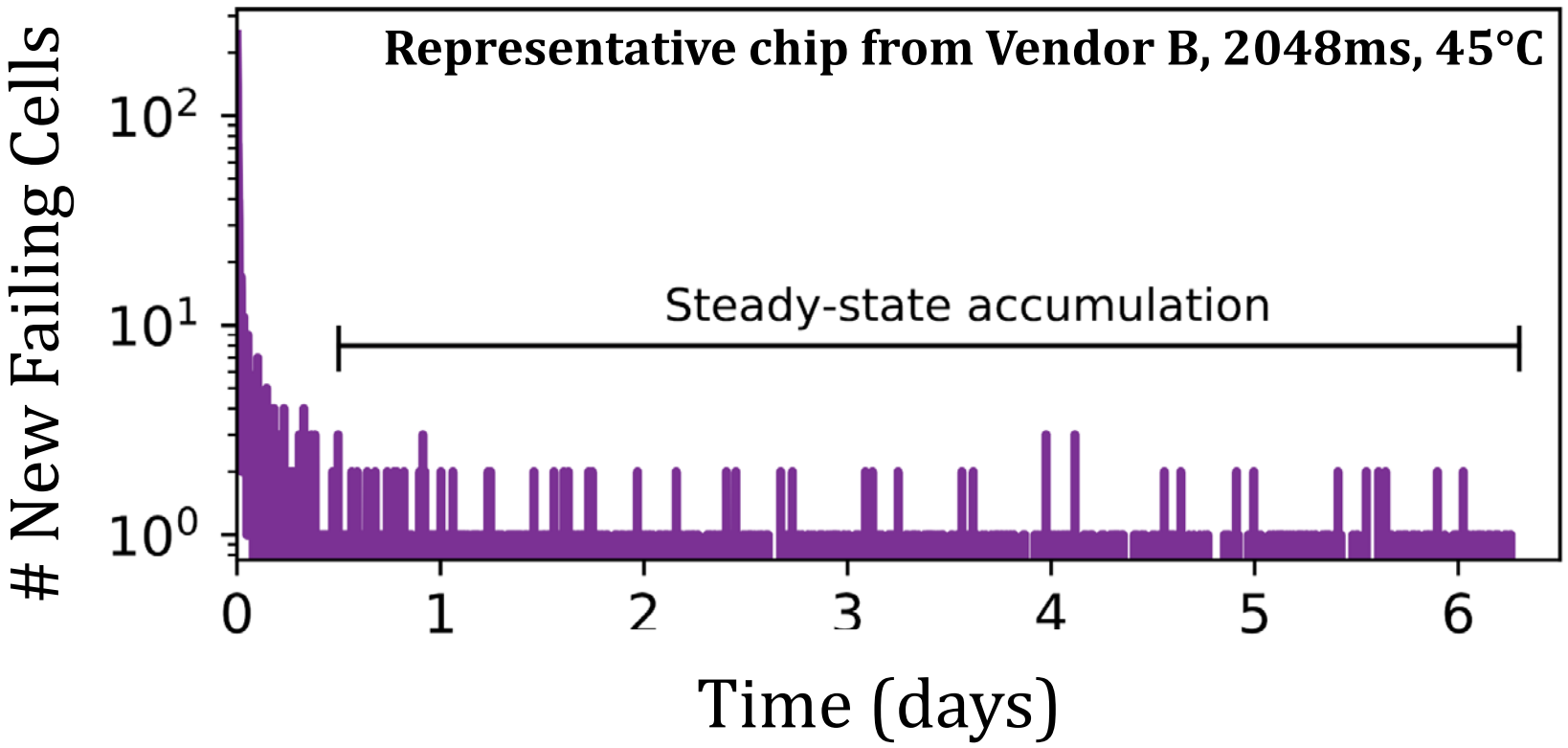
- **368 2y-nm LPDDR4 DRAM chips**
  - 4Gb chip size
  - From 3 major DRAM vendors

- **Thermally controlled testing chamber**
  - Ambient temperature range: {40°C – 55°C} ± 0.25°C
  - DRAM temperature is held at 15°C above ambient

# LPDDR4 Studies

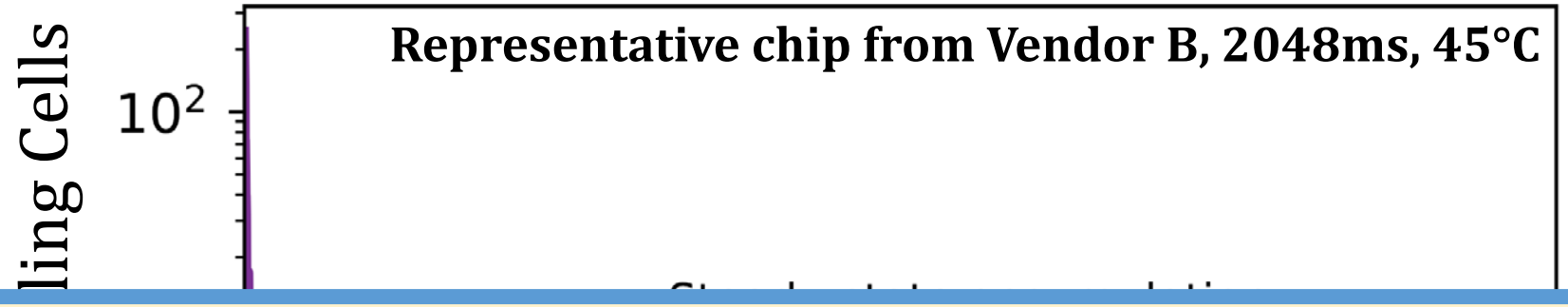
1. Temperature
2. Data Pattern Dependence
3. Retention Time Distributions
4. **Variable Retention Time**
5. **Individual Cell Characterization**

# Long-term Continuous Profiling




- New failing cells continue to appear over time
  - Attributed to **variable retention time (VRT)**
- The set of failing cells changes over time

