AVATAR: A VARIABLE-RETENTION TIME AWARE REFRESH FOR DRAM

DSN-45
06/24/2015
Rio de Janeiro, Brazil

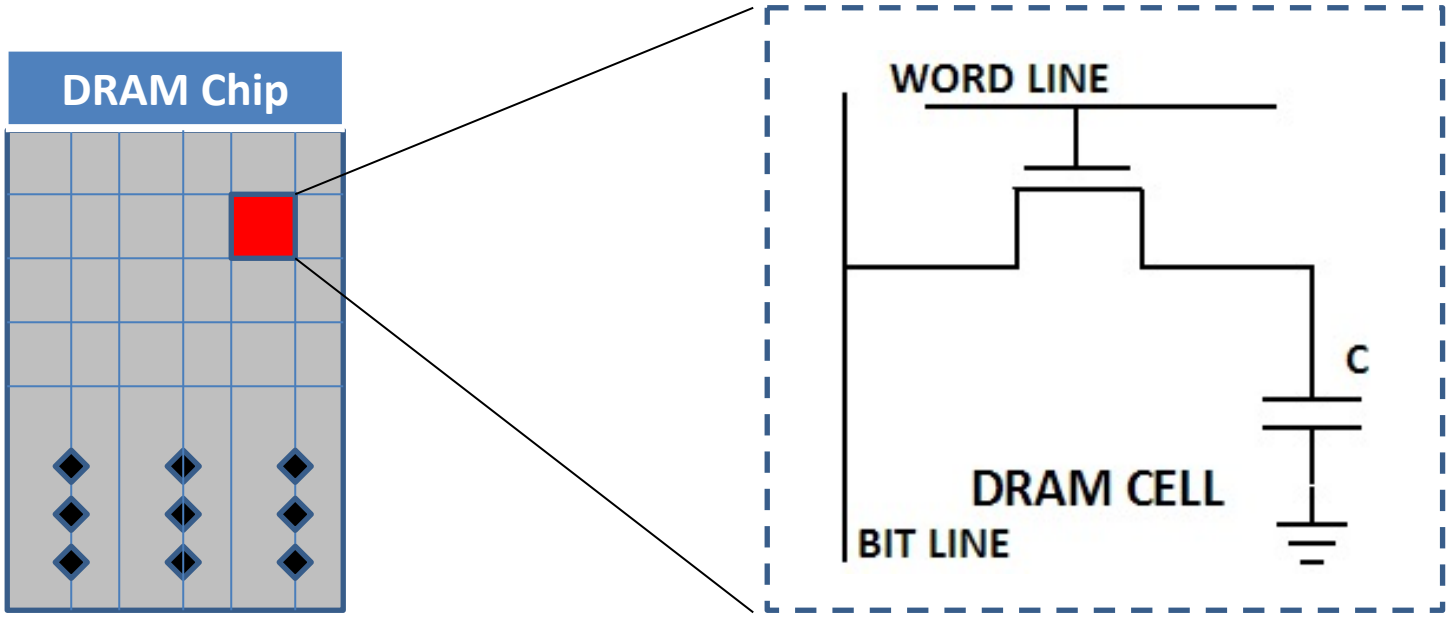
Moinuddin Qureshi, Georgia Tech

Dae-Hyun Kim
Prashant Nair

Samira Khan
Onur Mutlu

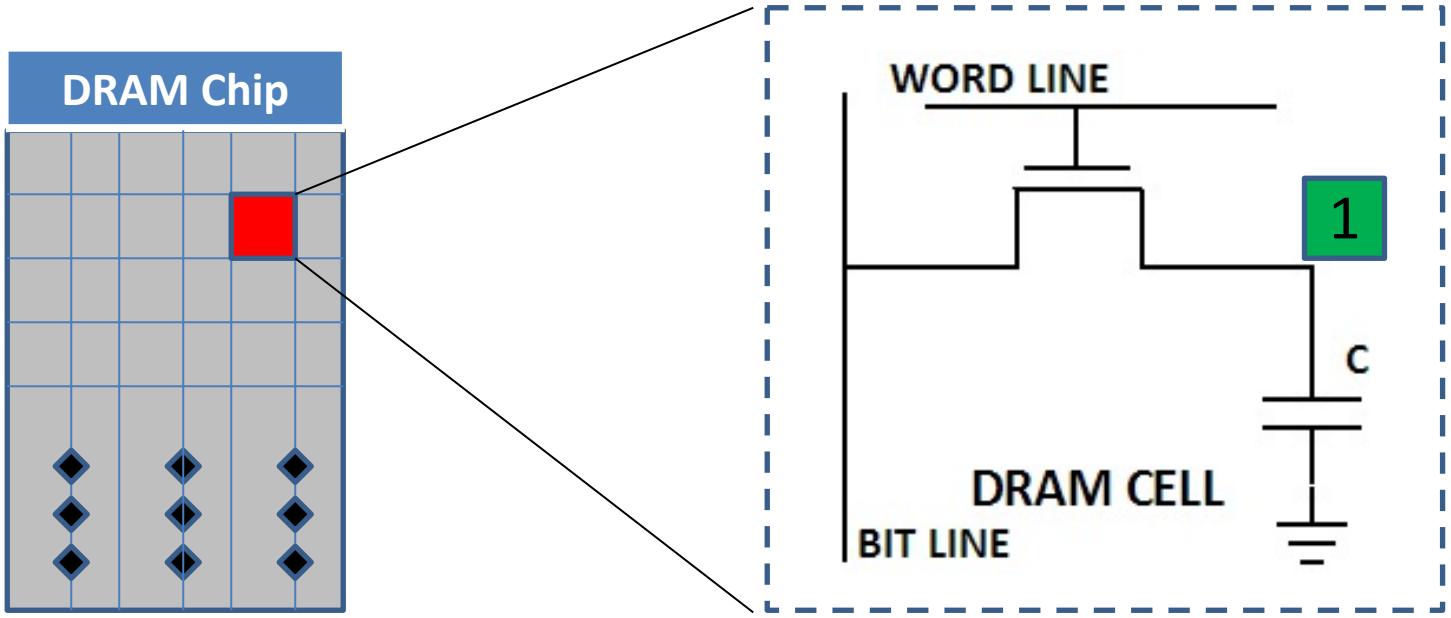
DRAM BACKGROUND

Dynamic Random Access Memory (DRAM) stores data as charge on capacitor



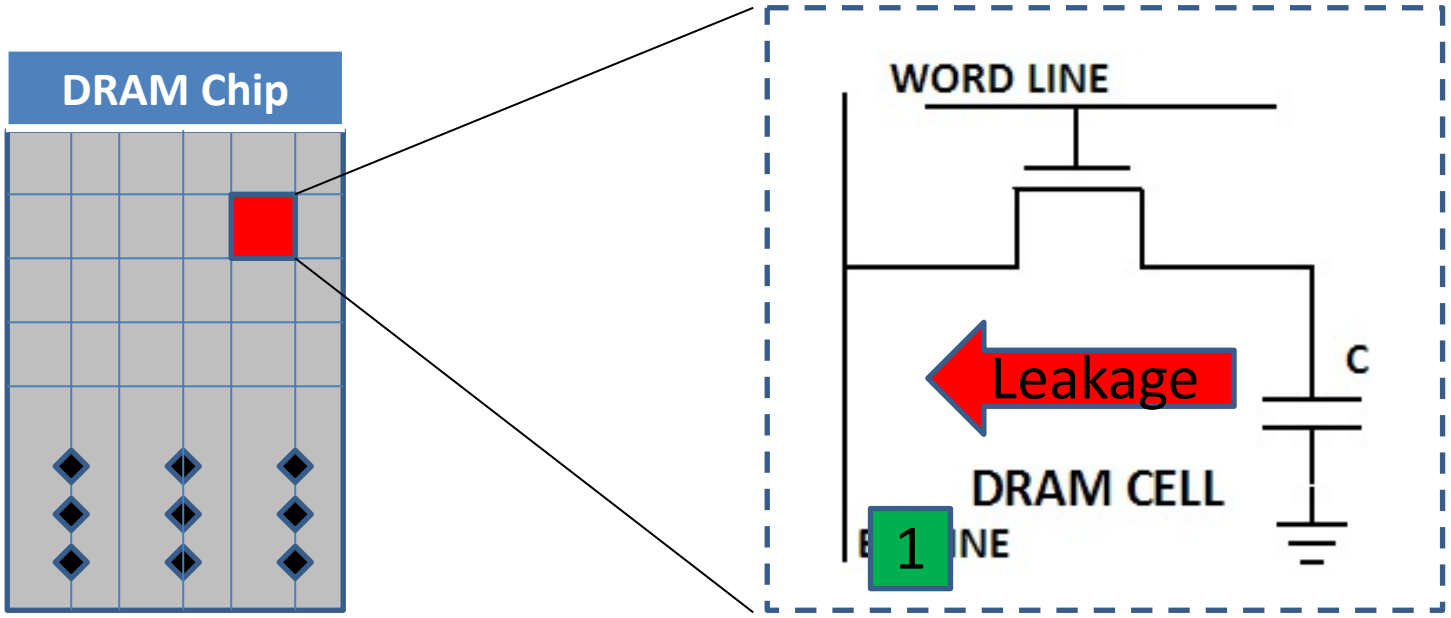
DRAM BACKGROUND

Dynamic Random Access Memory (DRAM) stores data as charge on capacitor



DRAM BACKGROUND

Dynamic Random Access Memory (DRAM) stores data as charge on capacitor

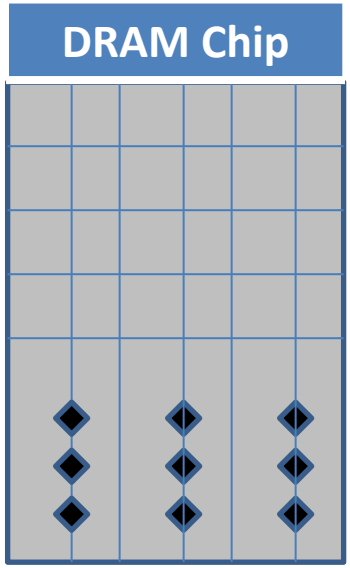


DRAM is a volatile memory → charge leaks quickly

DRAM REFRESH

Retention Time: The time for which cell/memory retains data

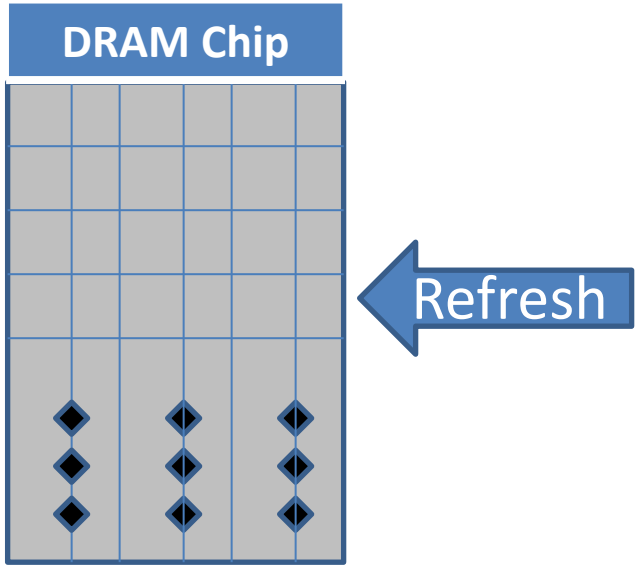
DRAM maintains data by “refresh” operations at row granularity



DRAM REFRESH

Retention Time: The time for which cell/memory retains data

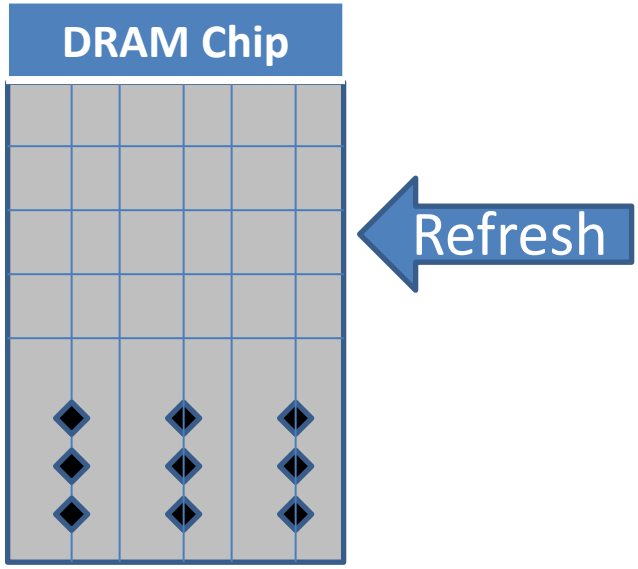
DRAM maintains data by "refresh" operations at row granularity



DRAM REFRESH

Retention Time: The time for which cell/memory retains data

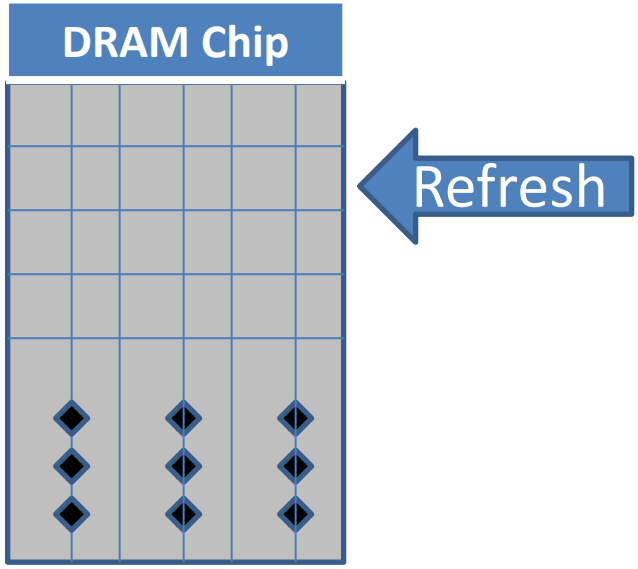
DRAM maintains data by "refresh" operations at row granularity



DRAM REFRESH

Retention Time: The time for which cell/memory retains data

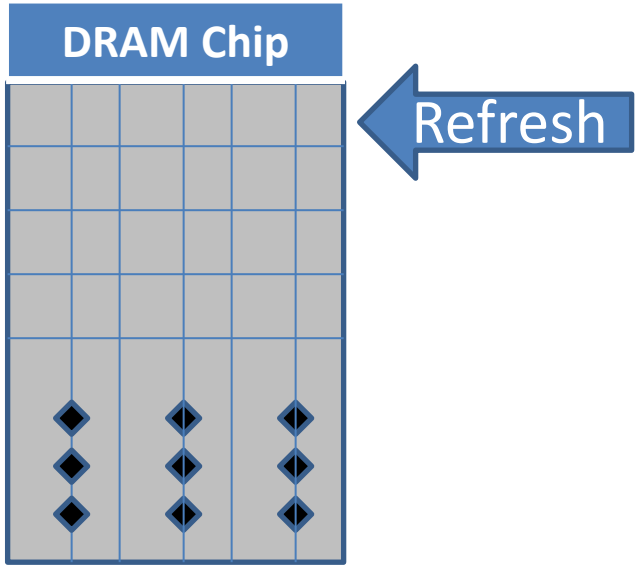
DRAM maintains data by "refresh" operations at row granularity



DRAM REFRESH

Retention Time: The time for which cell/memory retains data

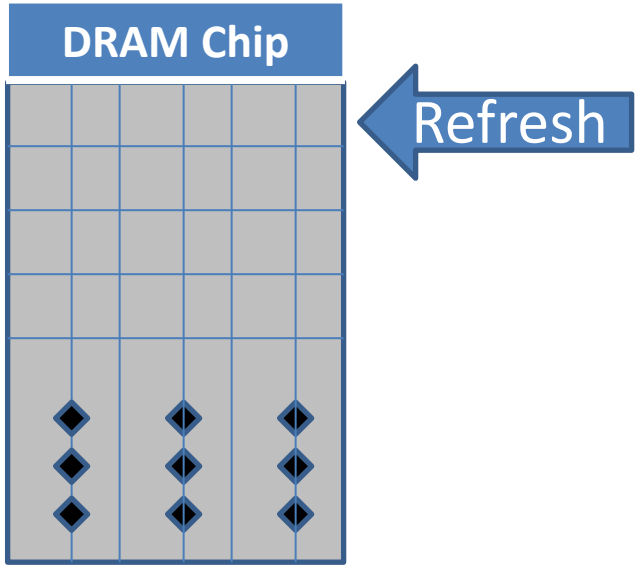
DRAM maintains data by “refresh” operations at row granularity



DRAM REFRESH

Retention Time: The time for which cell/memory retains data

DRAM maintains data by "refresh" operations at row granularity

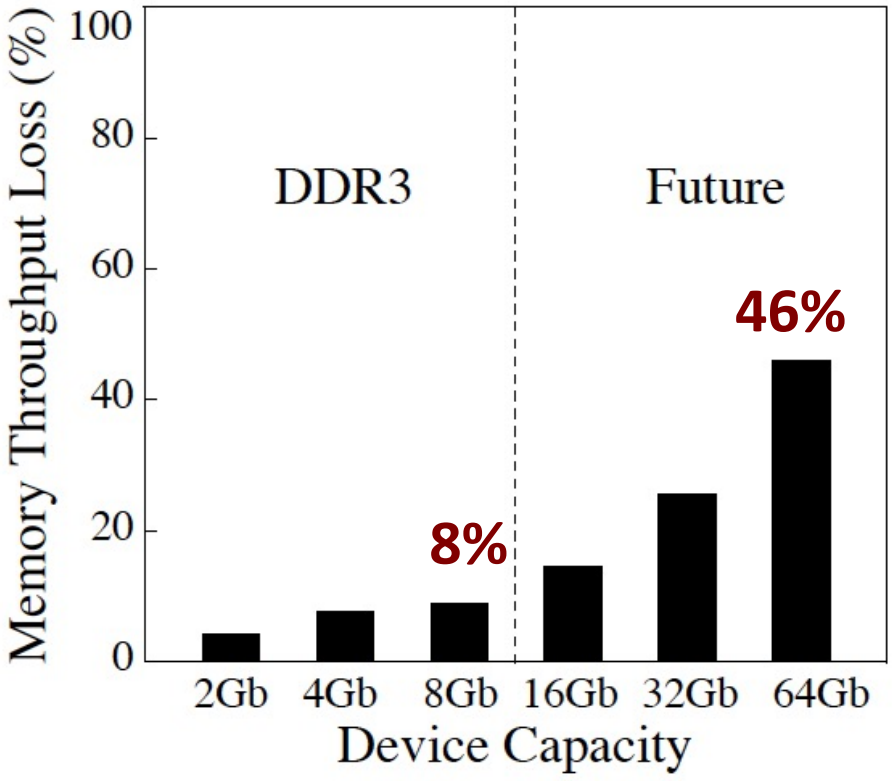


Refresh period determined by "worst-case" cell: 64ms (JEDEC)