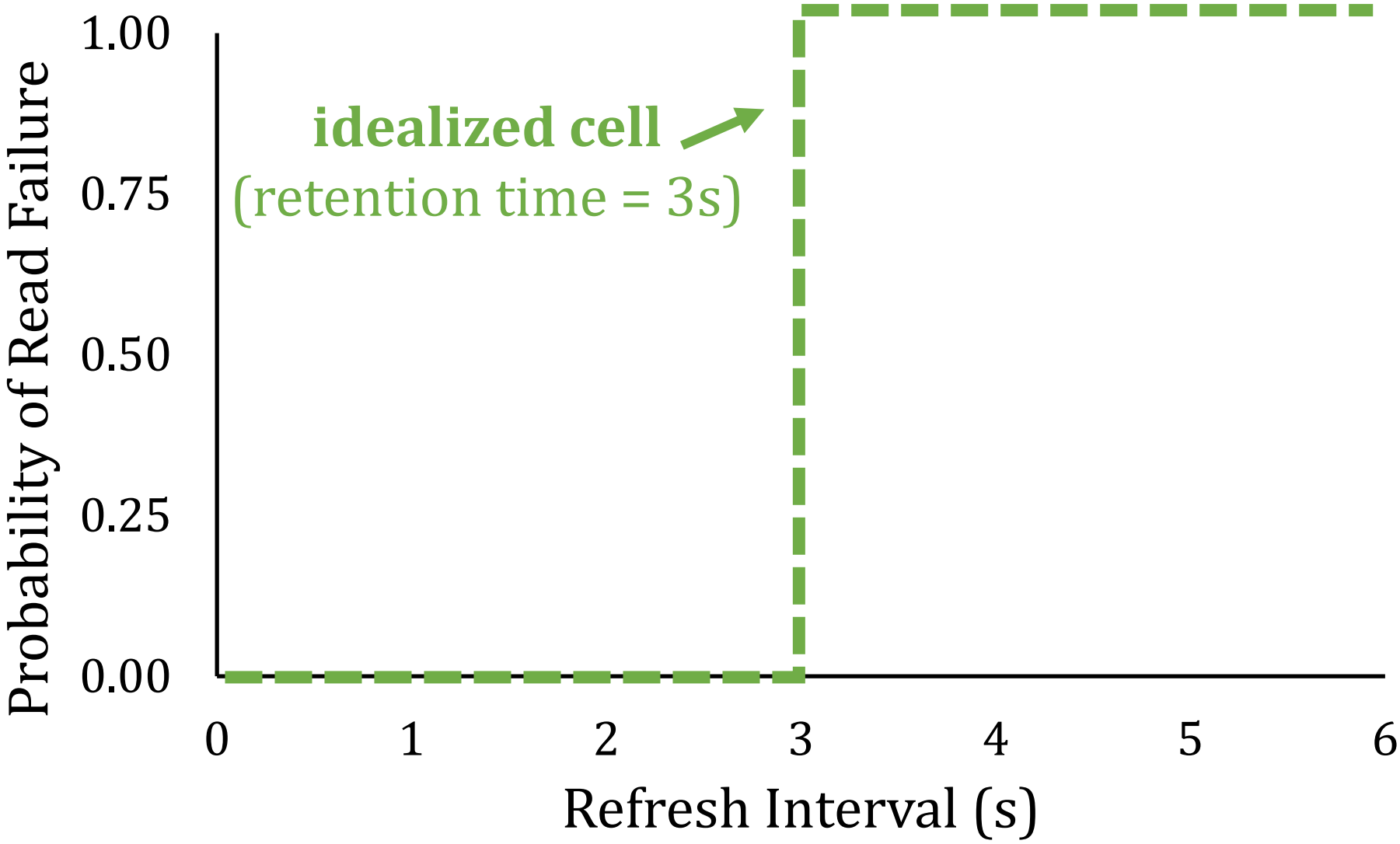
# Long-term Continuous Profiling



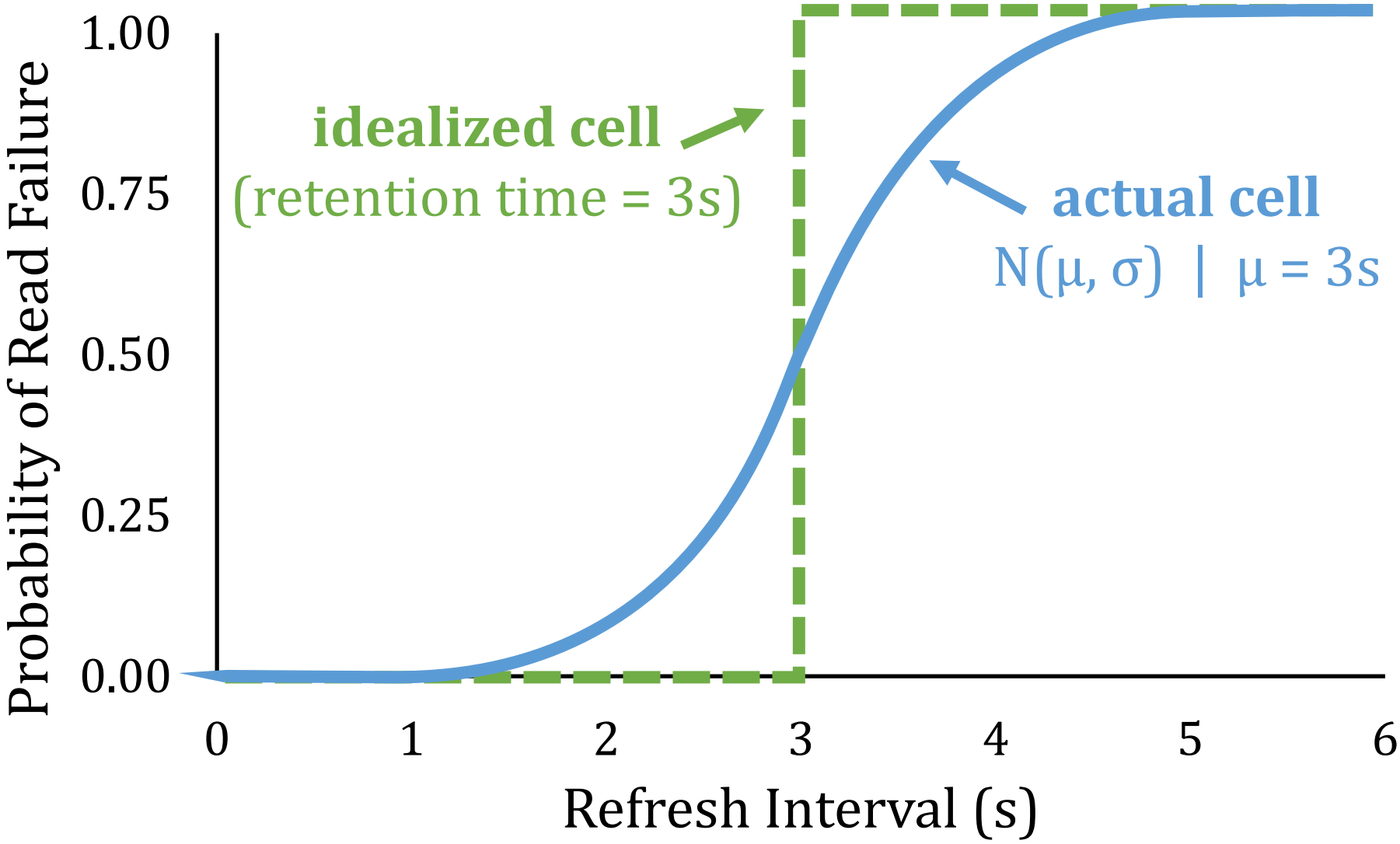

**Error correction codes (ECC) and online profiling are *necessary* to manage new failing cells**

- New failing cells continue to appear over time
  - Attributed to **variable retention time (VRT)**
- The set of failing cells changes over time

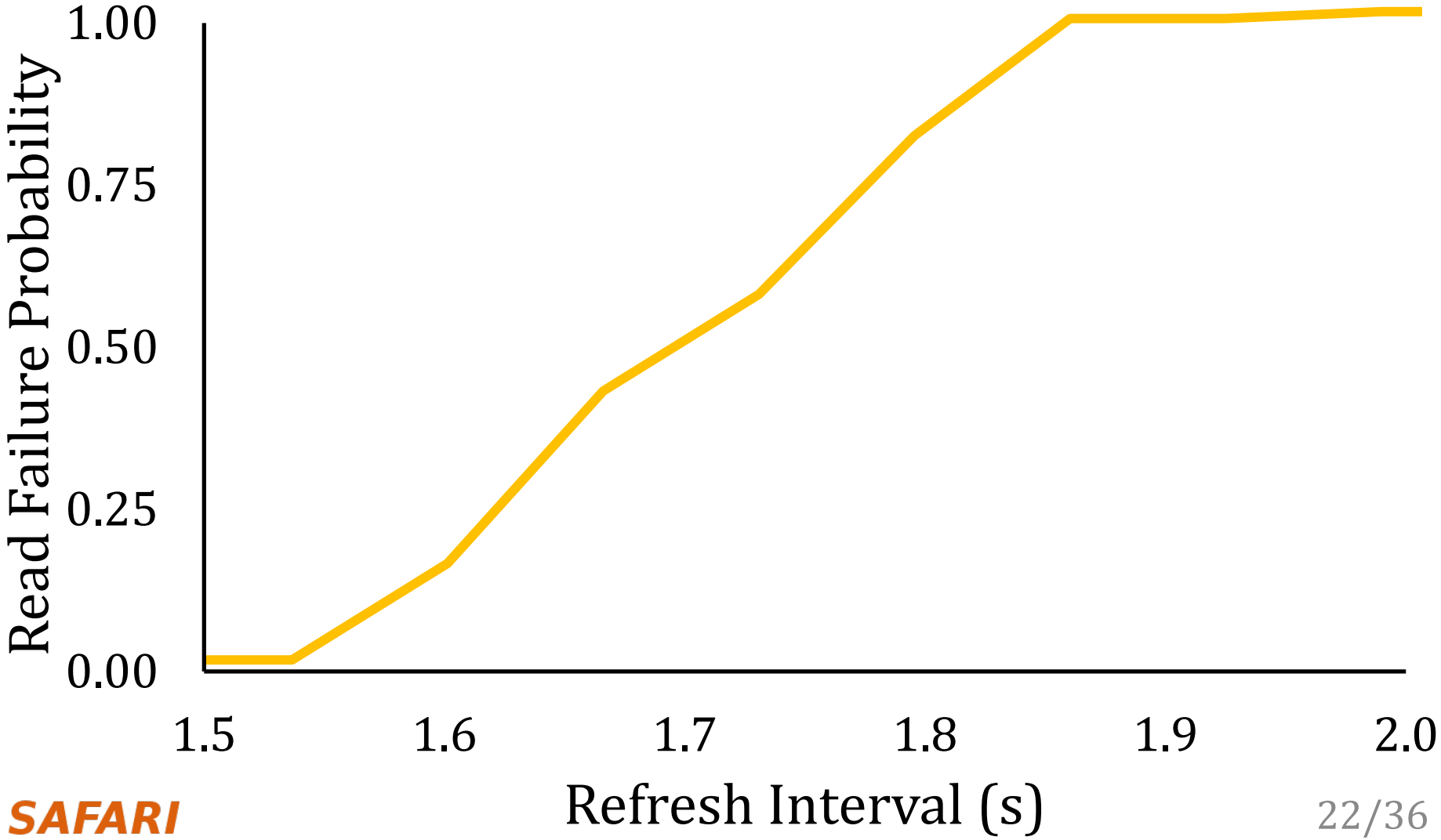
# Single-cell Failure Probability (Cartoon)



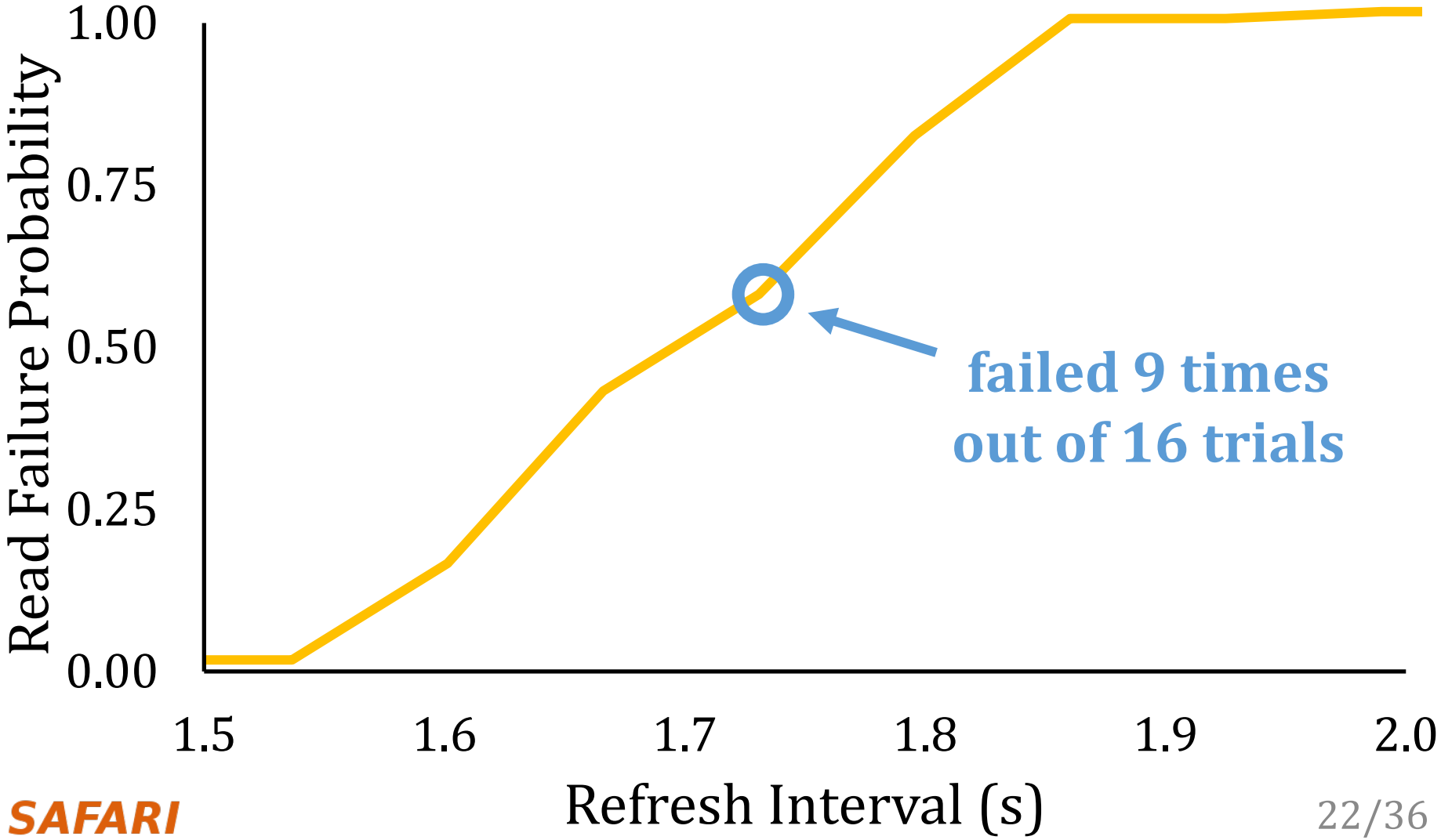
# Single-cell Failure Probability (Cartoon)