DRAM relies on refresh (64ms) for data integrity

"REFRESH WALL" FOR DRAM SYSTEMS

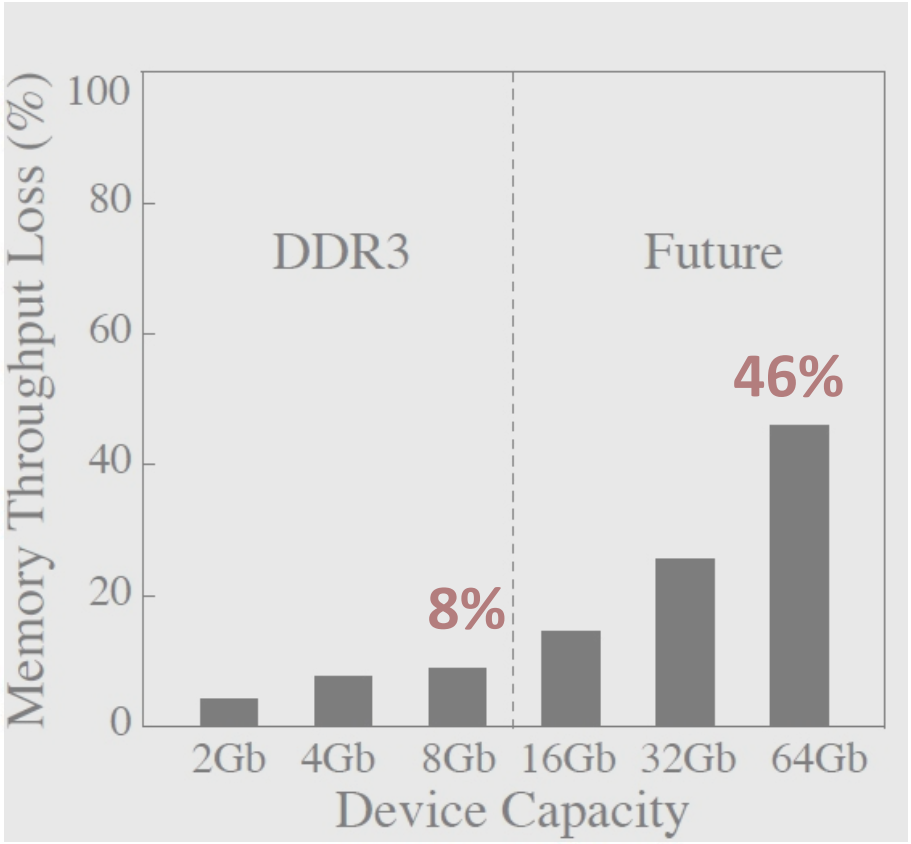
Refresh cost proportional to capacity ➔ Exponentially increasing

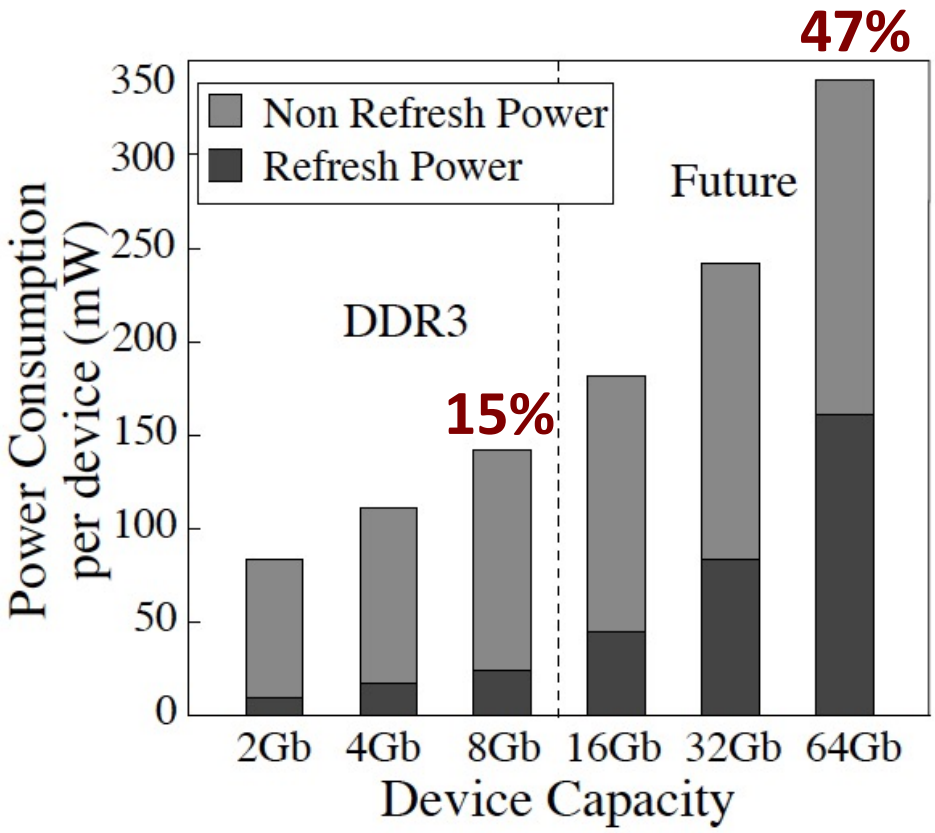


*Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

"REFRESH WALL" FOR DRAM SYSTEMS

Refresh cost proportional to capacity ➔ Exponentially increasing

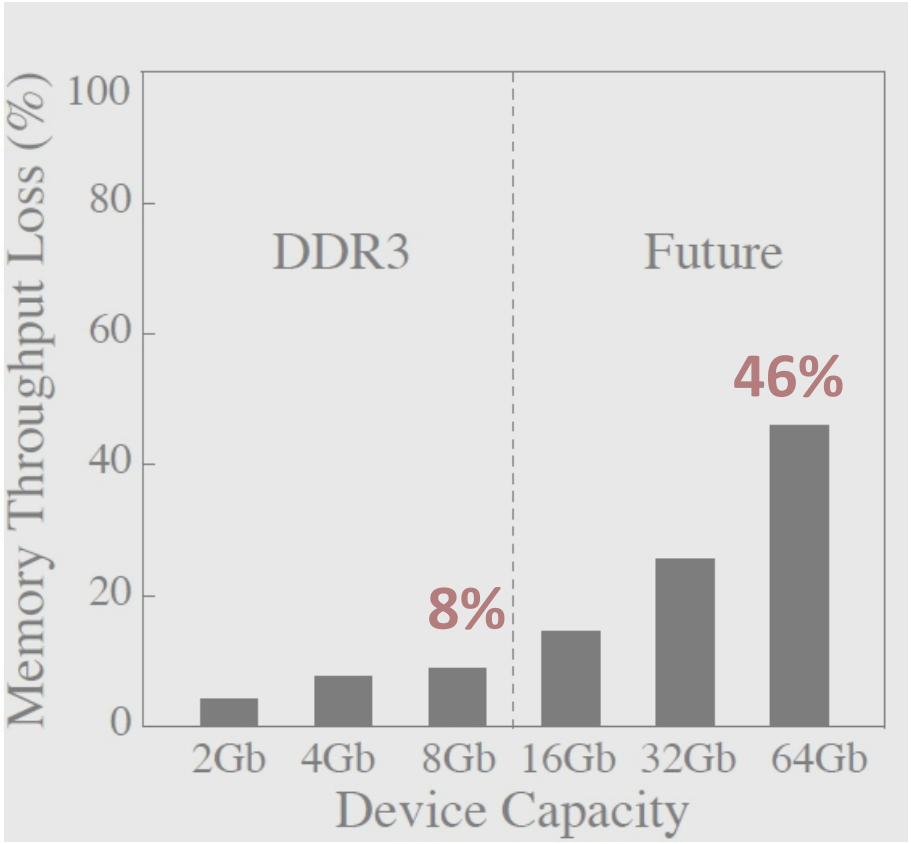


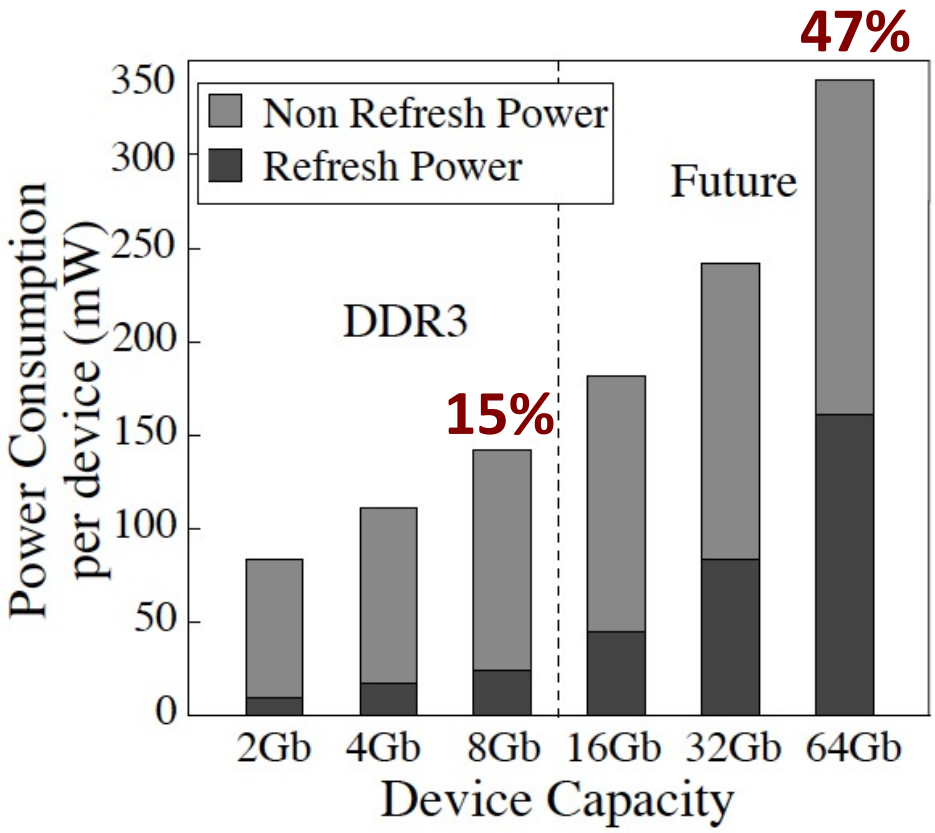


*Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

"REFRESH WALL" FOR DRAM SYSTEMS

Refresh cost proportional to capacity ➔ Exponentially increasing





Refresh consumes significant time and energy

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

NOT ALL RETENTION TIME IS CREATED EQUAL

Retention time of cells vary significantly: most cells >> 64ms

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

NOT ALL RETENTION TIME IS CREATED EQUAL

Retention time of cells vary significantly: most cells >> 64ms

Exploit variability in retention time ➔ Multirate Refresh

Normal Refresh (64ms) & Slow Refresh (e.g. 256ms+)

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

NOT ALL RETENTION TIME IS CREATED EQUAL

Retention time of cells vary significantly: most cells >> 64ms

Exploit variability in retention time ➔ Multirate Refresh

Normal Refresh (64ms) & Slow Refresh (e.g. 256ms+)

Row contains a cell with retention time < period of Slow Refresh

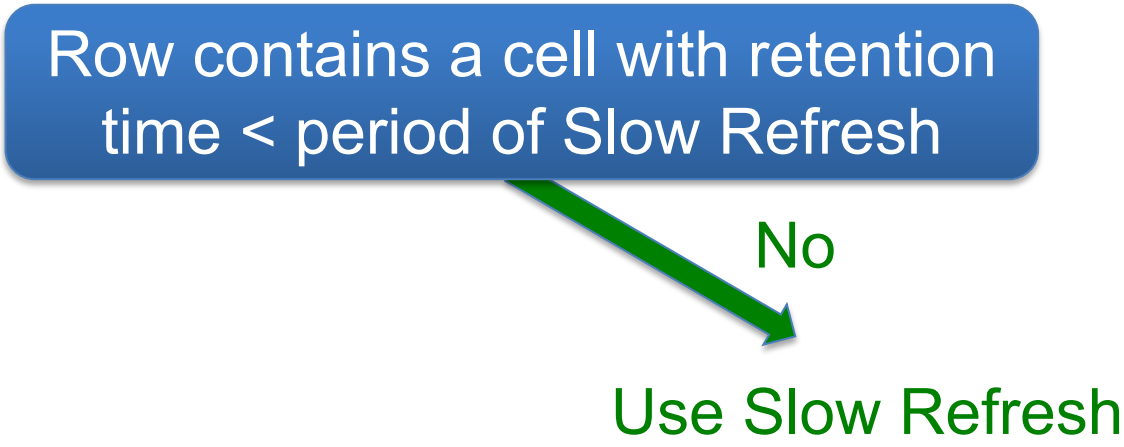
*Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

NOT ALL RETENTION TIME IS CREATED EQUAL

Retention time of cells vary significantly: most cells >> 64ms

Exploit variability in retention time ➔ Multirate Refresh

Normal Refresh (64ms) & Slow Refresh (e.g. 256ms+)



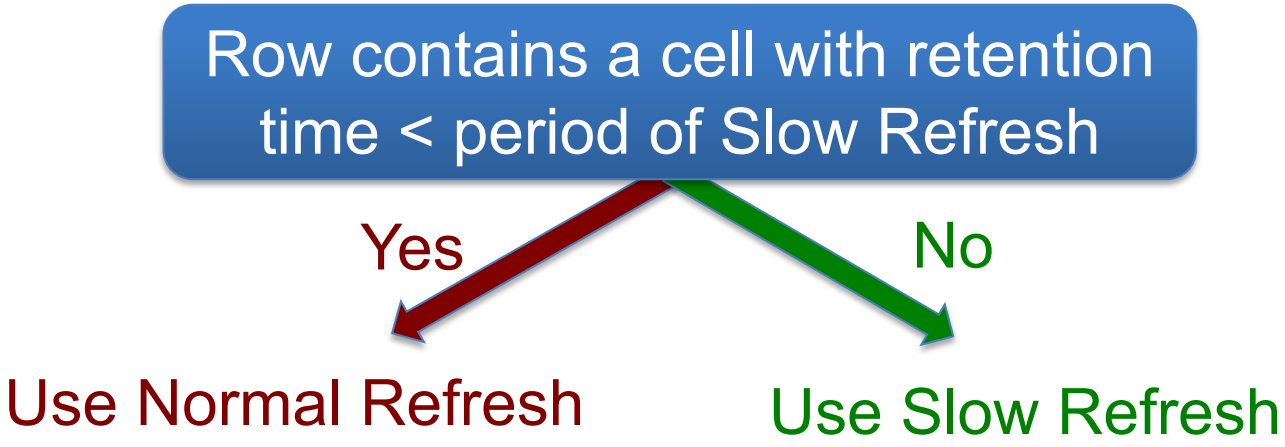
*Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

NOT ALL RETENTION TIME IS CREATED EQUAL

Retention time of cells vary significantly: most cells >> 64ms

Exploit variability in retention time ➔ Multirate Refresh

Normal Refresh (64ms) & Slow Refresh (e.g. 256ms+)



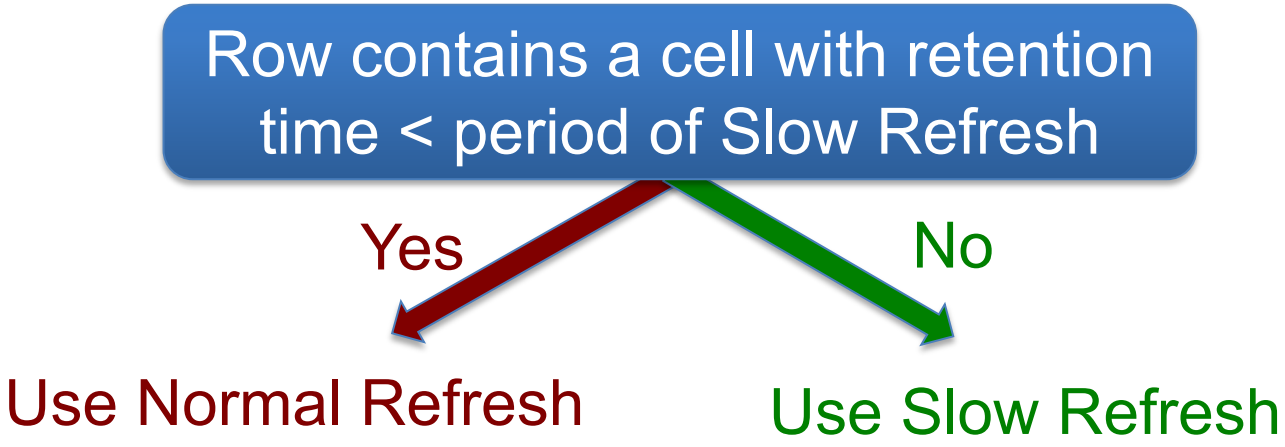
*Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

NOT ALL RETENTION TIME IS CREATED EQUAL

Retention time of cells vary significantly: most cells >> 64ms

Exploit variability in retention time ➔ Multirate Refresh

Normal Refresh (64ms) & Slow Refresh (e.g. 256ms+)