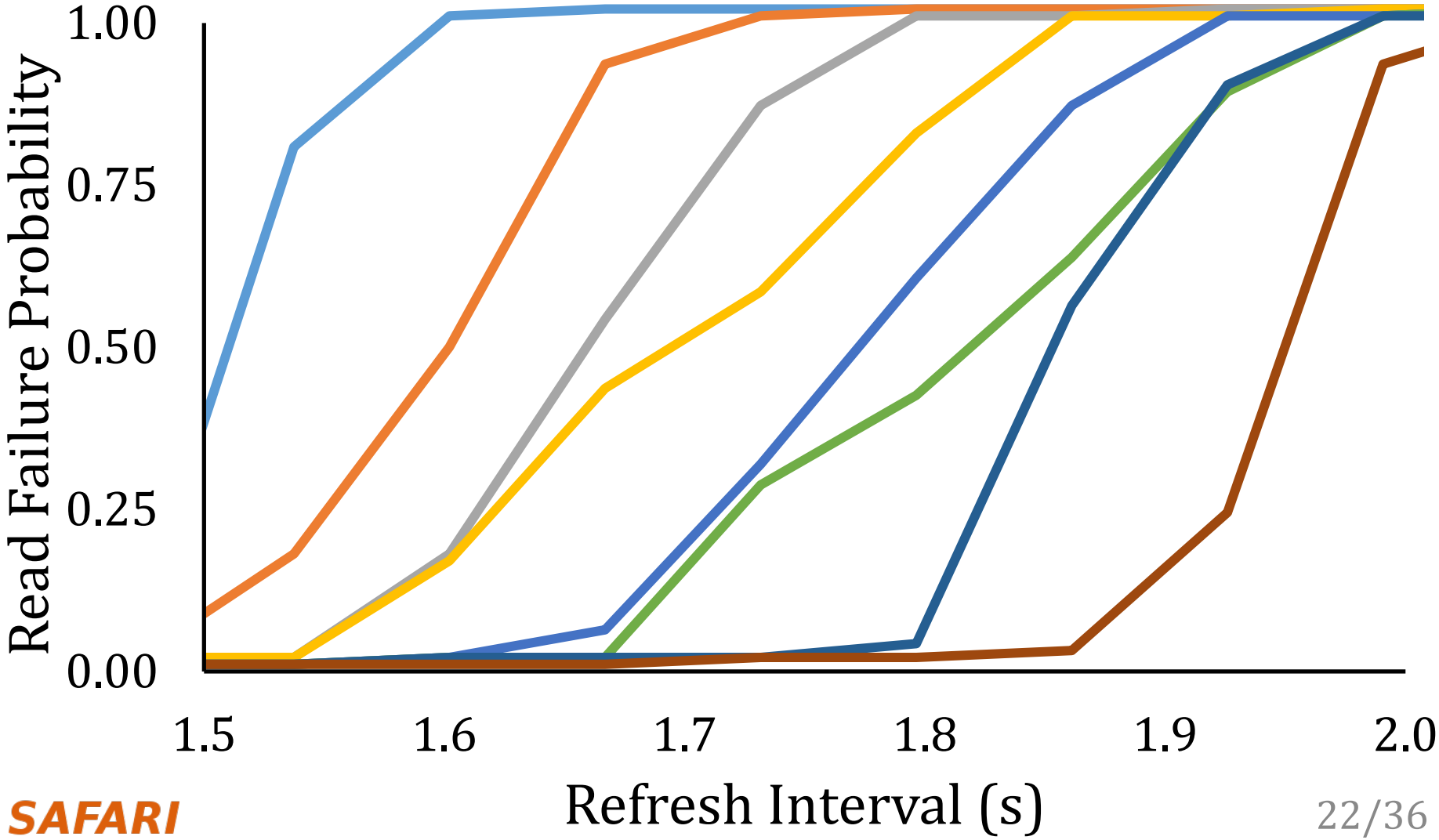
# Single-cell Failure Probability (Real)



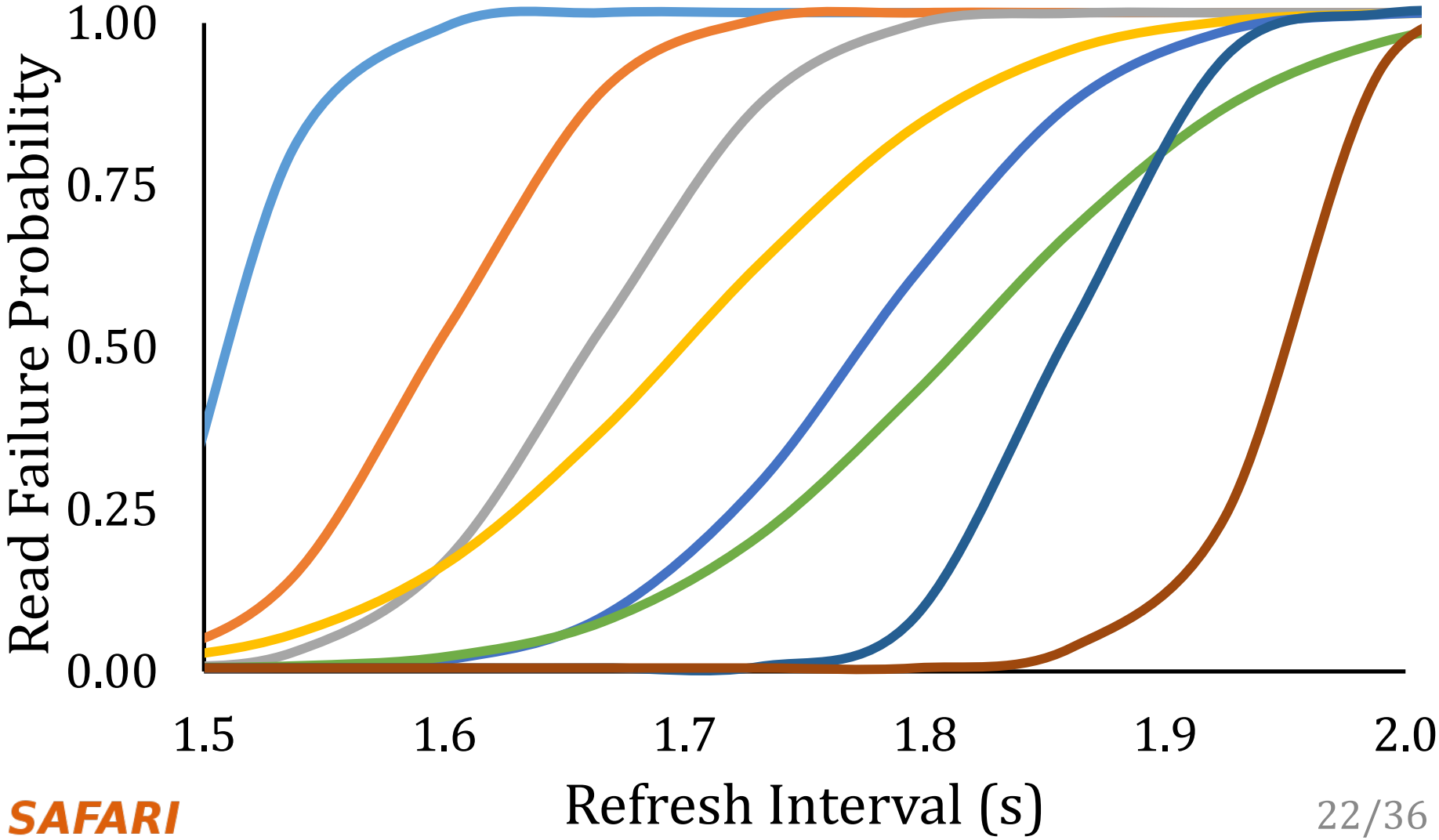
# Single-cell Failure Probability (Real)



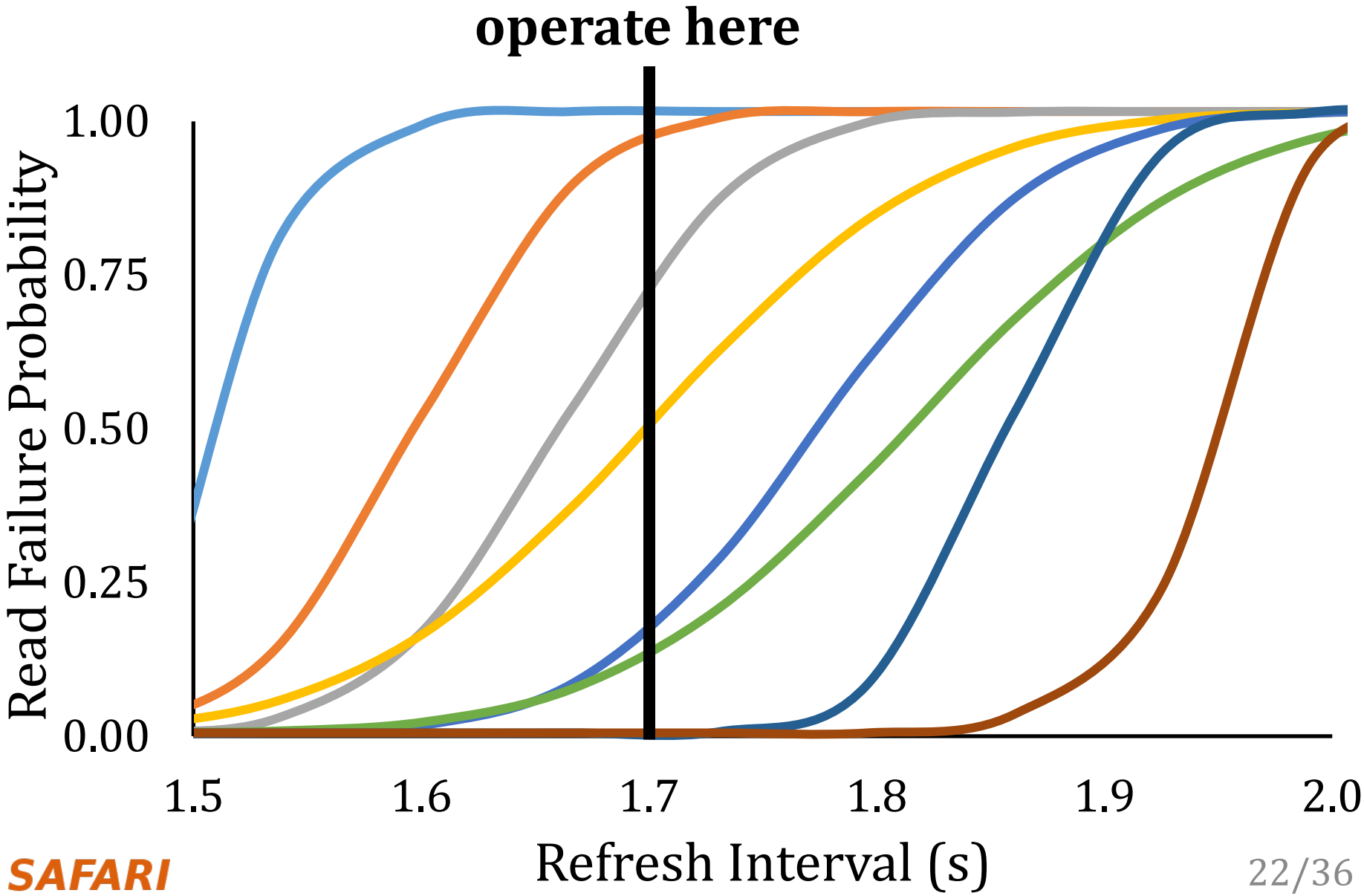
# Single-cell Failure Probability (Real)



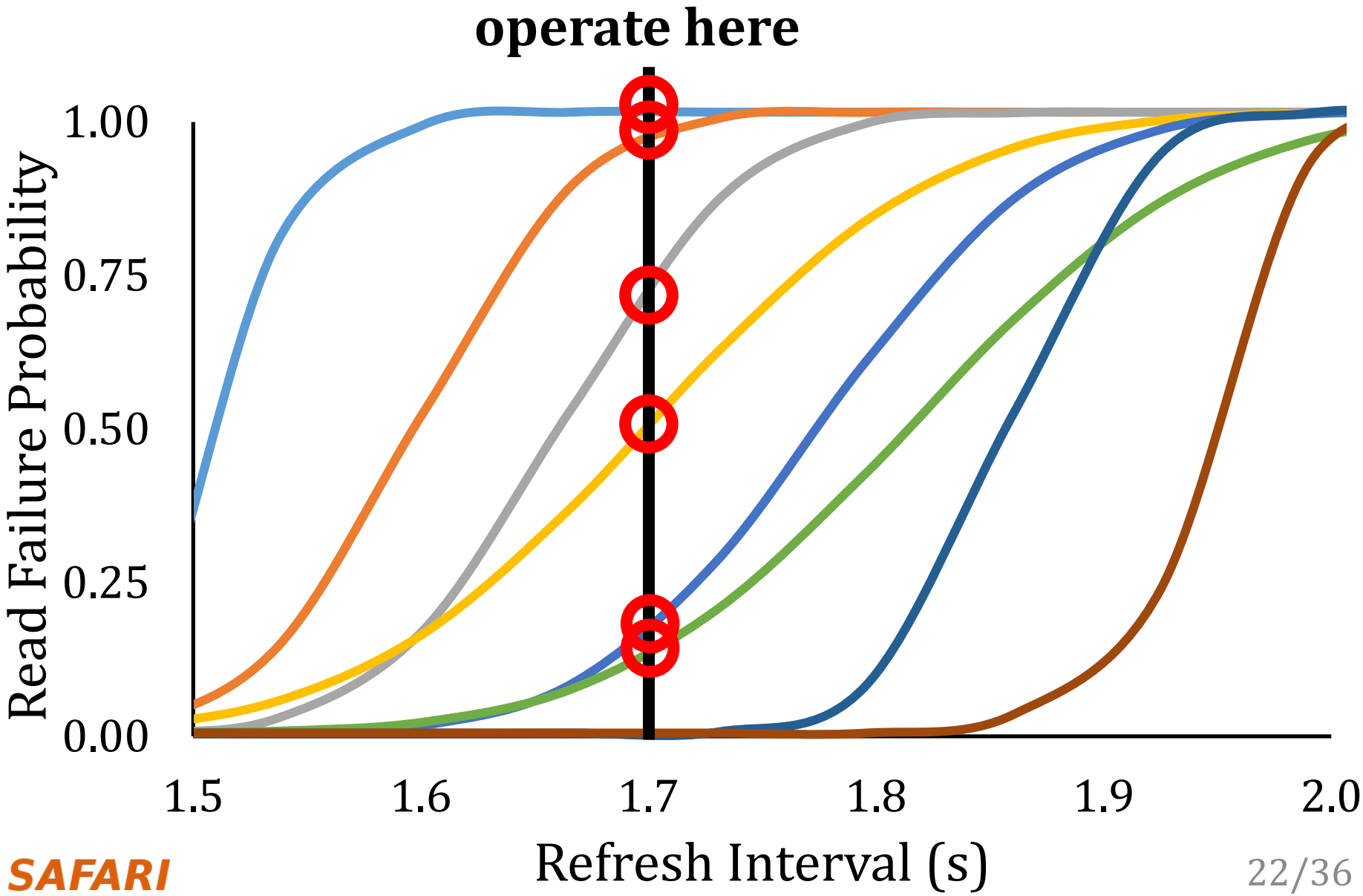
# Single-cell Failure Probability (Real)



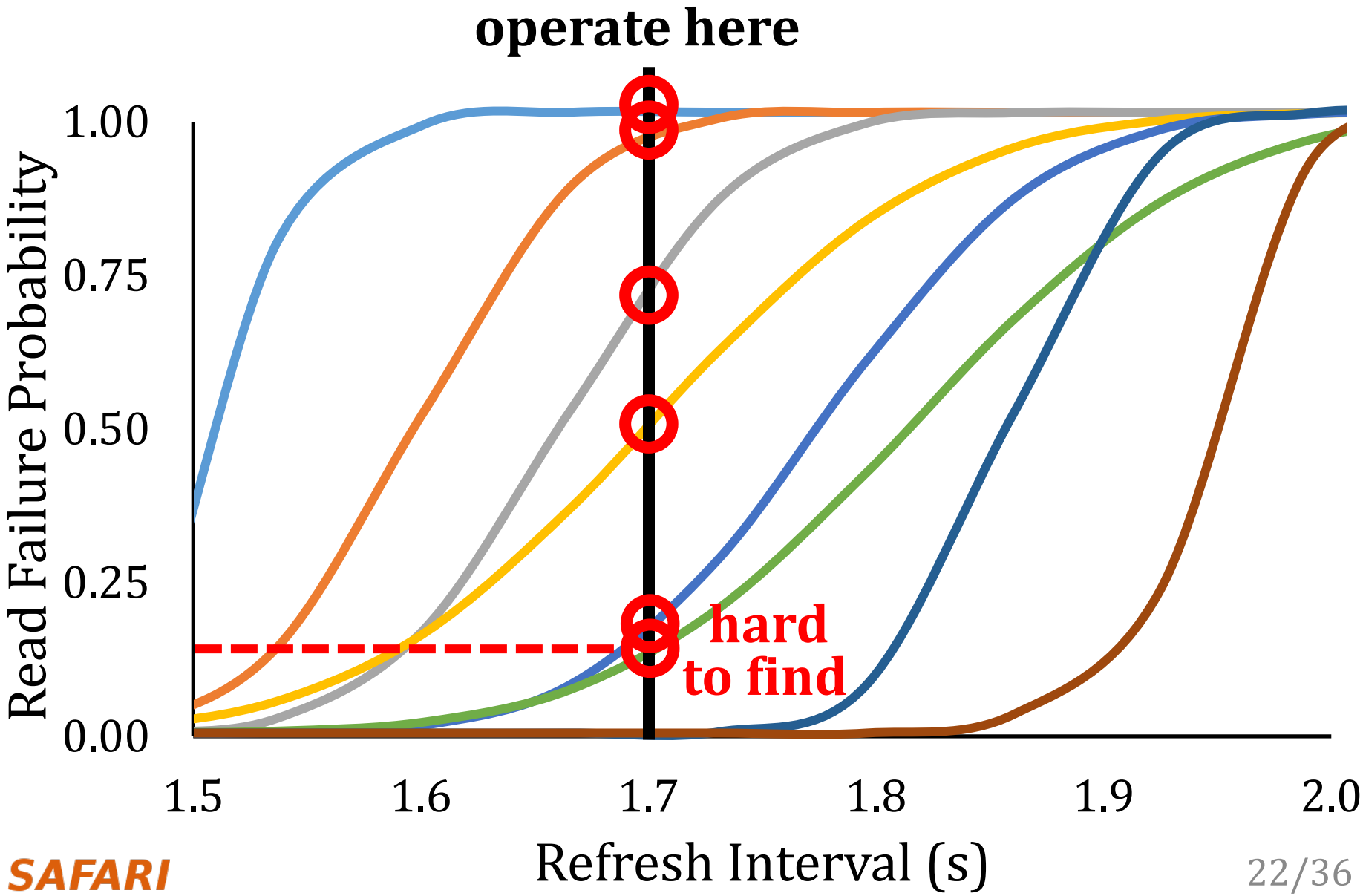
# Single-cell Failure Probability (Real)