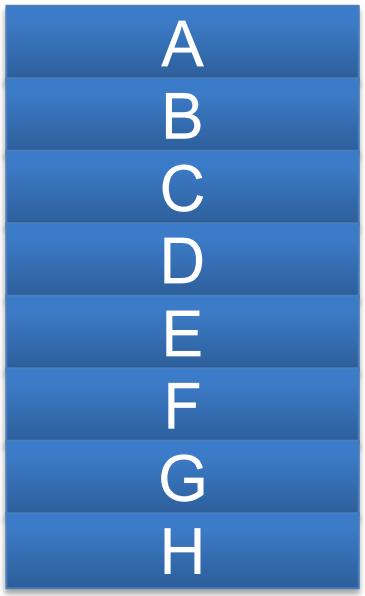


Efficient DRAM refresh by exploiting variability

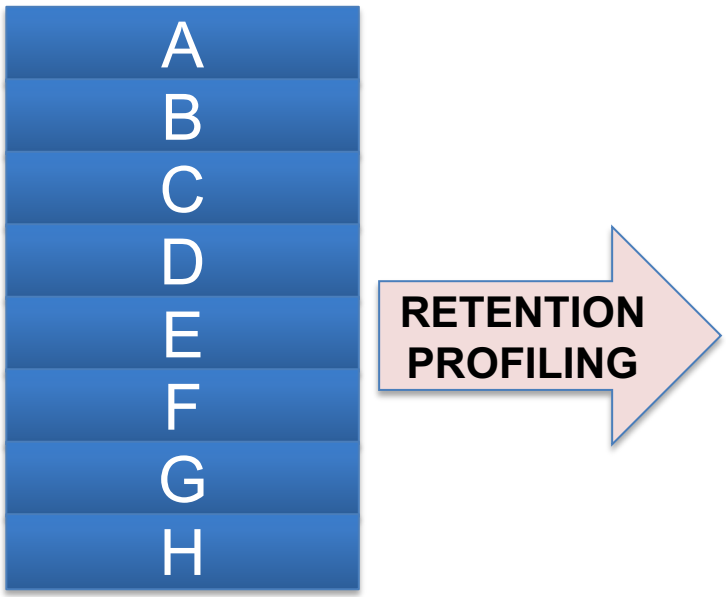
*Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

MULTI RATE REFRESH: DESIGN & EFFECTIVENESS

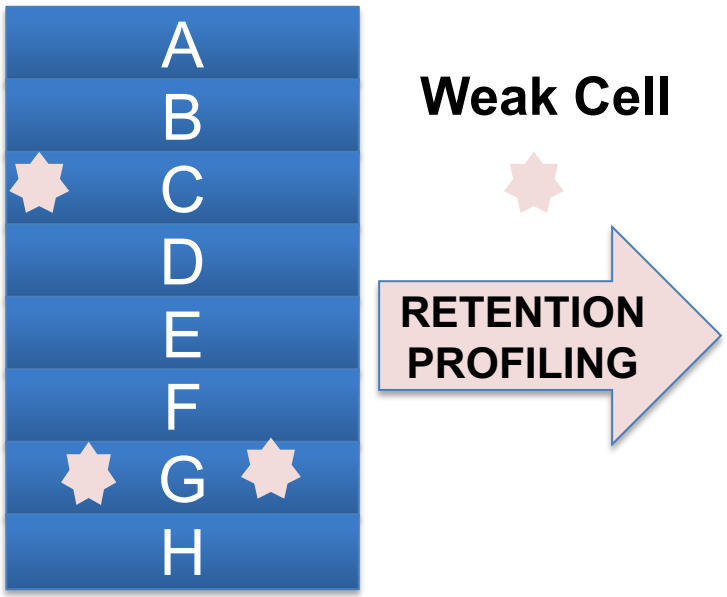
DRAM Rows



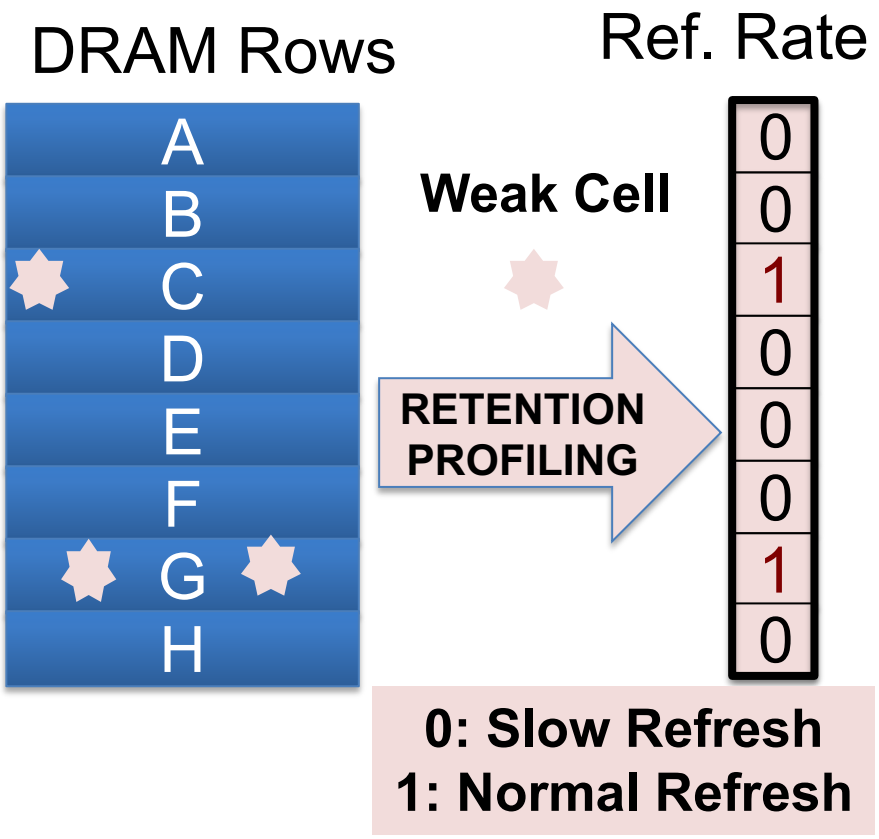
MULTI RATE REFRESH: DESIGN & EFFECTIVENESS



MULTI RATE REFRESH: DESIGN & EFFECTIVENESS

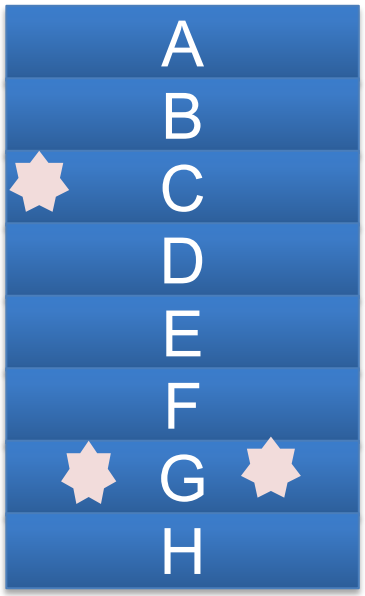


MULTI RATE REFRESH: DESIGN & EFFECTIVENESS



MULTI RATE REFRESH: DESIGN & EFFECTIVENESS

DRAM Rows



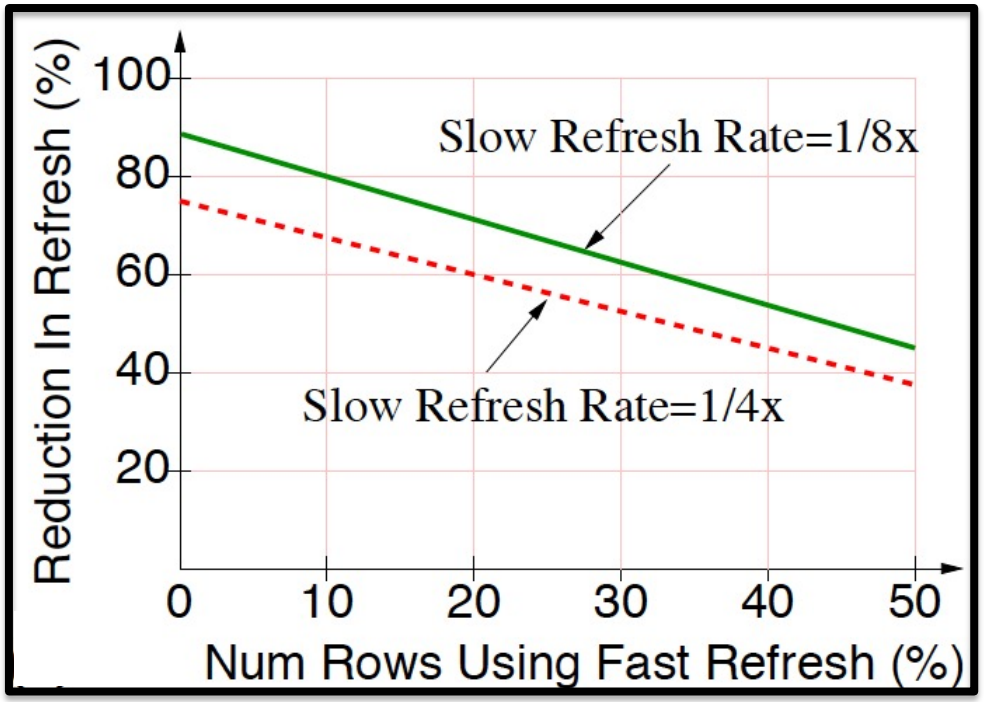
Weak Cell

RETENTION PROFILING

Ref. Rate Table

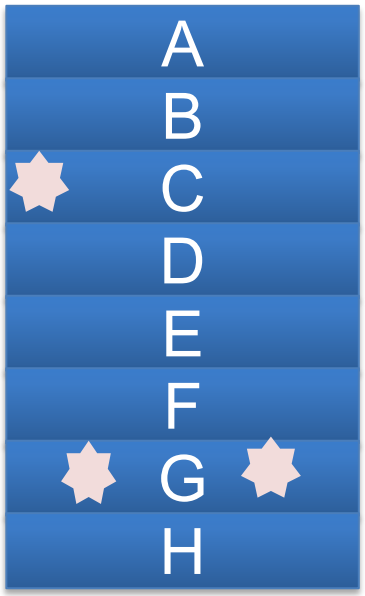
0
0
1
0
0
0
1
0

0: Slow Refresh
1: Normal Refresh



MULTI RATE REFRESH: DESIGN & EFFECTIVENESS

DRAM Rows



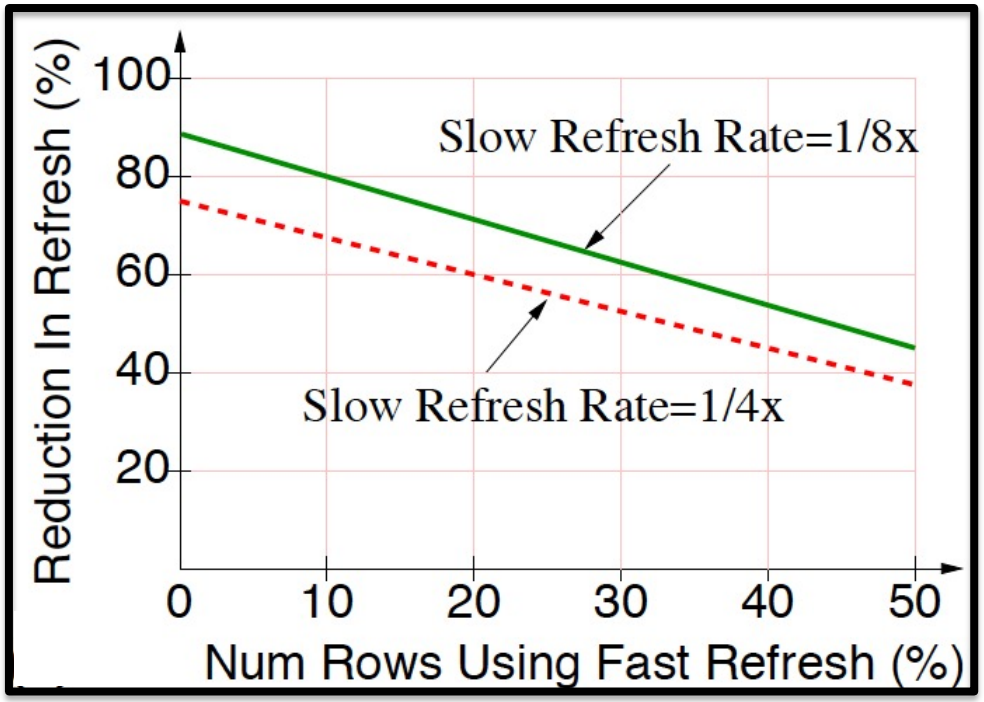
Weak Cell

RETENTION PROFILING

Ref. Rate Table

0
0
1
0
0
0
1
0

0: Slow Refresh
1: Normal Refresh

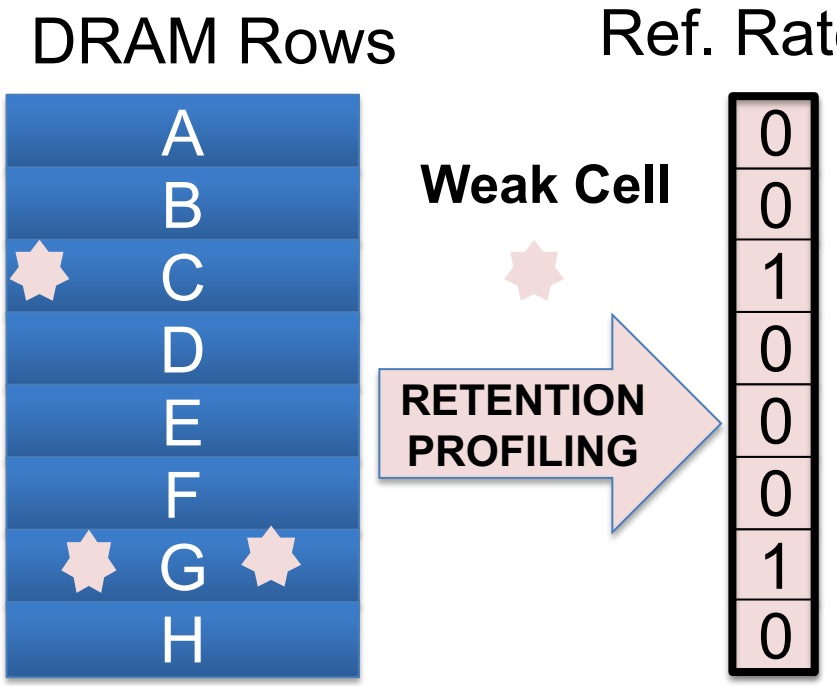


Multi rate refresh can reduce refresh by 70%+

VARIABLE RETENTION TIME (VRT): THE NEMESIS

Multirate refresh relies on retention time to remain unchanged

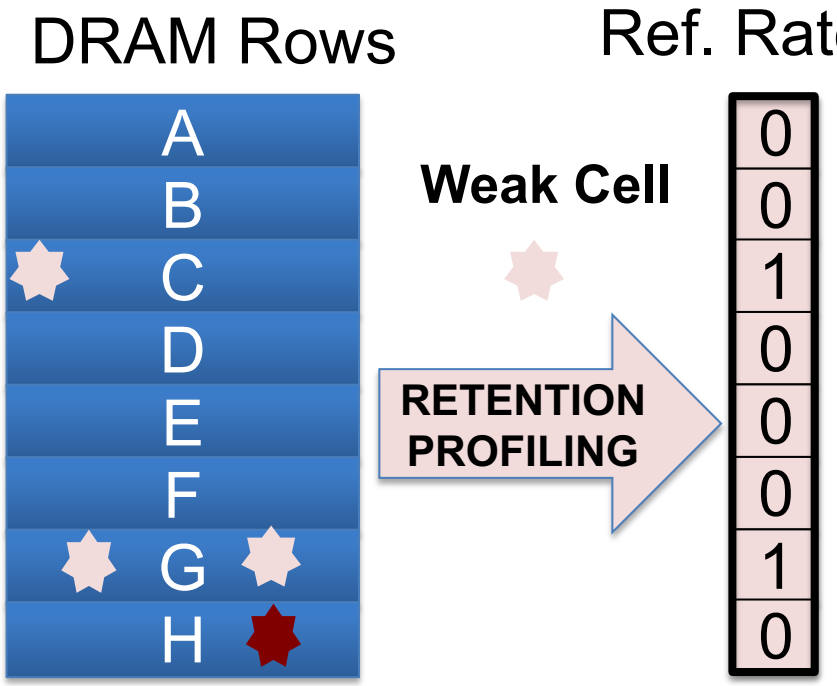
Retention time can vary at runtime due to VRT



VARIABLE RETENTION TIME (VRT): THE NEMESIS

Multirate refresh relies on retention time to remain unchanged

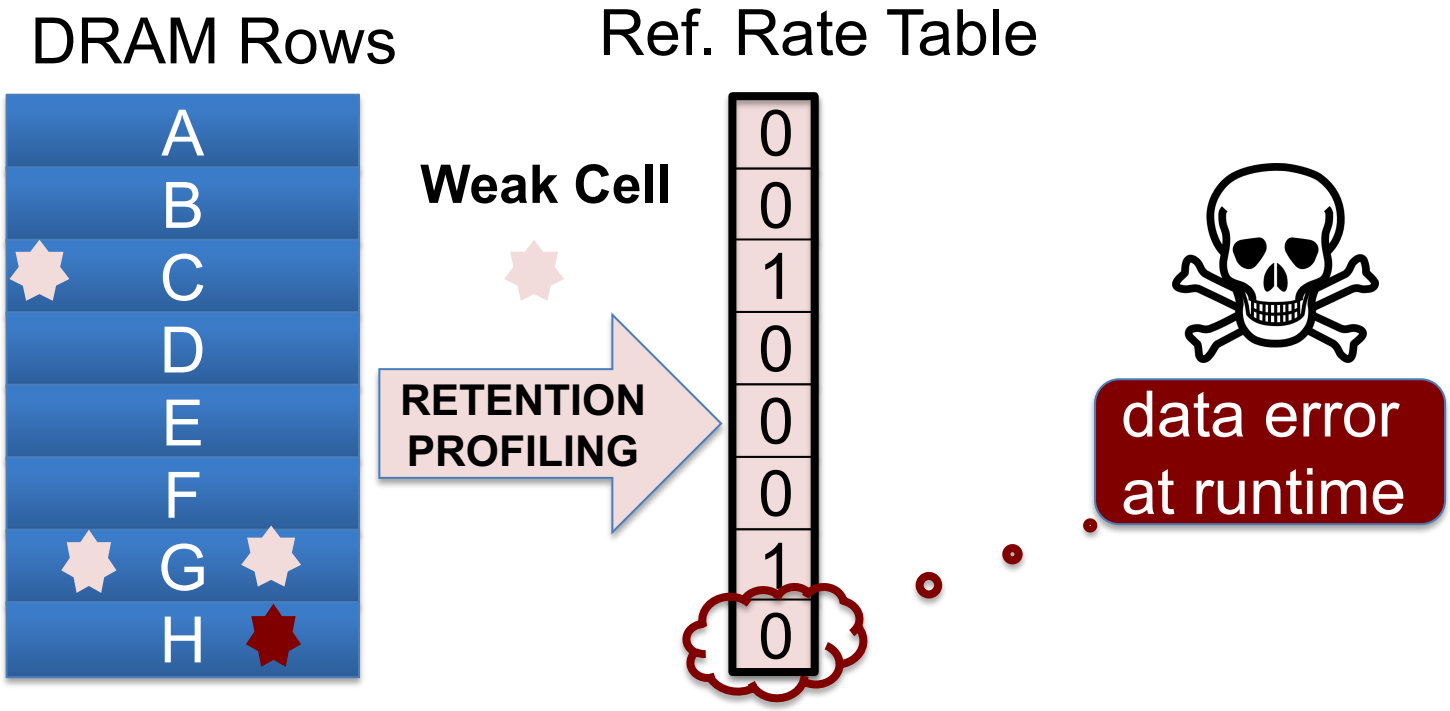
Retention time can vary at runtime due to VRT



VARIABLE RETENTION TIME (VRT): THE NEMESIS

Multirate refresh relies on retention time to remain unchanged

Retention time can vary at runtime due to VRT



VRT renders multi-rate refresh unusable in practice

GOALS

VRT considered one of the biggest impediment to DRAM scaling
-- [Samung & Intel, Memory Forum 2014]

Our study investigates the following questions:

1. Can we analyze VRT using architecture level models?
2. Can we overcome VRT simply by using ECC DIMM?
3. If not, what is a low cost solution to mitigate VRT?