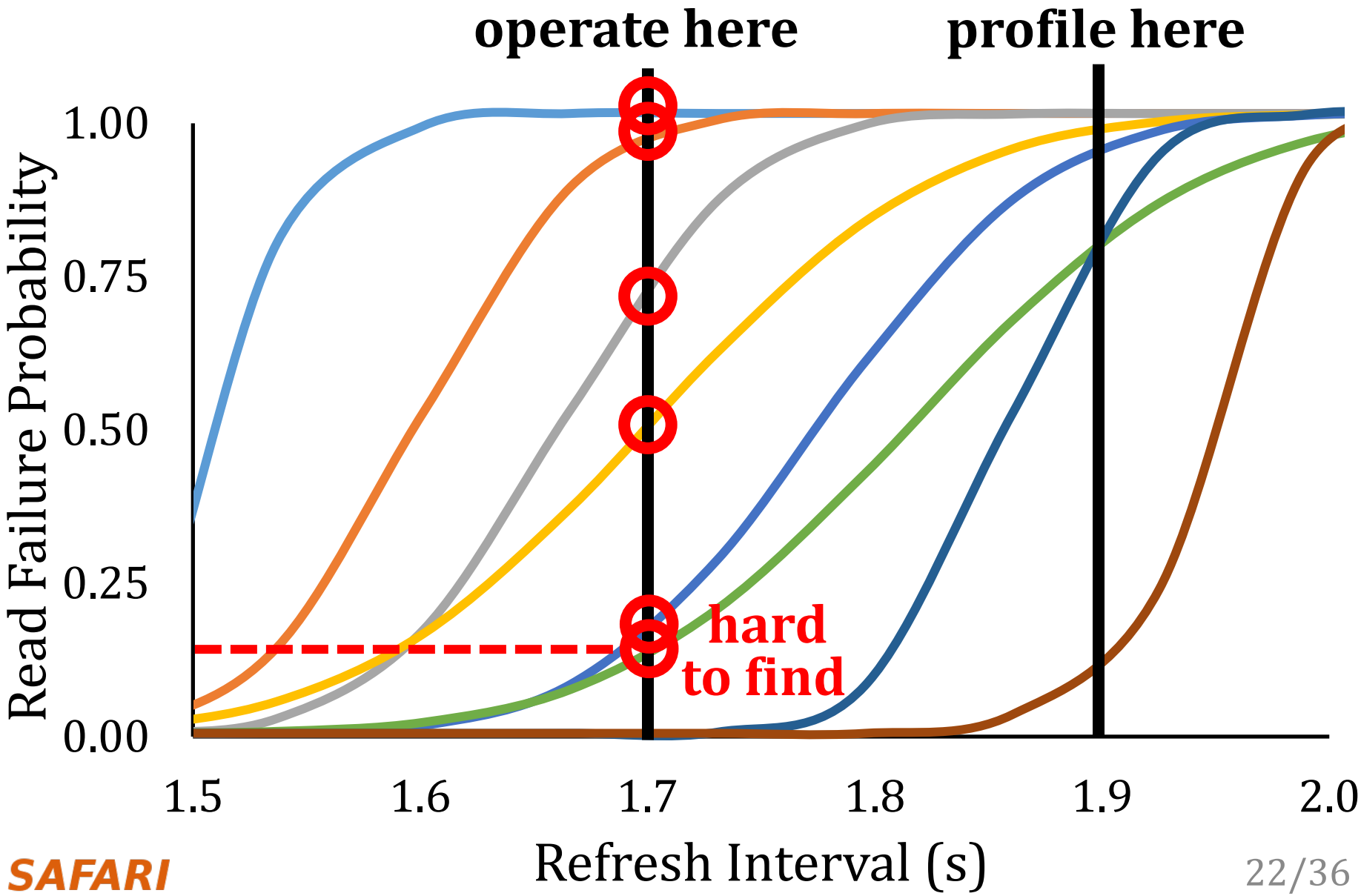
# Single-cell Failure Probability (Real)



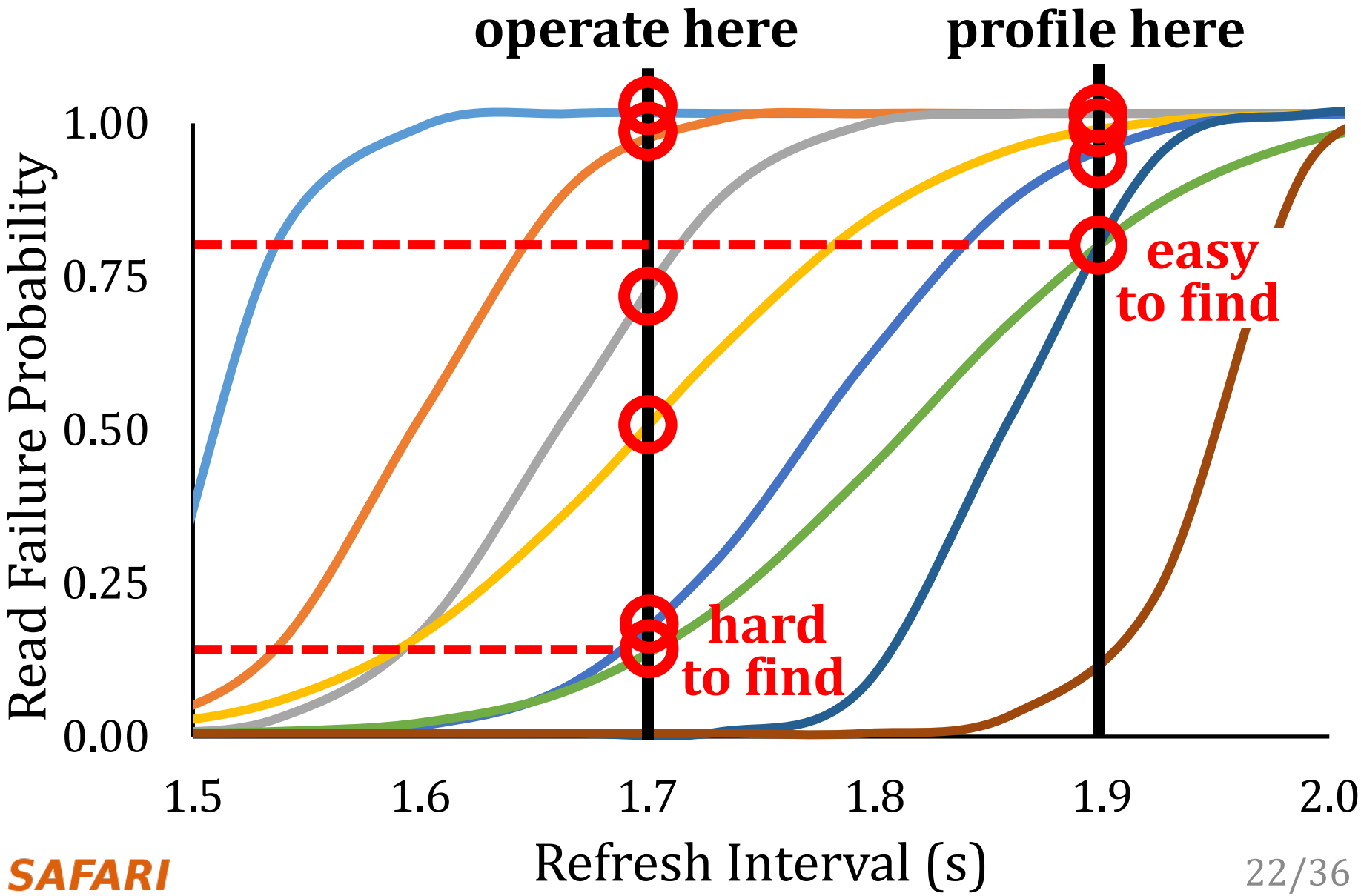
# Single-cell Failure Probability (Real)