OUTLINE

- Background
- VRT: mechanism, measurement, model
- Can't we fix VRT by simply using ECC DIMM?
- AVATAR
- Results
- Summary

WHY DOES VRT OCCUR? WHEN IS IT HARMFUL?

VRT caused by fluctuations in Gate Induced Drain Leakage.

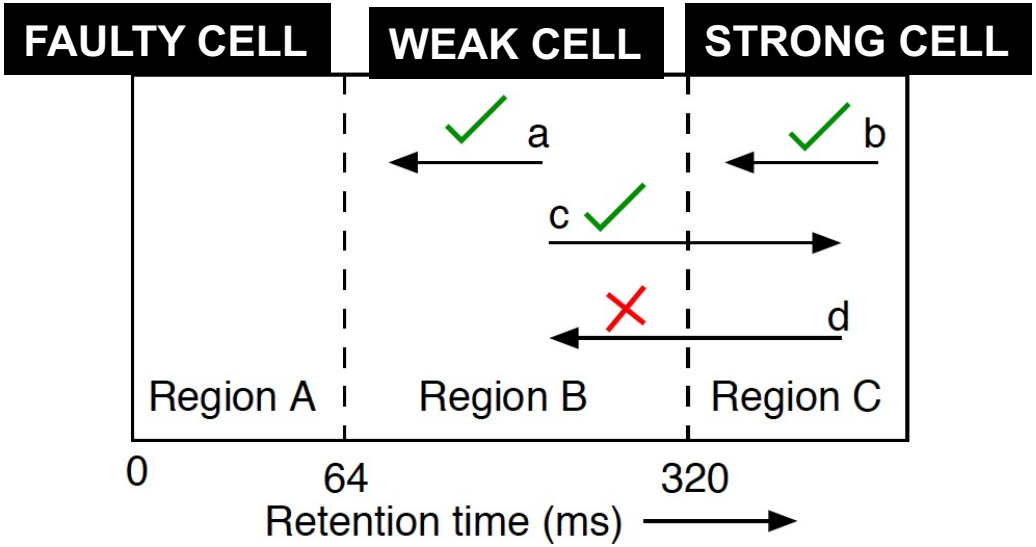
External factors: mechanical stress, high temperature etc.

WHY DOES VRT OCCUR? WHEN IS IT HARMFUL?

VRT caused by fluctuations in Gate Induced Drain Leakage.

External factors: mechanical stress, high temperature etc.

Not all VRT is harmful



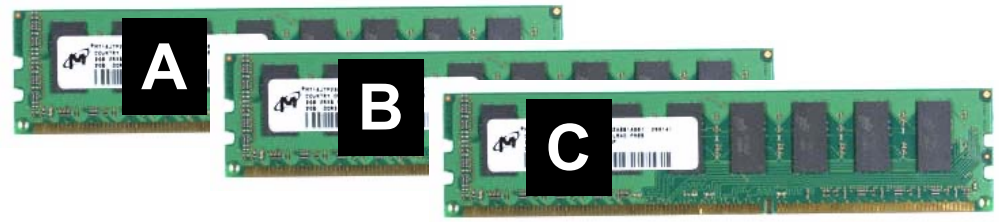
VRT problematic when strong cell becomes weak

EXPERIMENTAL SETUP

Test platform: DDR3 testing platform Xilinx ML605 FPGA development board in temperature controlled setting

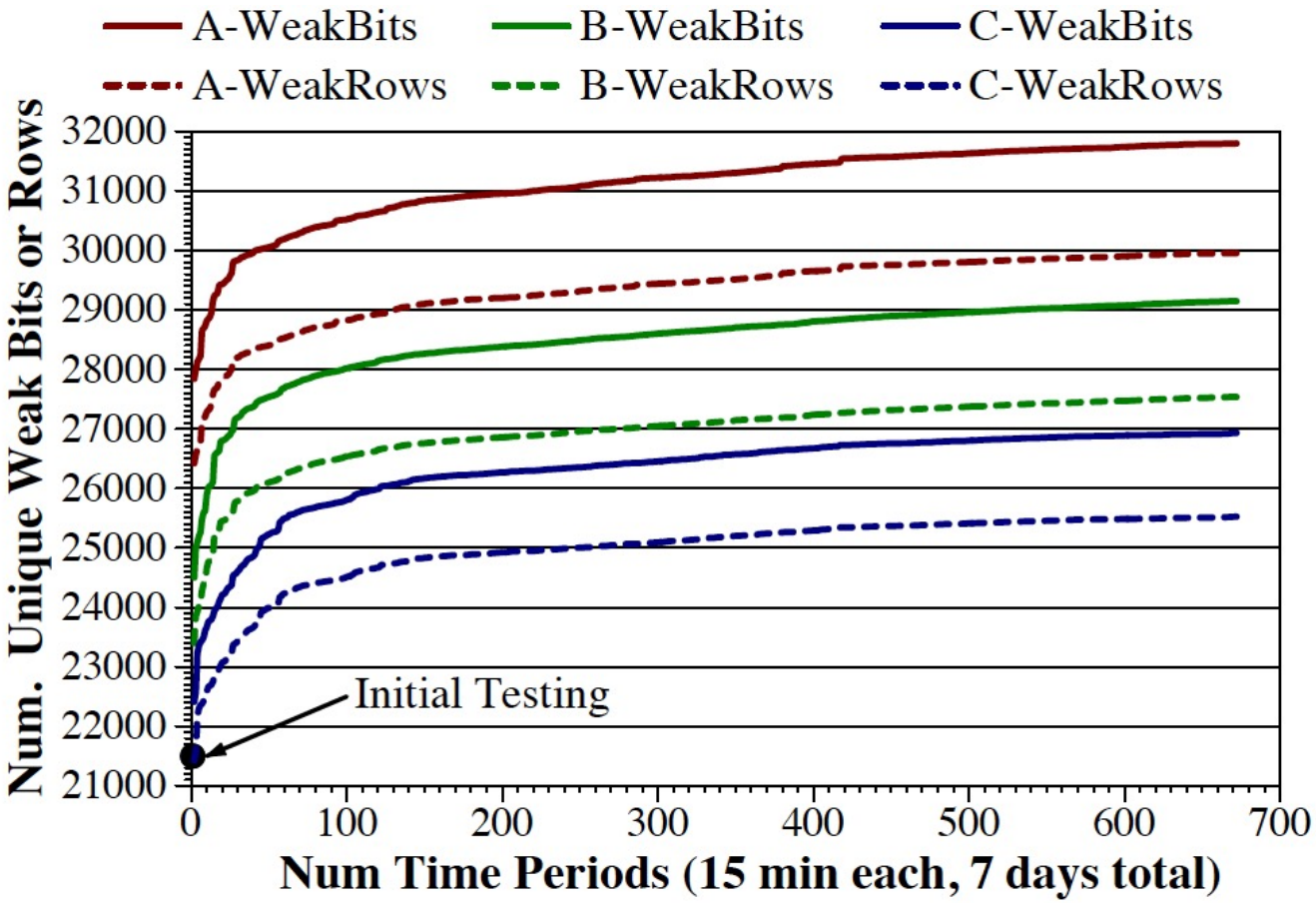
Slow Refresh: Studied refresh of 4s at 45C, corresponds to 328ms at 85C [khan+ SIGMETRICS'14, Liu+ ISCA'13]

Test: Write specific pattern, read pattern, log id of erroneous cell
Statistics collected every 15 minutes, over 7 days (672 rounds)



Three (2GB) modules, one each from different DRAM vendor

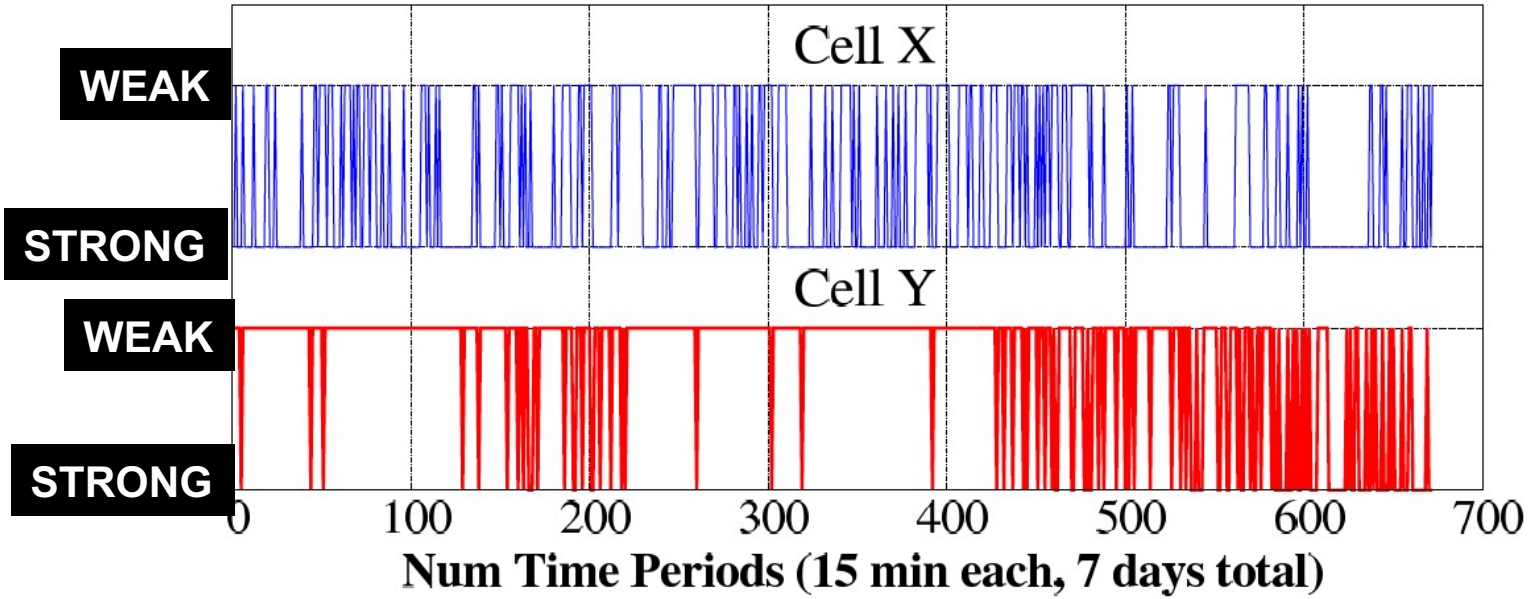
1: POPULATION OF WEAK CELLS INCREASES



Even after several days of testing, VRT causes new (previously unidentified) cells to cause failures

2: VRT-CELLS CAN SWITCH RANDOMLY

Cell with retention time < 328ms ➔ Weak Cell, else Strong Cell

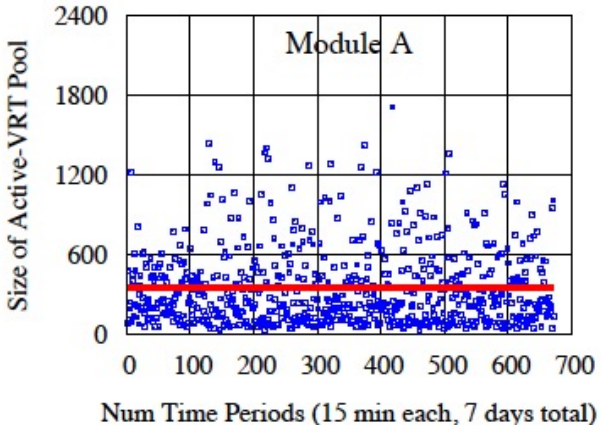


A VRT cell can randomly and frequently transition between strong and weak states

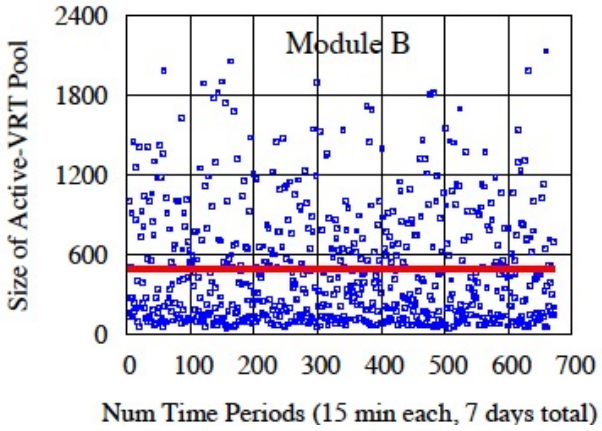
3: SIZE OF ACTIVE-VRT POOL VARIES

Active-VRT Cell: Cell that failed during the given 15-min round

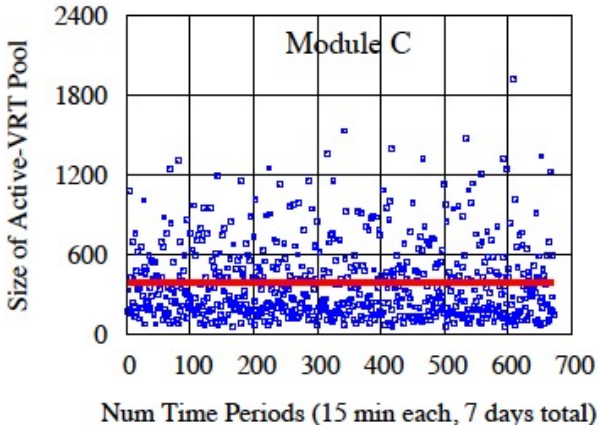
Active-VRT Pool (AVP): Group of Active VRT Cells