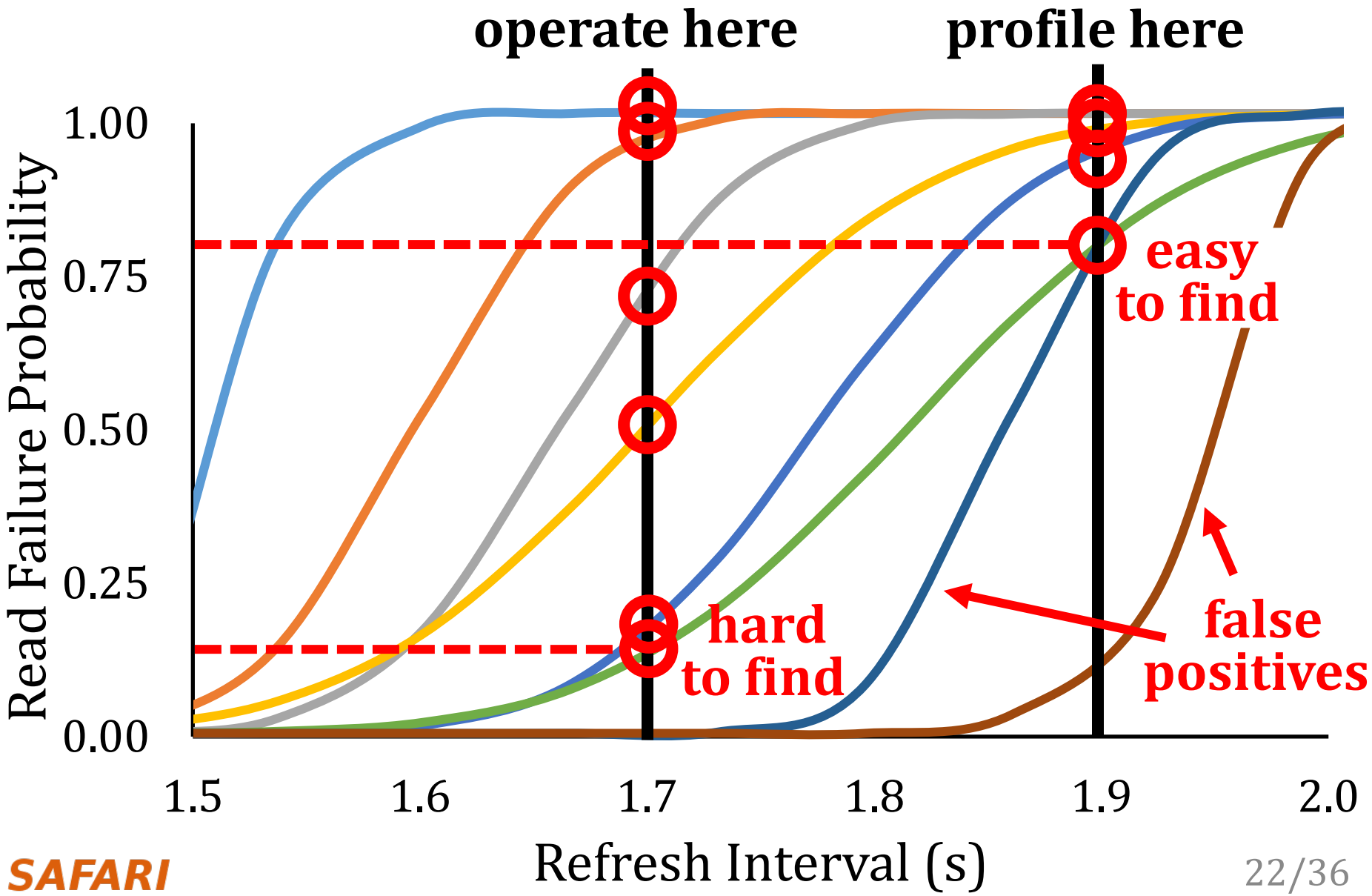
# Single-cell Failure Probability (Real)



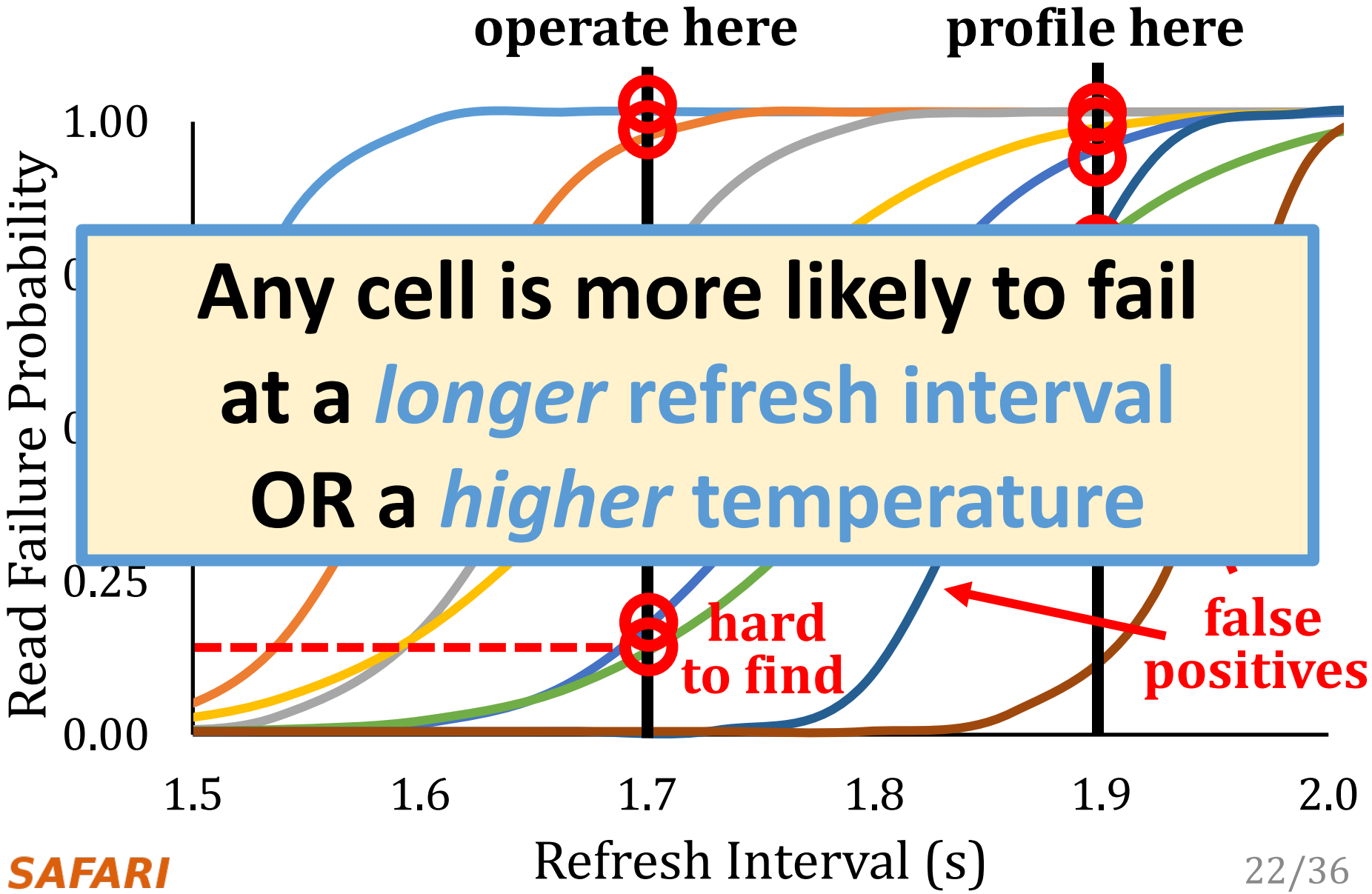
# Single-cell Failure Probability (Real)



# Single-cell Failure Probability (Real)



# Single-cell Failure Probability (Real)

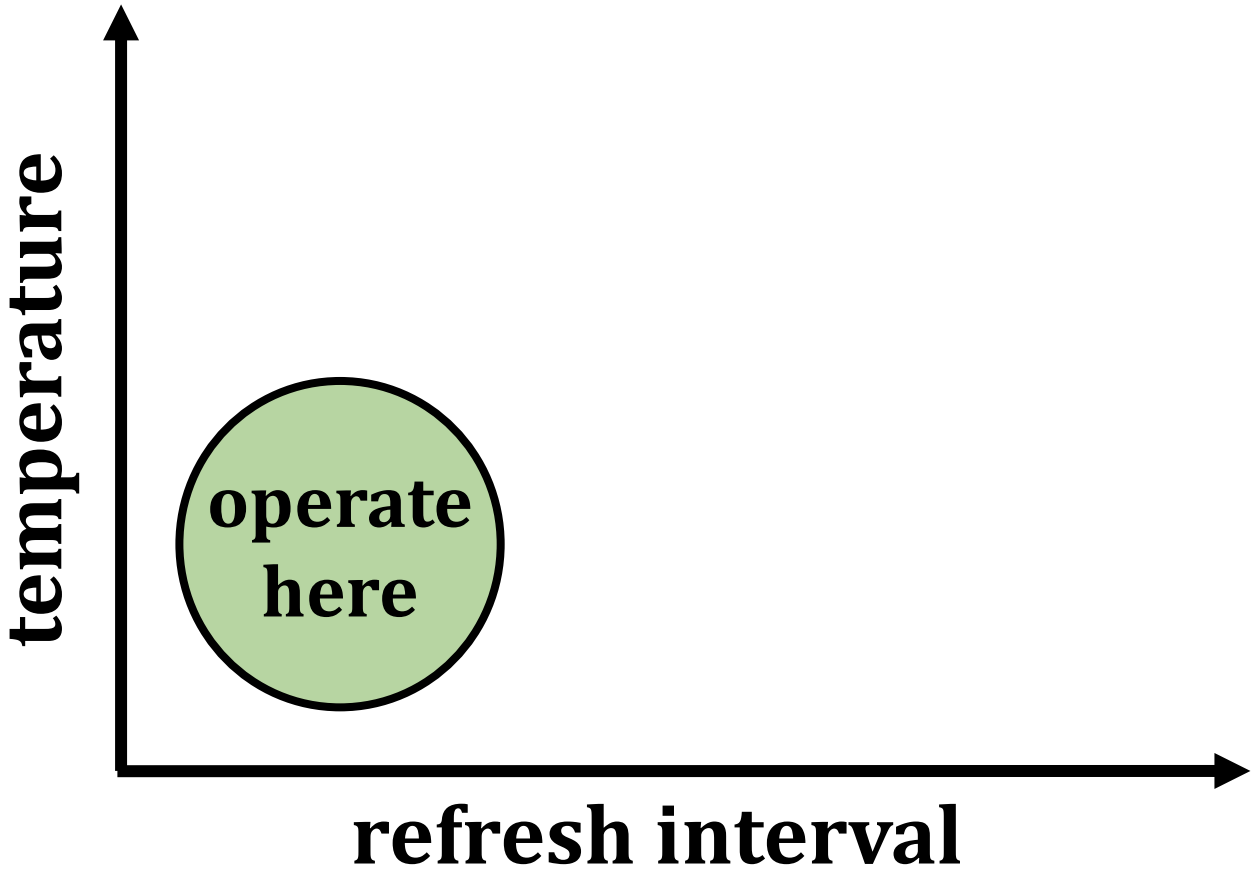


# REAPER Outline

1. DRAM Refresh Background
2. Failure Profiling Challenges
3. Current Approaches
4. LPDDR4 Characterization
5. **Reach Profiling**
6. End-to-end Evaluation