Avg=347



Avg=492



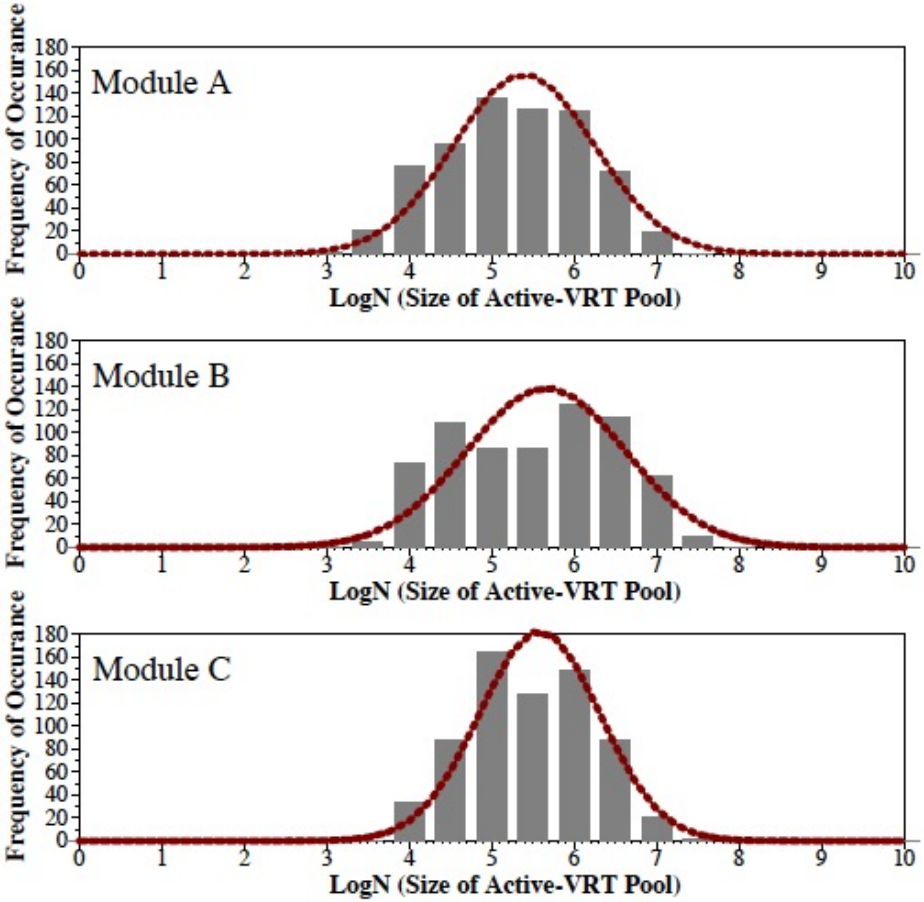
Avg=388

The size of AVP varies dynamically for all modules

MODELING THE DYNAMIC SIZE OF AVP

Predicting the exact AVP size is difficult, but it can be modeled

Observation:
AVP size tends to follow lognormal distribution

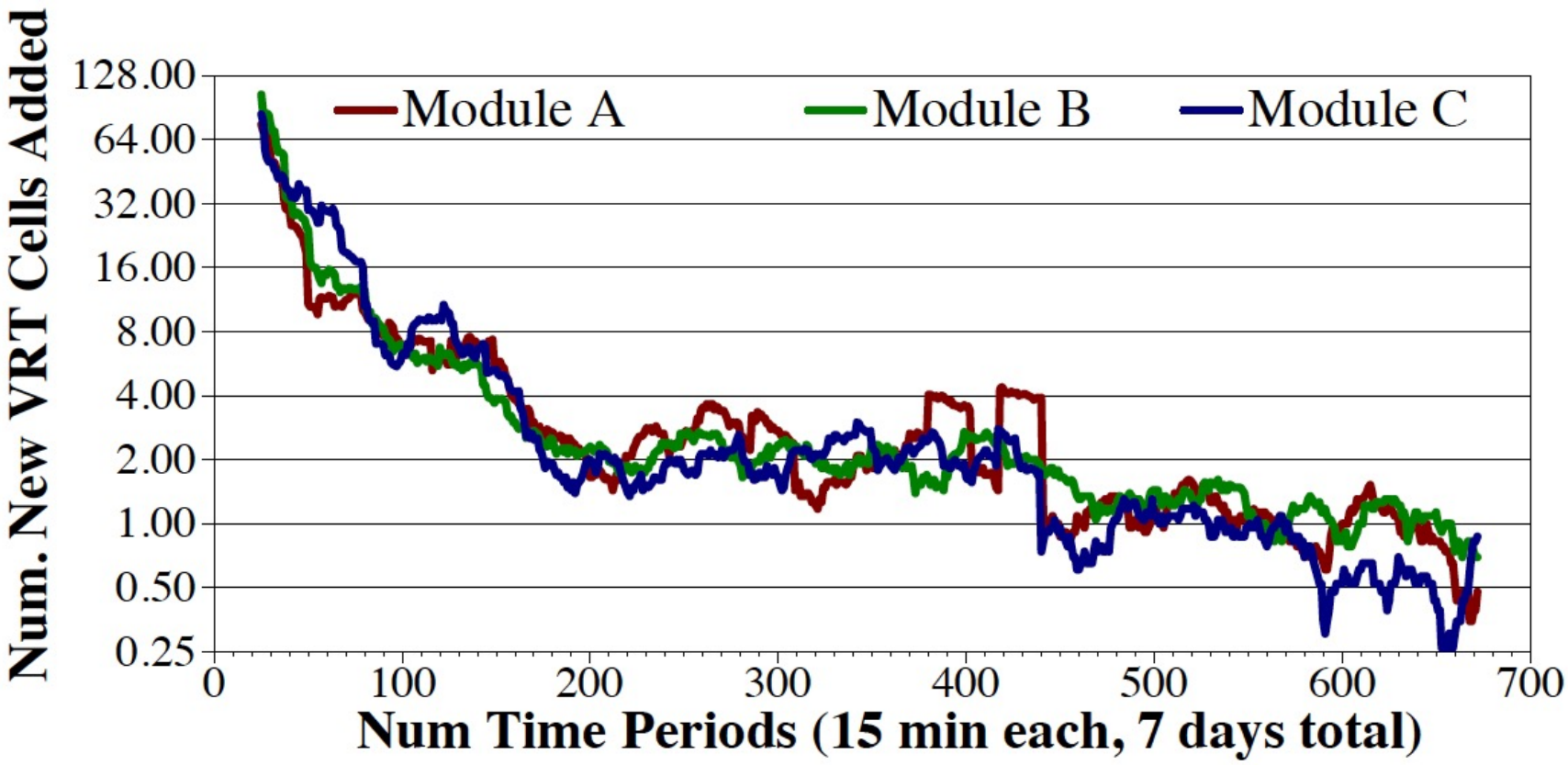


AVP size modeled using lognormal distribution

4: RATE OF NEW VRT CELLS STEADIES

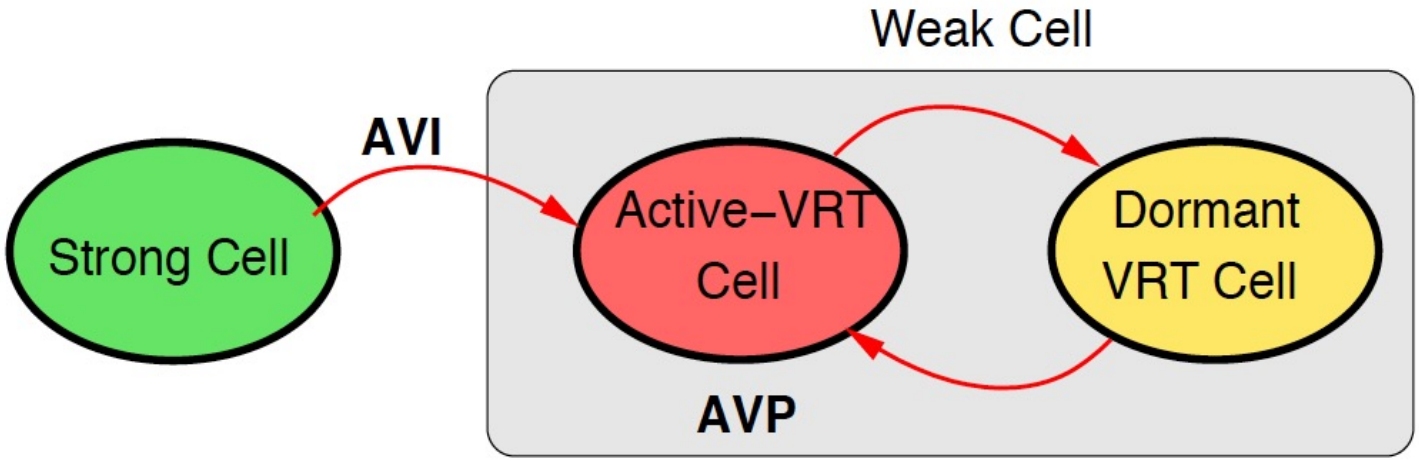
Active-VRT Injection (AVI) Rate

The rate at which new cells become Active-VRT cells

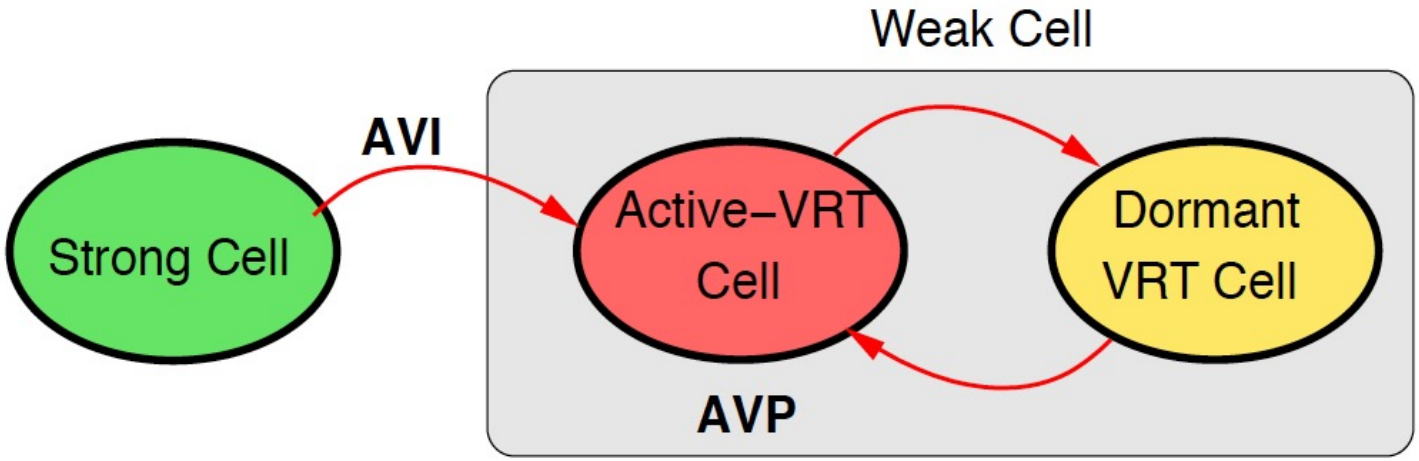


AVP reduces to ~1 new cell per 15-min period

ARCHITECTURE MODEL FOR CELL UNDER VRT



ARCHITECTURE MODEL FOR CELL UNDER VRT



Two key parameters:

Active-VRT Pool (AVP): How many VRT cells in this period?

Active-VRT Injection (AVI): How many new (previously undiscovered) cells became weak in this period?

Model has two parameters: AVP and AVI

ARCHITECTURE MODEL FOR VRT

```
Input: Mu,Sdev, for the logn of Active-VRT pool
Input:  K, rate of discovering new VRT cells
```

ARCHITECTURE MODEL FOR VRT

```
Input: Mu,Sdev, for the logn of Active-VRT pool
Input:  K, rate of discovering new VRT cells
```

```
PoolSize = Rand (LogNormDist[Mu,Sdev])
Insert K new elements in Pool
Remove K elements from Pool
P[TimePeriod] = System Failure Probability
```

ARCHITECTURE MODEL FOR VRT

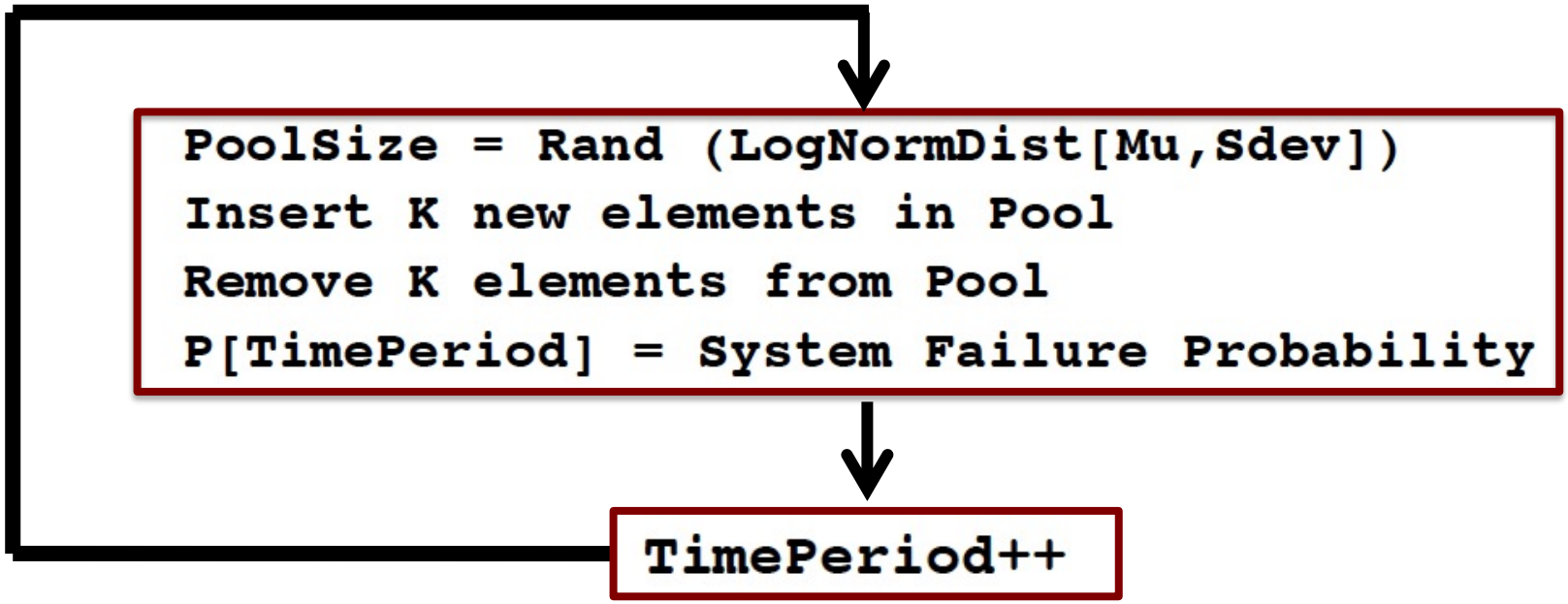
```
Input: Mu,Sdev, for the logn of Active-VRT pool
Input:  K, rate of discovering new VRT cells
```

```
PoolSize = Rand (LogNormDist[Mu,Sdev])
Insert K new elements in Pool
Remove K elements from Pool
P[TimePeriod] = System Failure Probability
```

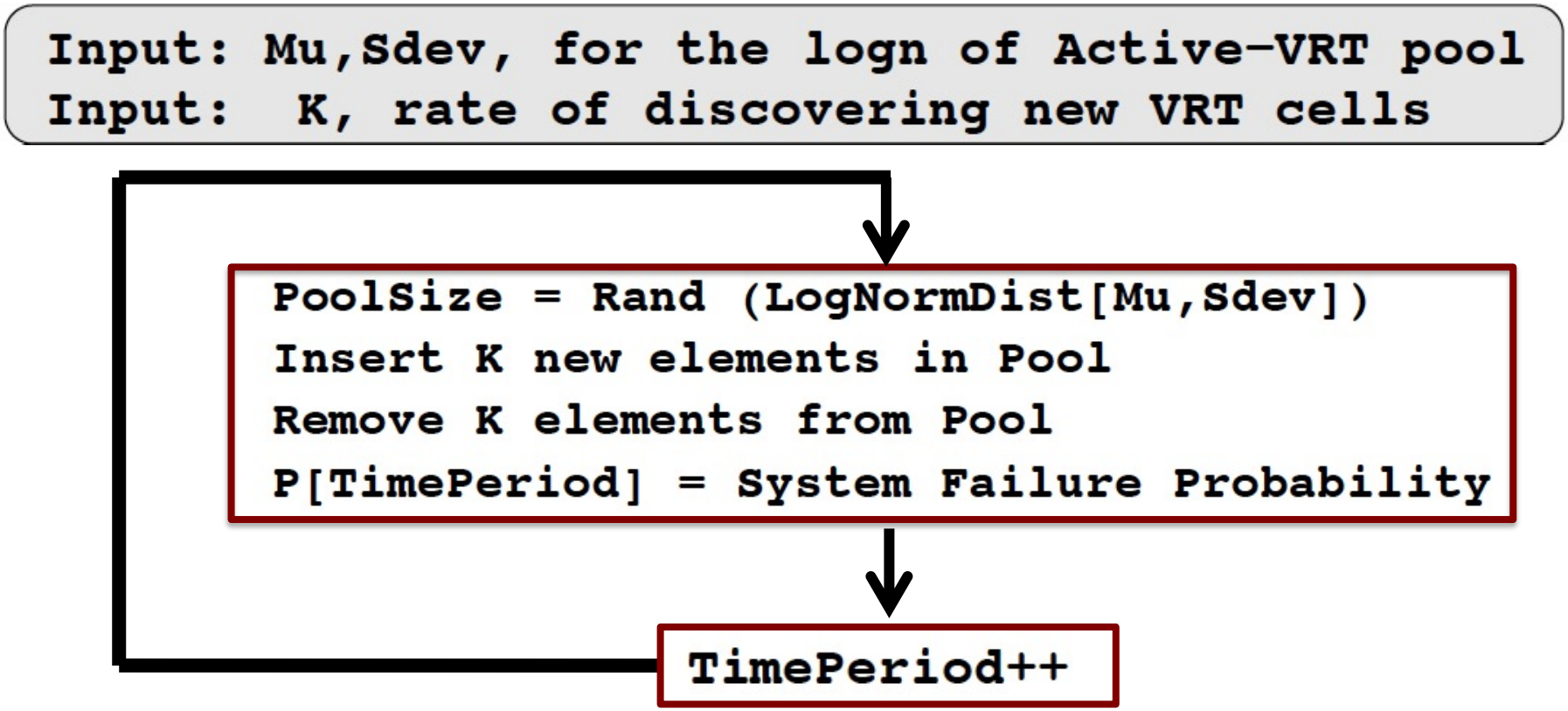
```
TimePeriod++
```

ARCHITECTURE MODEL FOR VRT

```
Input: Mu,Sdev, for the logn of Active-VRT pool
Input:  K, rate of discovering new VRT cells
```



ARCHITECTURE MODEL FOR VRT



Parameter scaling for larger systems: 2GB DIMM to 8GB DIMM
AVP size increased by 4x: from ~400 to ~1600
AVI rate increased by 4x: from 1 to 4

OUTLINE

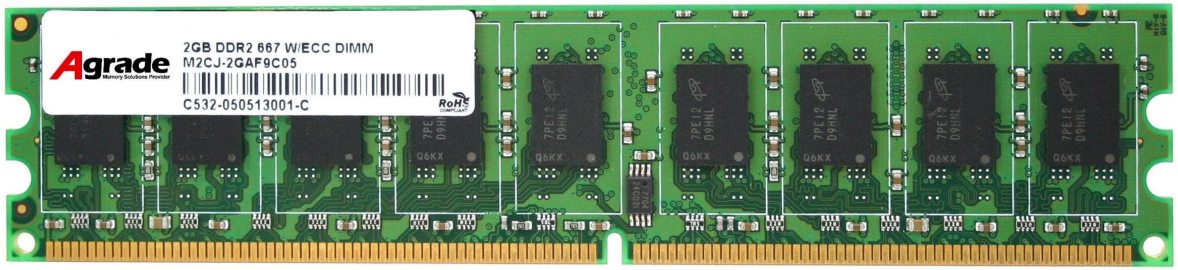
- Background
- VRT: mechanism, measurement, model

- Can't we fix VRT by simply using ECC DIMM?

- AVATAR
- Results
- Summary

BACKGROUND ON ECC DIMM

ECC DIMM can tolerate 1 error per word (8 bytes)



Typically used to tolerate soft error but can also be used to fix a bit error due to VRT

A multi-bit error per word ➔ uncorrectable error

What is time to double error per word under VRT?

ANALYTICAL MODEL FOR ECC DIMM

W words in memory (strong rows only)

P words have 1 bit error already (AVP)

K new weak cells get injected in given time quanta

$$P(DIMM\ has\ no\ uncorrectable\ error) = (1 - \frac{\mathcal{P}}{W})^K$$

ANALYTICAL MODEL FOR ECC DIMM

W words in memory (strong rows only)

P words have 1 bit error already (AVP)

K new weak cells get injected in given time quanta

$$P(DIMM\ has\ no\ uncorrectable\ error) = (1 - \frac{\mathcal{P}}{W})^K$$

For T time quanta, and D DIMMS

ANALYTICAL MODEL FOR ECC DIMM

W words in memory (strong rows only)

P words have 1 bit error already (AVP)

K new weak cells get injected in given time quanta

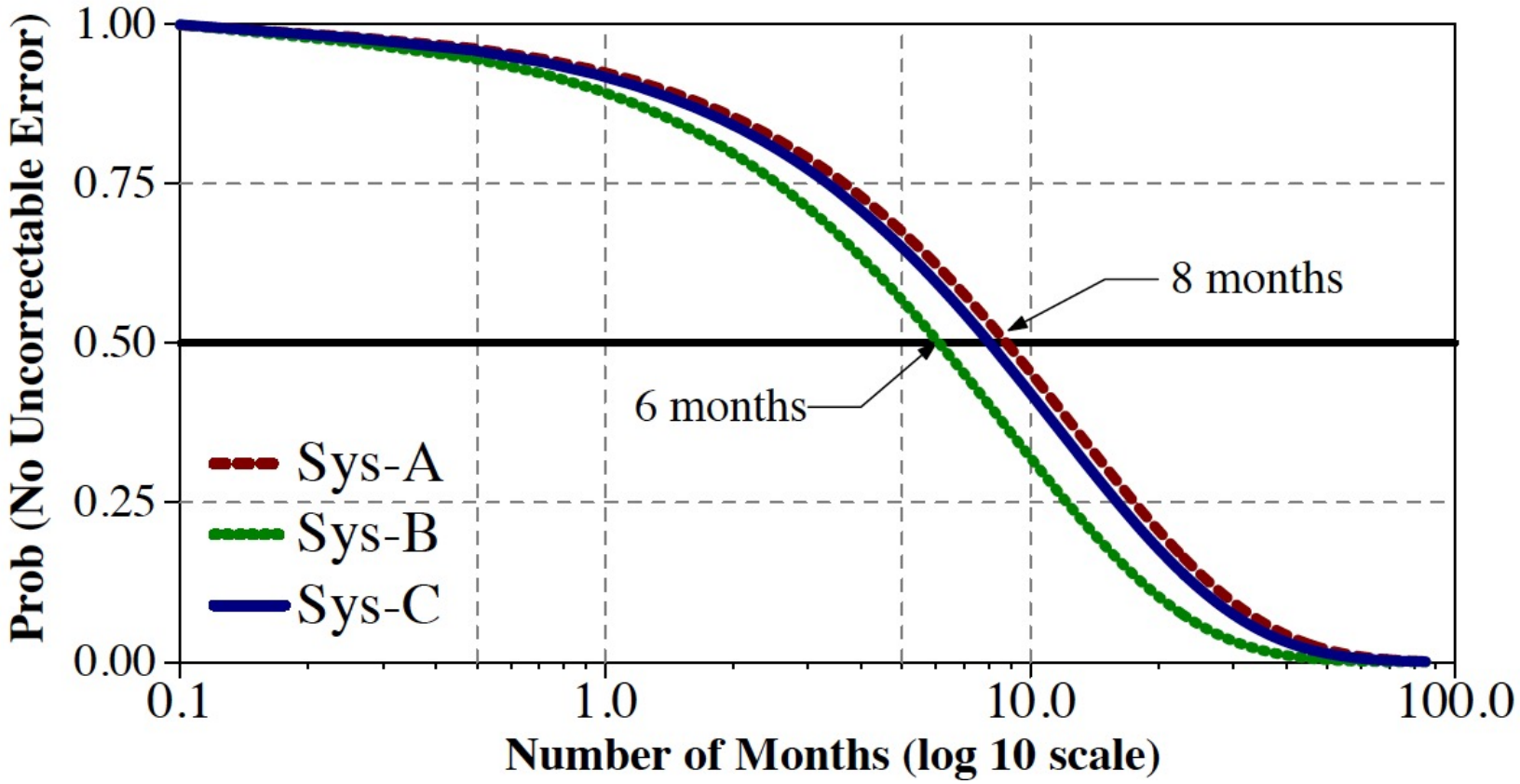
$$P(DIMM\ has\ no\ uncorrectable\ error) = (1 - \frac{\mathcal{P}}{W})^{K}$$

For T time quanta, and D DIMMS

$$P(System\ has\ no\ uncorrectable\ error) = (1 - \frac{\mathcal{P}}{W})^{K \cdot T \cdot D}$$

EVEN WITH ECC-DIMM, ERROR RATE IS HIGH

System: Four channels, each with 8GB DIMM



VRT still causes an error every ~6-8 months

OUTLINE

- Background
- VRT: mechanism, measurement, model
- Can't we fix VRT by simply using ECC DIMM?

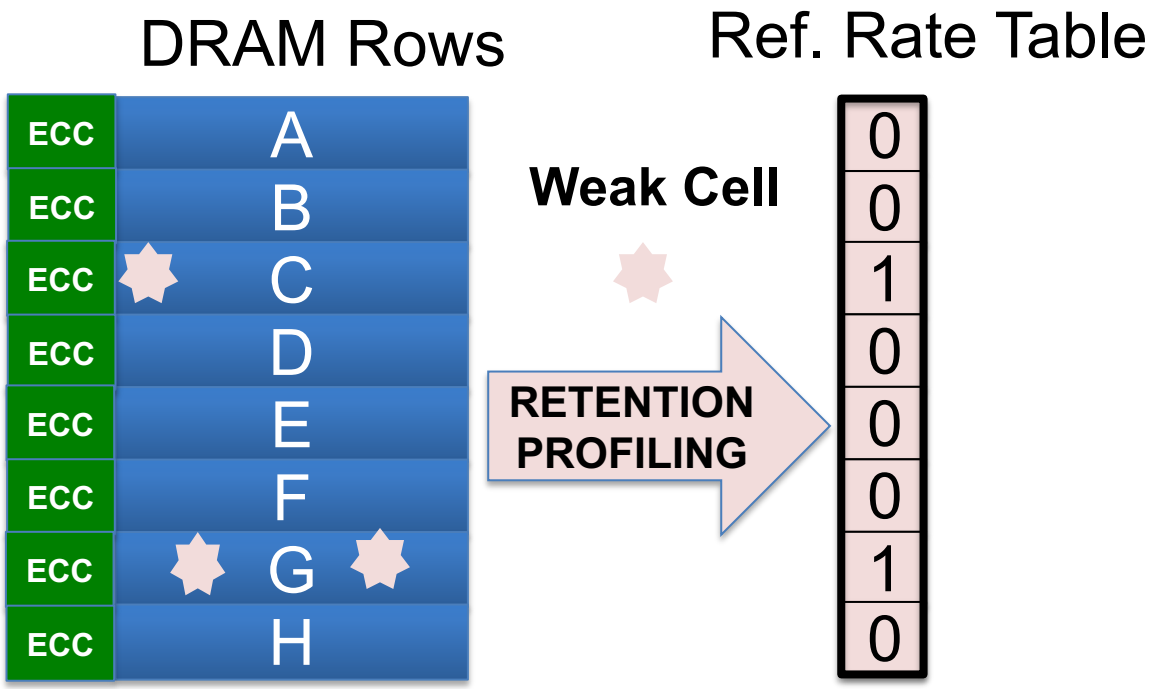
- AVATAR

- Results
- Summary

AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

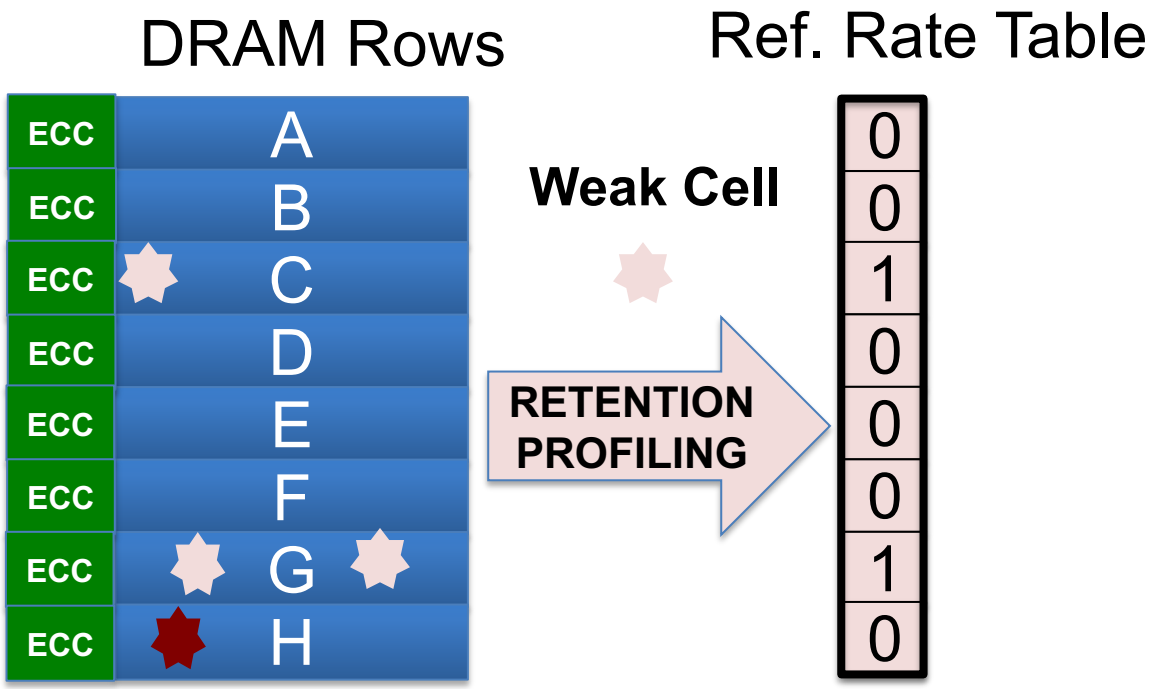
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

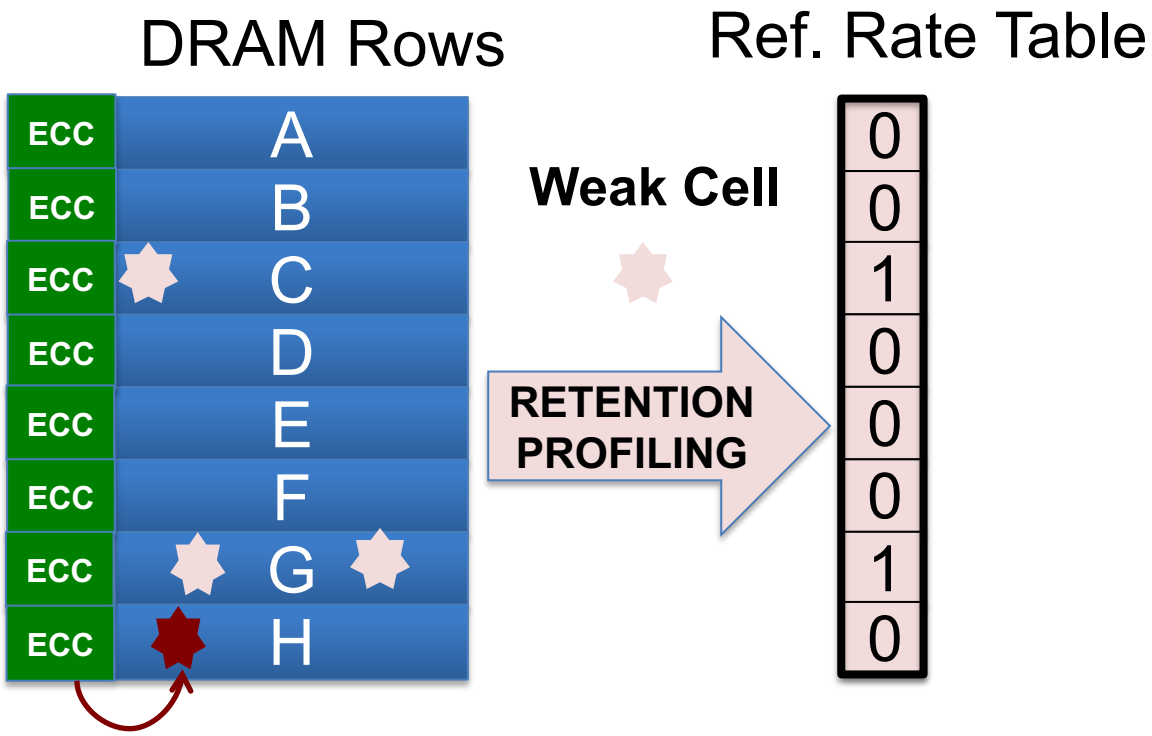
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

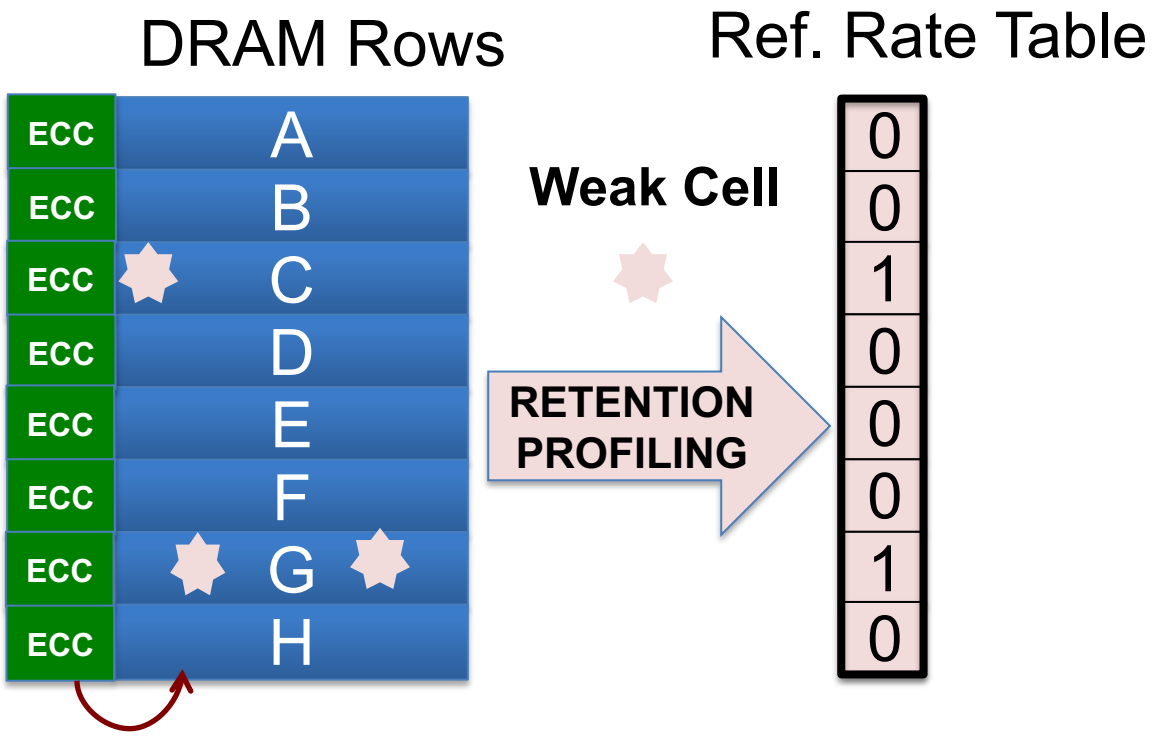
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

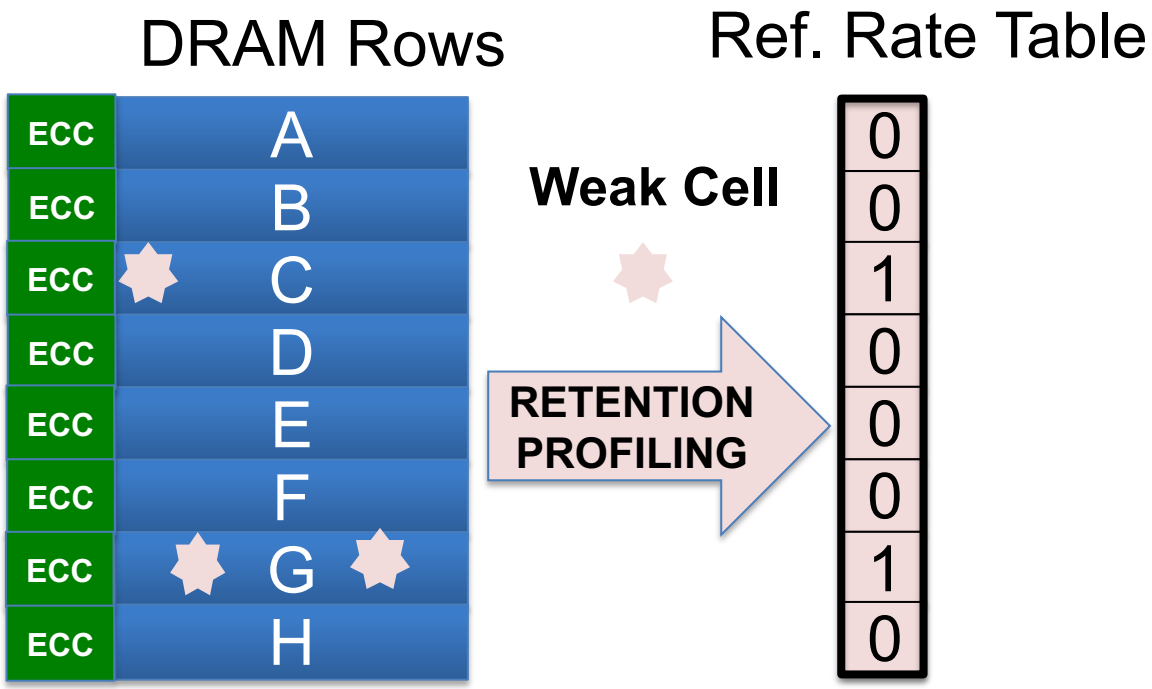
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