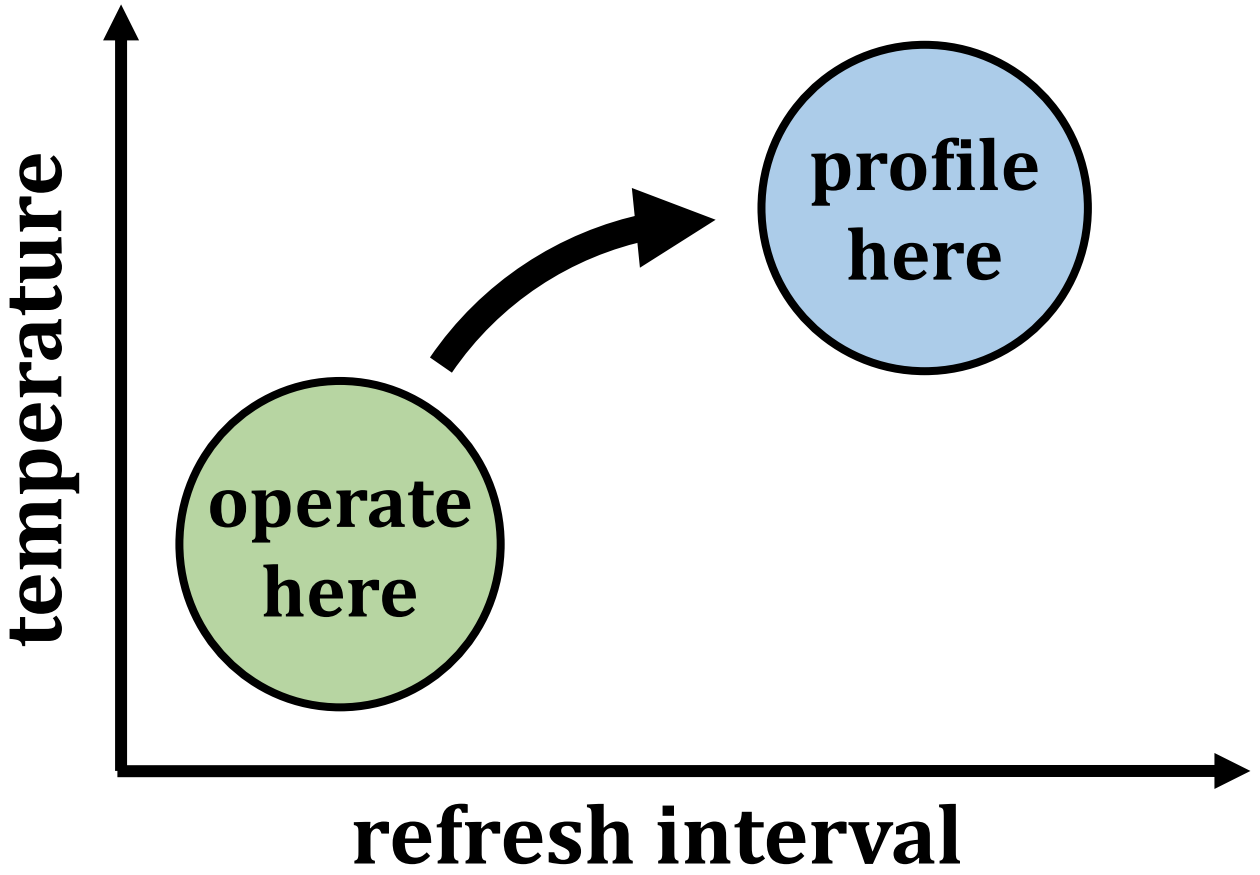
# Reach Profiling

**Key idea:** profile at a *longer refresh interval* and/or a *higher temperature*



# Reach Profiling

**Key idea:** profile at a *longer refresh interval* and/or a *higher temperature*



# Reach Profiling

**Key idea:** profile at a *longer refresh interval* and/or a *higher temperature*

- **Pros**
  - **Fast + Reliable:** reach profiling searches for cells *where they are most likely to fail*
- **Cons**
  - **False Positives:** profiler may identify cells that fail under profiling conditions, but not under operating conditions

# Towards an Implementation

Reach profiling is a **general methodology**

3 key questions for an implementation:

**What are desirable profiling conditions?**

**How often should the system profile?**

**What information does the profiler need?**

# Three Key Profiling Metrics

1. **Runtime:** how long profiling takes

2. **Coverage:** portion of all possible failures discovered by profiling

3. **False positives:** number of cells observed to fail during profiling but never during actual operation

# Three Key Profiling Metrics

1. **Runtime:** how long profiling takes

2. **Coverage:** portion of all possible failures discovered by profiling

We explore how these three metrics change under **many** different profiling conditions

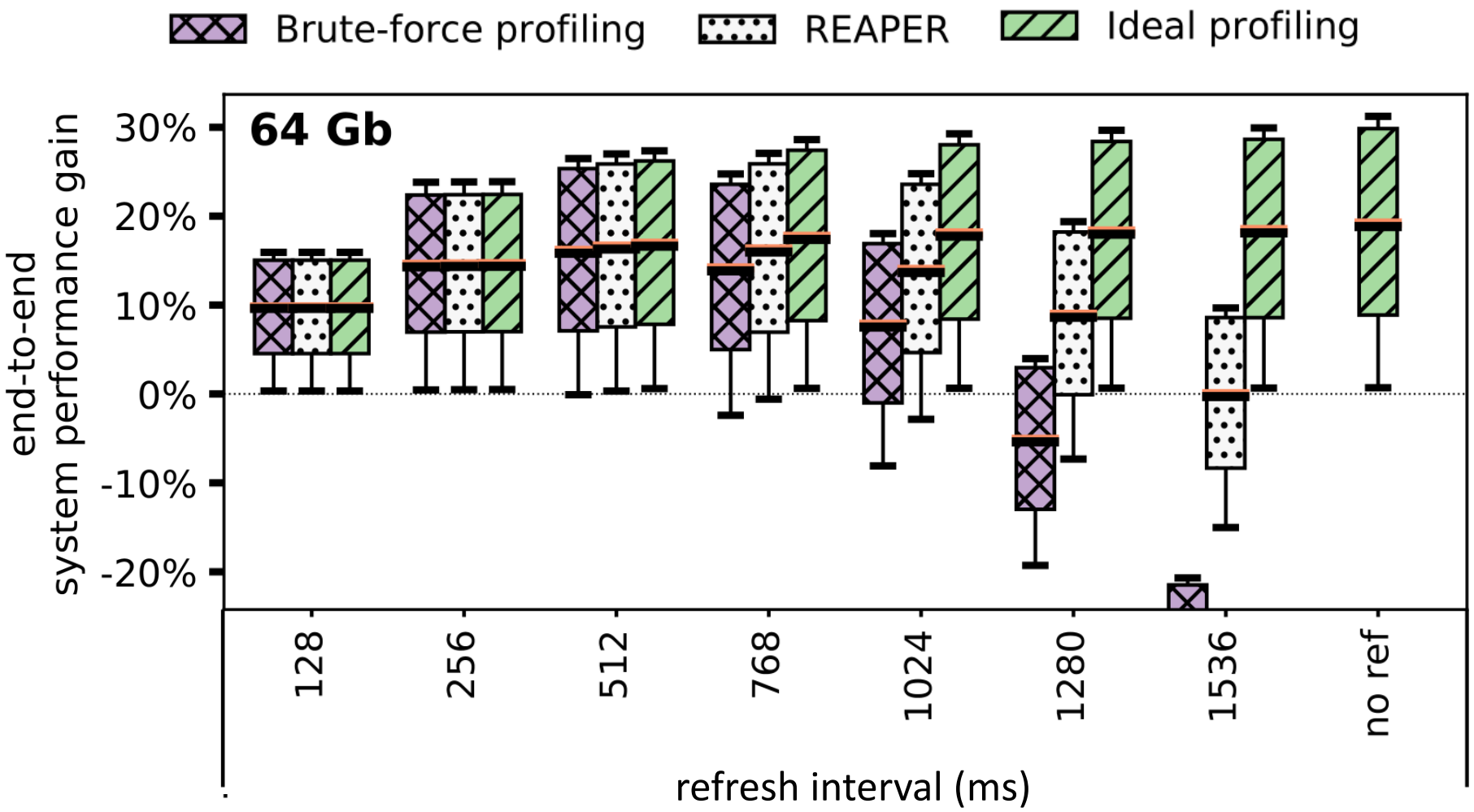
# Evaluation Methodology

- Simulators
  - **Performance**: Ramulator [Kim+, CAL'15]
  - **Energy**: DRAMPower [Chandrasekar+, DSD'11]