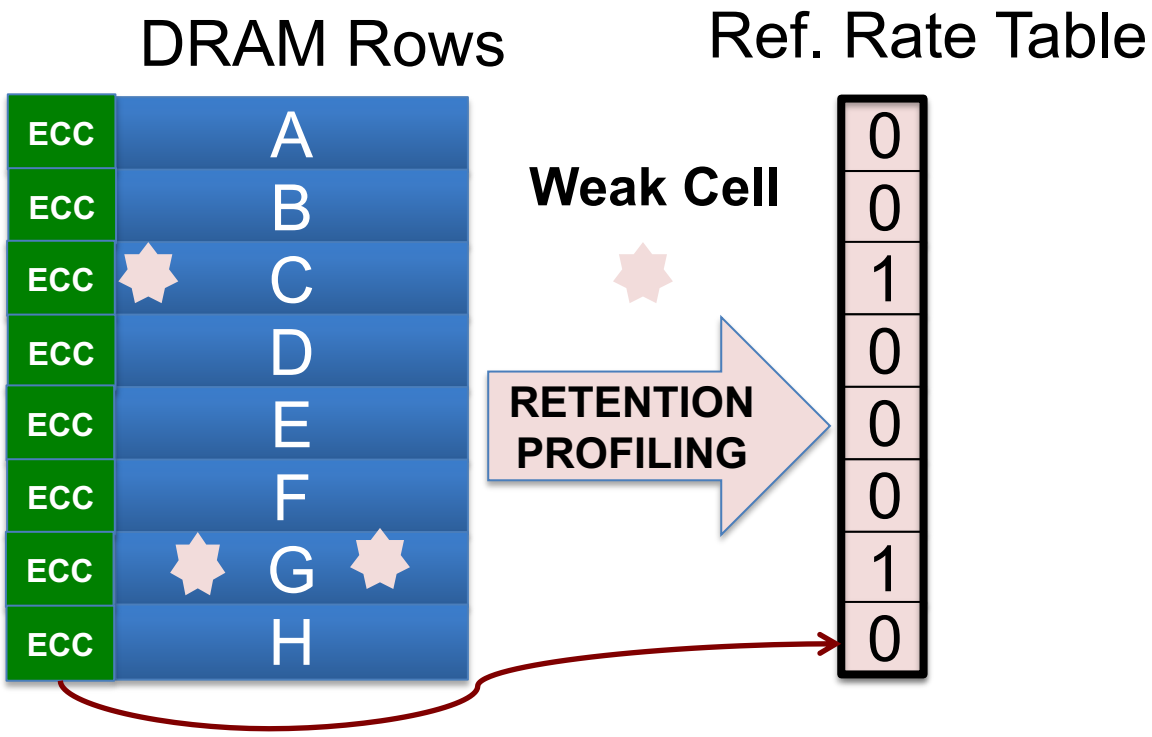
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

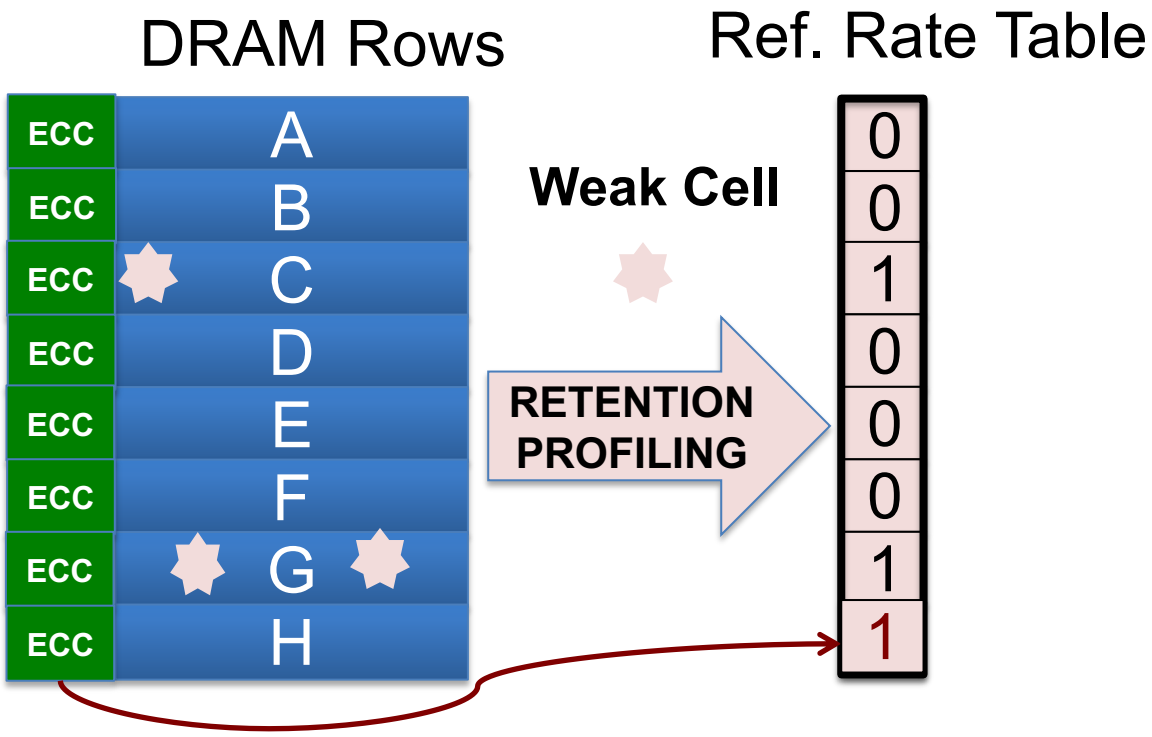
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

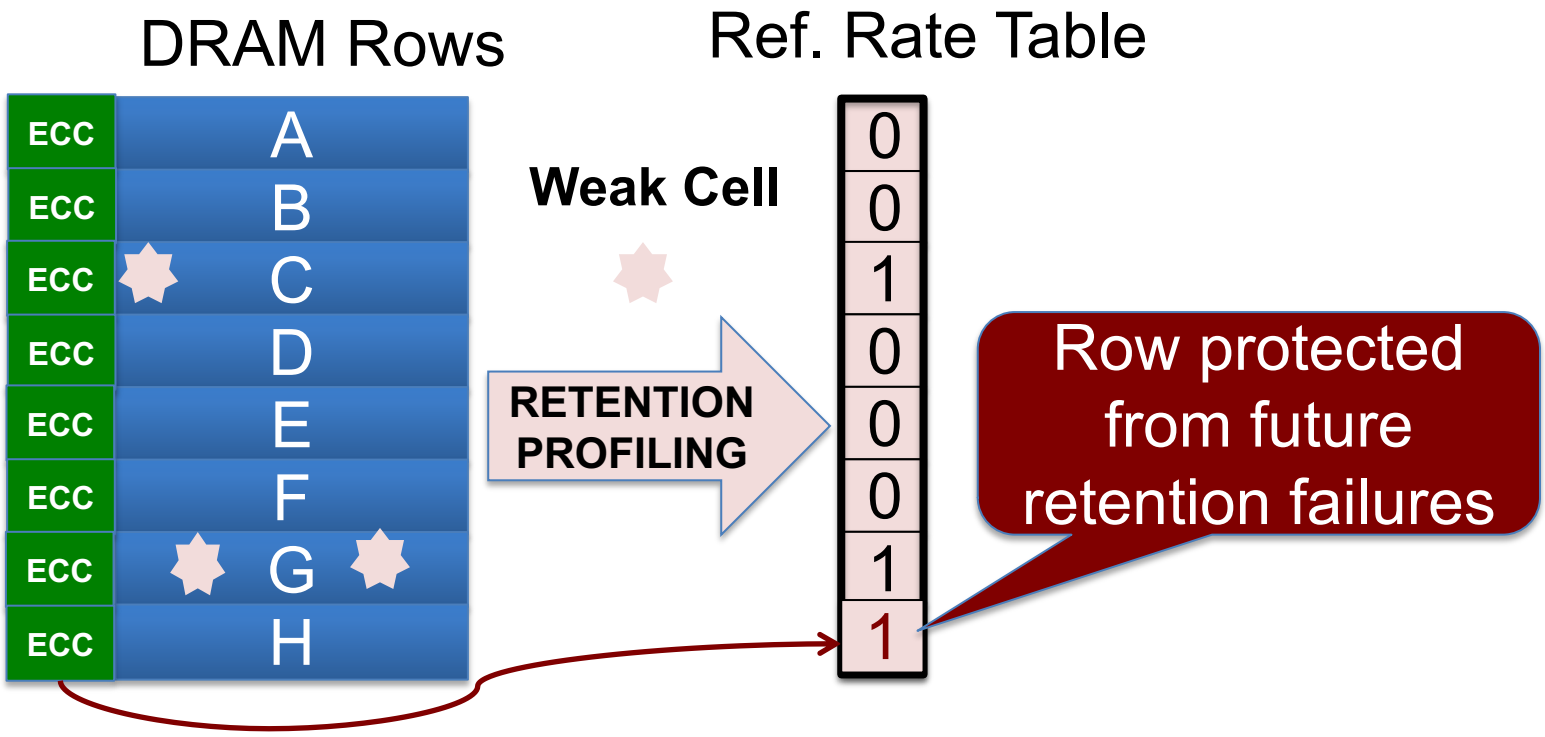
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

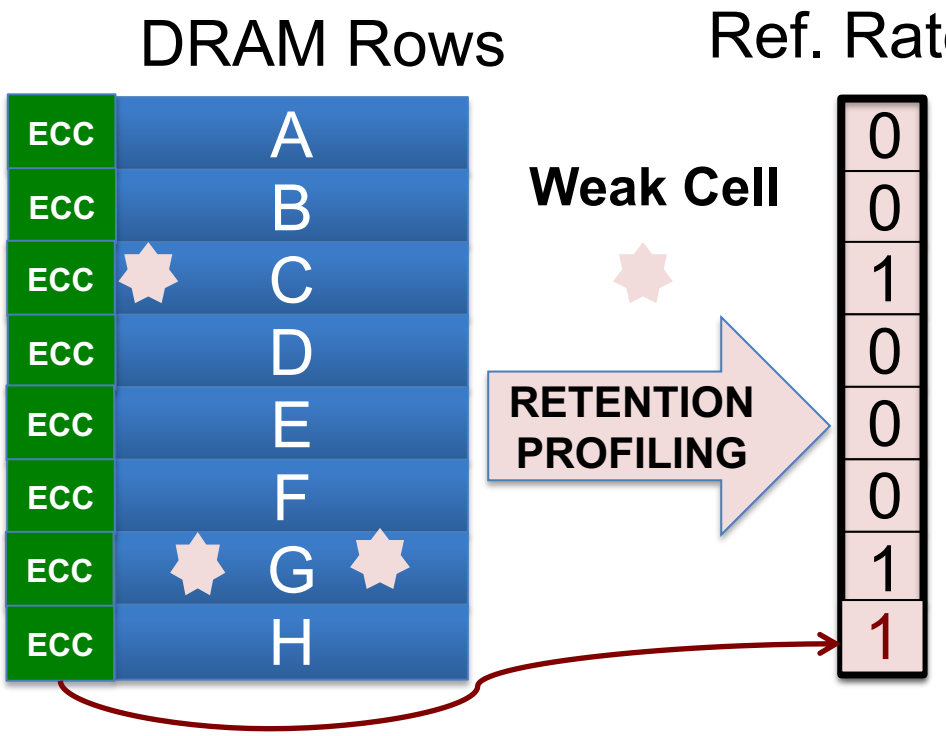
Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