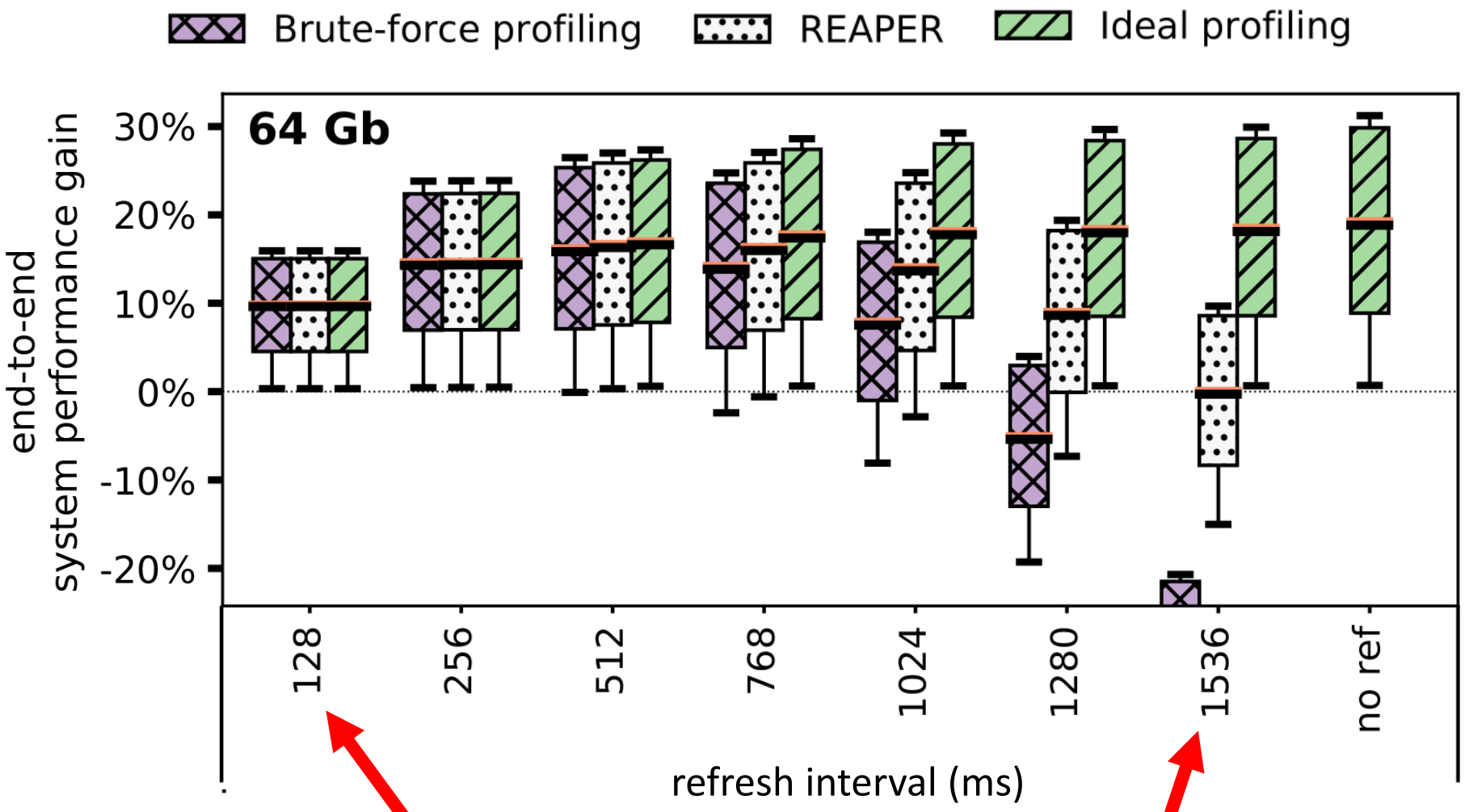
- Configuration
  - 4-core (4GHz), 8MB LLC
  - LPDDR4-3200, 4 channels, 1 rank/channel

- Workloads
  - 20 random 4-core benchmark mixes
  - SPEC CPU2006 benchmark suite

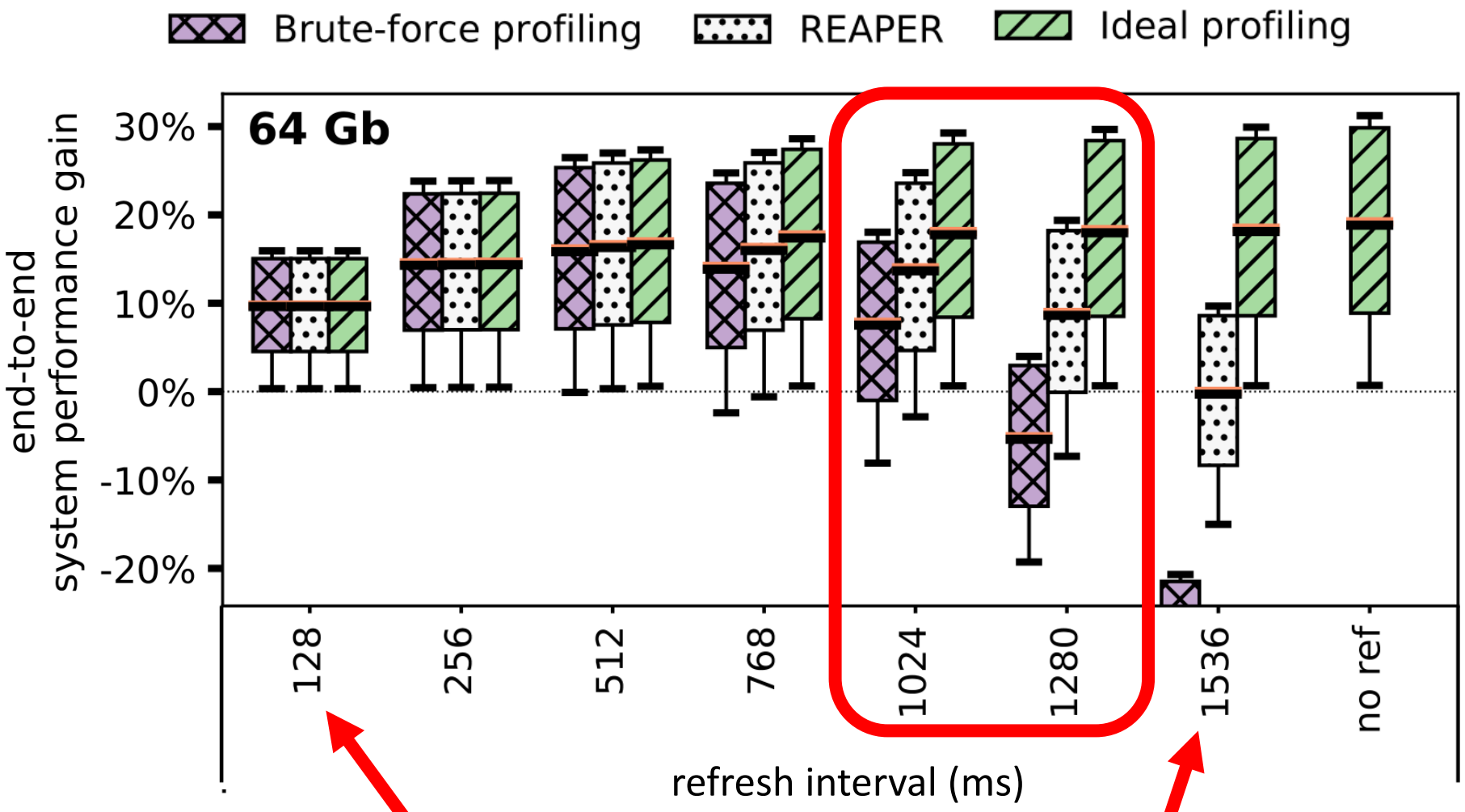
# Simulated End-to-end Performance



# Simulated End-to-end Performance



# Simulated End-to-end Performance

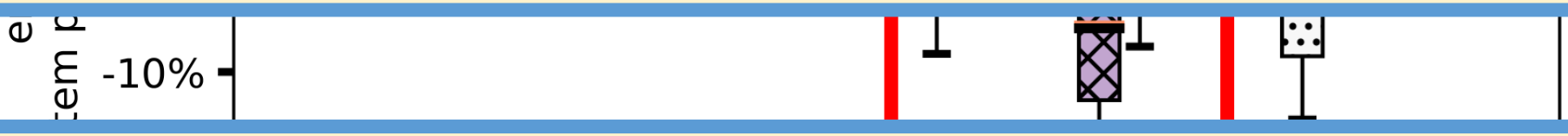


# Simulated End-to-end Performance


**On average, REAPER enables:**

**16.3% system performance improvement**

**36.4% DRAM power reduction**




**REAPER enables longer refresh intervals, which are unreasonable using brute-force profiling**

Reprofile rarely

Reprofile often

# Other Analyses in the Paper

- **Detailed LPDDR4 characterization data**
  - Temperature dependence effects
  - Retention time distributions
  - Data pattern dependence
  - Variable retention time
  - Individual cell failure distributions
- **Profiling tradeoff space characterization**
  - Runtime, coverage, and false positive rate
  - Temperature and refresh interval
- **Probabilistic model for tolerable failure rates**
- **Detailed results for end-to-end evaluations**

# REAPER Summary

**Problem**:

- DRAM refresh performance and energy overhead is high
- Current approaches to retention failure profiling are slow or unreliable

**Goals**:

1. Thoroughly analyze profiling tradeoffs
2. Develop a **fast** and **reliable** profiling mechanism

**Key Contributions**:

1. **First** detailed characterization of 368 LPDDR4 DRAM chips
2. **Reach profiling:** Profile at a **longer refresh interval** or **higher temperature** than target conditions, where cells are more likely to fail

**Evaluation:**

- **2.5x** faster profiling with **99%** coverage and **50%** false positives
- REAPER enables **16.3% system performance improvement** and **36.4% DRAM power reduction**
- Enables longer refresh intervals that were previously unreasonable

# Handling Both DPD and VRT [ISCA'17]

- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  **"The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"**
  *Proceedings of the 44th International Symposium on Computer Architecture (ISCA)*, Toronto, Canada, June 2017.
  [Slides (pptx) (pdf)]
  [Lightning Session Slides (pptx) (pdf)]

- First experimental analysis of (mobile) LPDDR4 chips
- Analyzes the complex tradeoff space of retention time profiling
- Idea: enable fast and robust profiling at higher refresh intervals & temperatures

**The Reach Profiler (REAPER):**
**Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions**

Minesh Patel[§‡] Jeremie S. Kim[‡§] Onur Mutlu[§‡]
[§]ETH Zürich [‡]Carnegie Mellon University