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

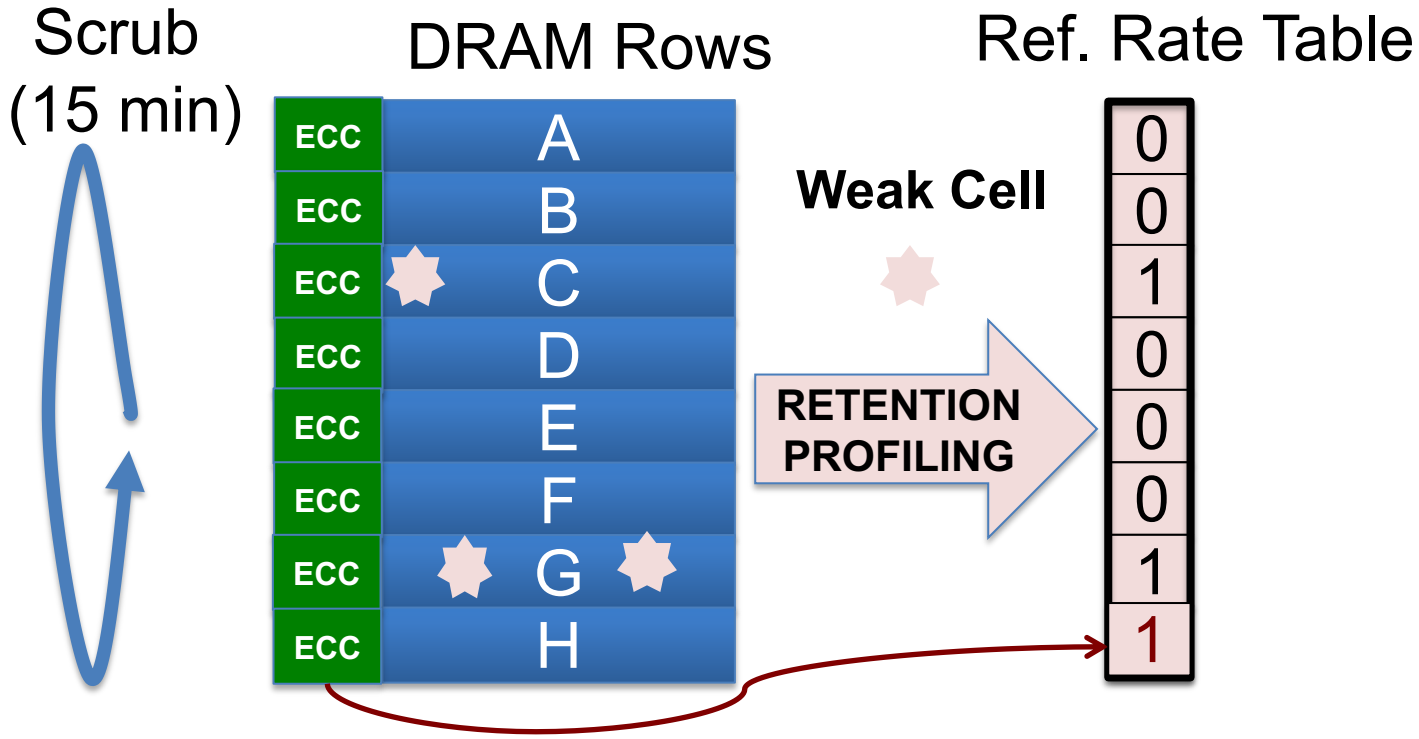


AVATAR mitigates VRT by breaking AVP Pool

AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

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

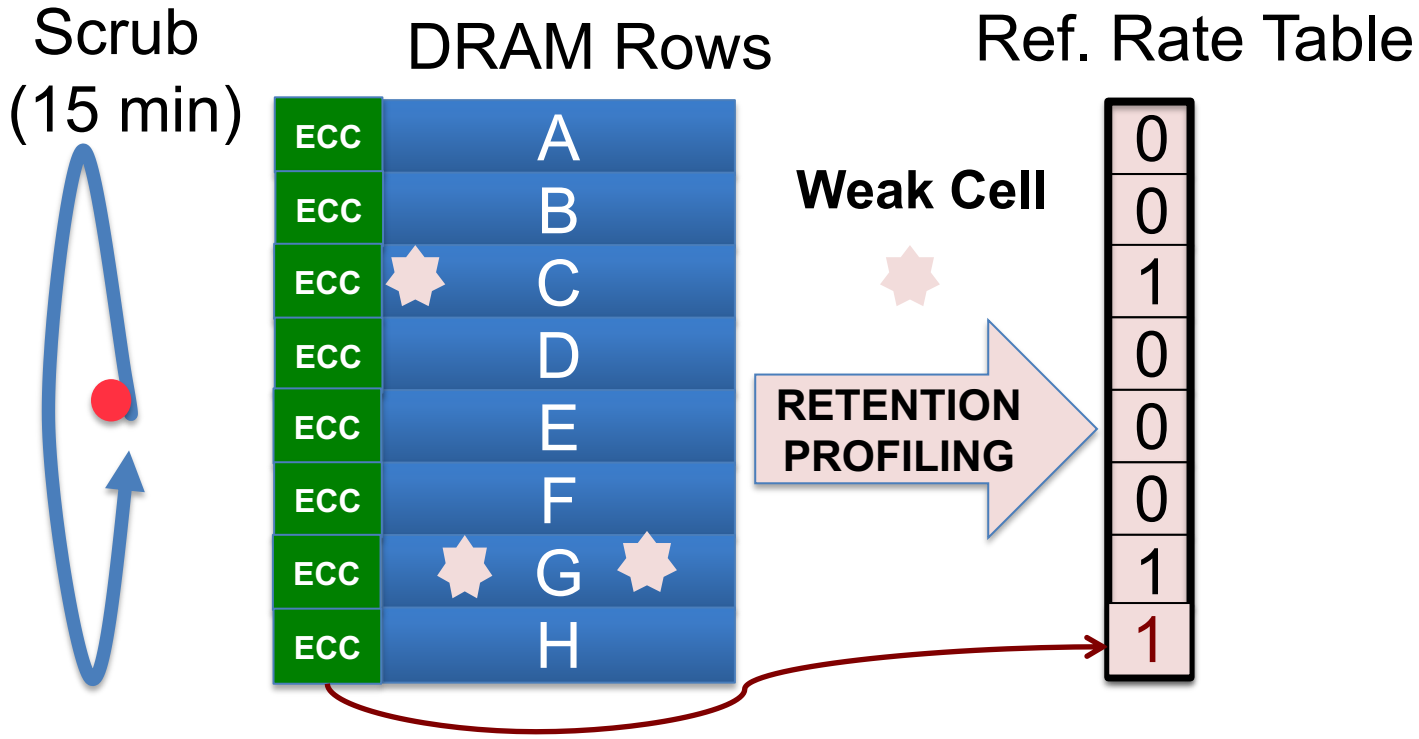


AVATAR mitigates VRT by breaking AVP Pool

AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

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

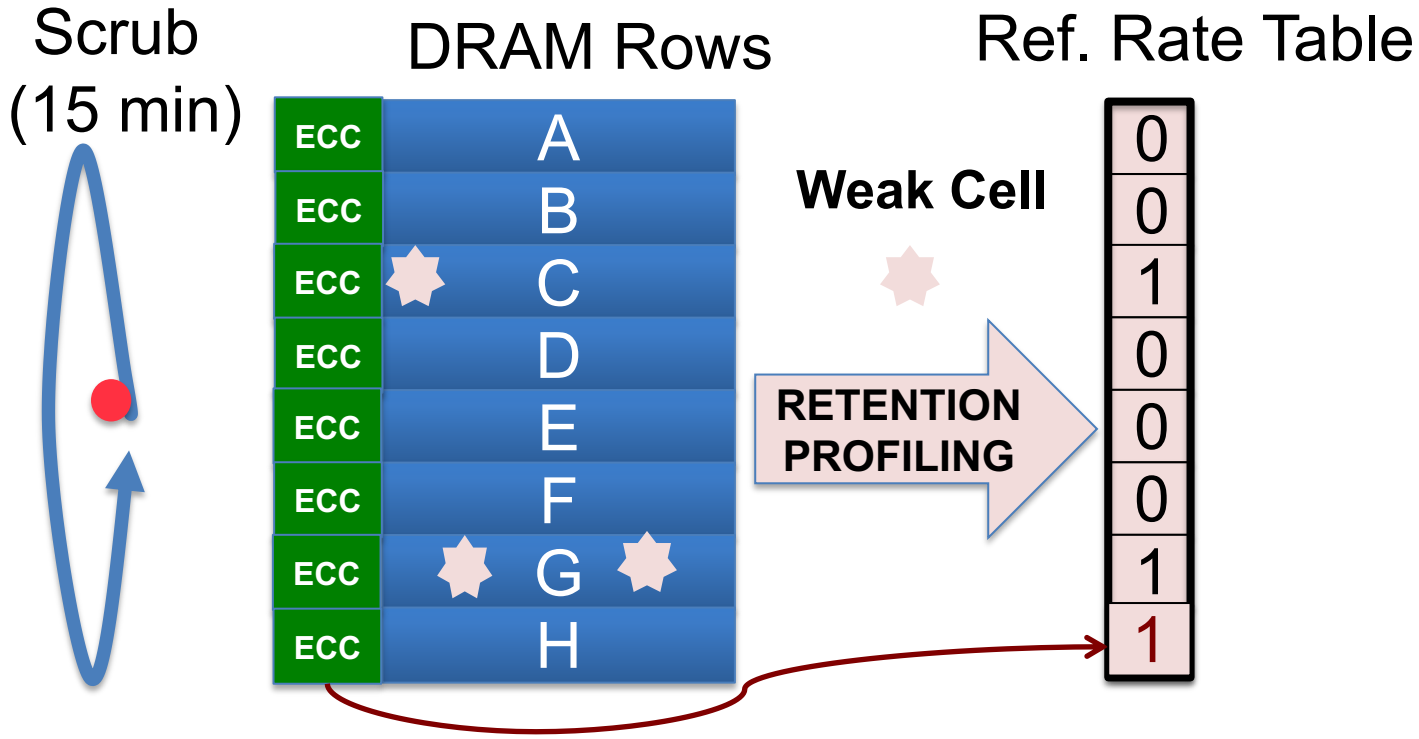


AVATAR mitigates VRT by breaking AVP Pool

AVATAR

Insight: Avoid forming Active VRT Pool ➔ Upgrade on ECC error

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



AVATAR mitigates VRT by breaking AVP Pool

AVATAR: ANALYTICAL MODEL

Only errors injected between scrub can clash with each other

Instead of 1000+ weak cells (AVP), deal with 4 errors (AVI)

AVATAR: ANALYTICAL MODEL

Only errors injected between scrub can clash with each other

Instead of 1000+ weak cells (AVP), deal with 4 errors (AVI)

W words in memory, K errors in time quanta (AVI Rate)

AVATAR: ANALYTICAL MODEL

Only errors injected between scrub can clash with each other

Instead of 1000+ weak cells (AVP), deal with 4 errors (AVI)

W words in memory, K errors in time quanta (AVI Rate)

$$Prob(DIMM\ has\ no\ uncorrectable\ error) = (1 - \frac{1}{W}) \times (1 - \frac{2}{W}) \times \ldots \times (1 - \frac{K-1}{W})$$

AVATAR: ANALYTICAL MODEL

Only errors injected between scrub can clash with each other

Instead of 1000+ weak cells (AVP), deal with 4 errors (AVI)

W words in memory, K errors in time quanta (AVI Rate)

$$Prob(\textit{DIMM has no uncorrectable error}) =$$
$$(1-\frac{1}{W}) \times (1-\frac{2}{W}) \times \ldots \times (1-\frac{K-1}{W}) \quad = e^{\frac{-K^2}{2W}}$$

AVATAR: ANALYTICAL MODEL

Only errors injected between scrub can clash with each other

Instead of 1000+ weak cells (AVP), deal with 4 errors (AVI)

W words in memory, K errors in time quanta (AVI Rate)

$$Prob(\textit{DIMM has no uncorrectable error}) = (1 - \frac{1}{W}) \times (1 - \frac{2}{W}) \times \ldots \times (1 - \frac{K-1}{W}) = e^{\frac{-K^2}{2W}}$$

For, T time quanta, and D DIMMS

AVATAR: ANALYTICAL MODEL

Only errors injected between scrub can clash with each other

Instead of 1000+ weak cells (AVP), deal with 4 errors (AVI)

W words in memory, K errors in time quanta (AVI Rate)

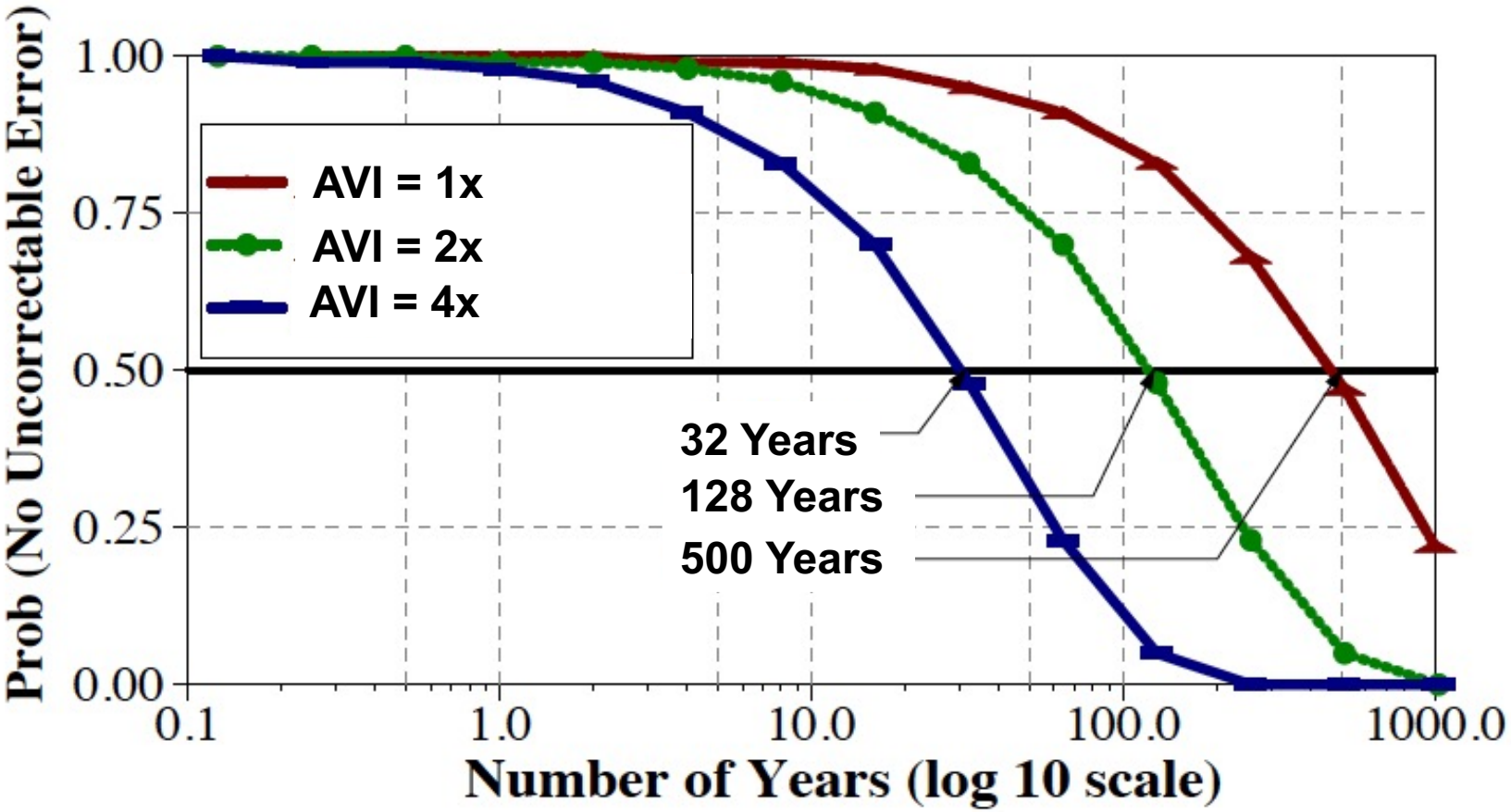
$$Prob(DIMM\ has\ no\ uncorrectable\ error) =$$
$$(1-\frac{1}{W}) \times (1-\frac{2}{W}) \times \ldots \times (1-\frac{K-1}{W}) \quad = e^{\frac{-K^2}{2W}}$$

For, T time quanta, and D DIMMS

$$Prob(System\ has\ no\ uncorrectable\ error) = e^{\frac{-DTK^2}{2W}}$$

AVATAR: TIME TO FAILURE

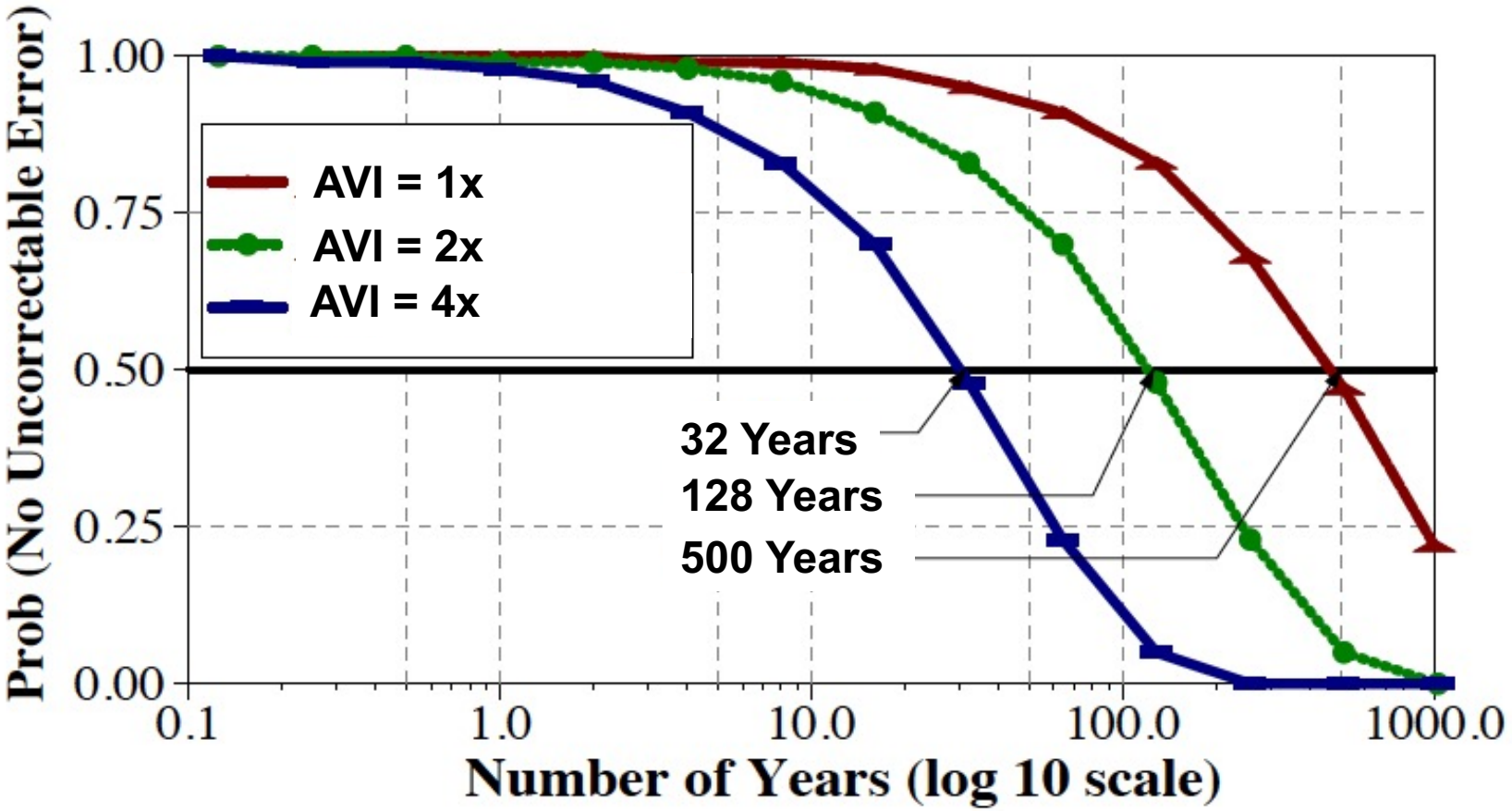
System: Four channels, each with 8GB DIMM



* We include the effect of soft error in the above lifetime analysis (details in the paper)

AVATAR: TIME TO FAILURE

System: Four channels, each with 8GB DIMM



AVATAR increases time to failure to 10s of years

* We include the effect of soft error in the above lifetime analysis (details in the paper)

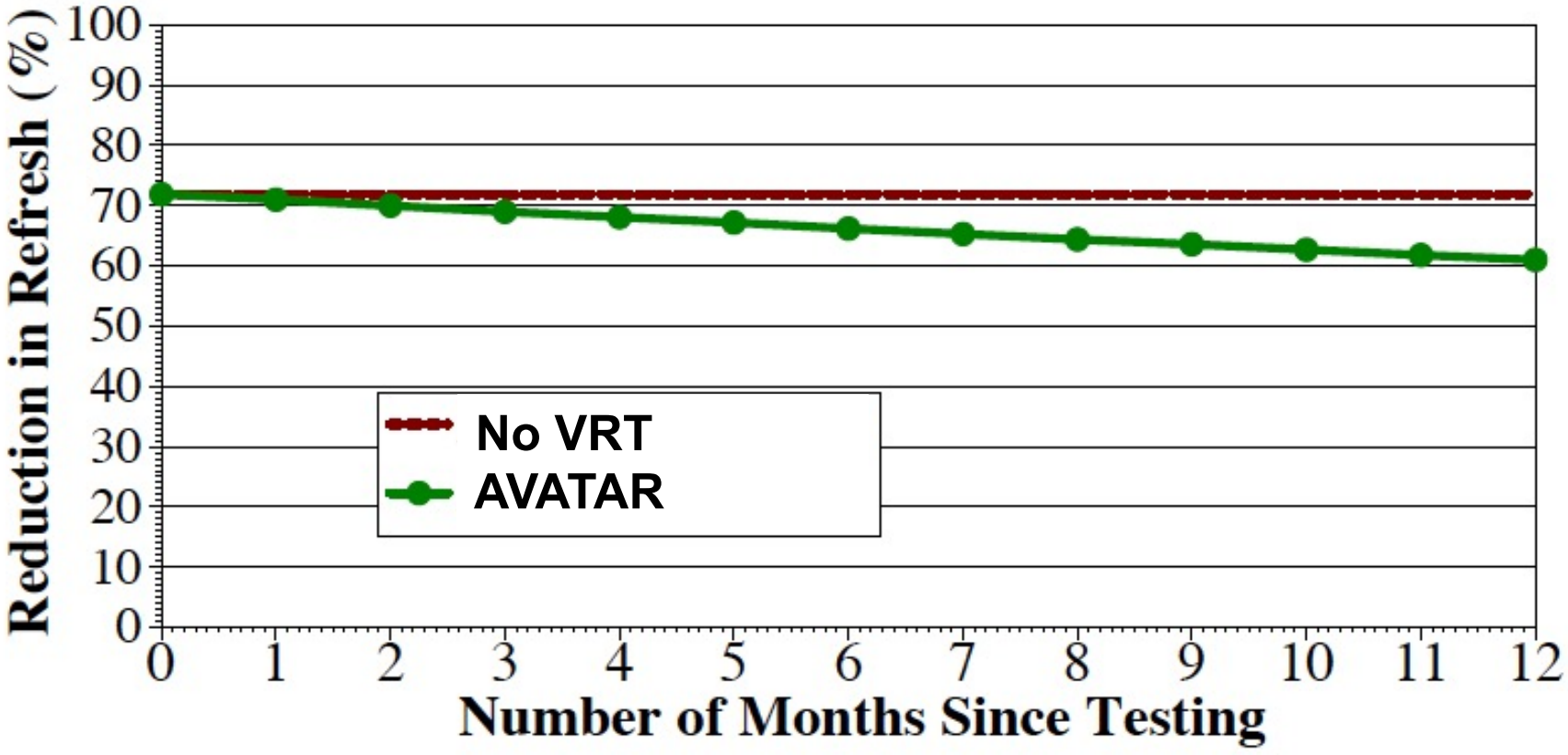
OUTLINE

- Background
- VRT: mechanism, measurement, model
- Can't we fix VRT by simply using ECC DIMM?
- AVATAR

- Results

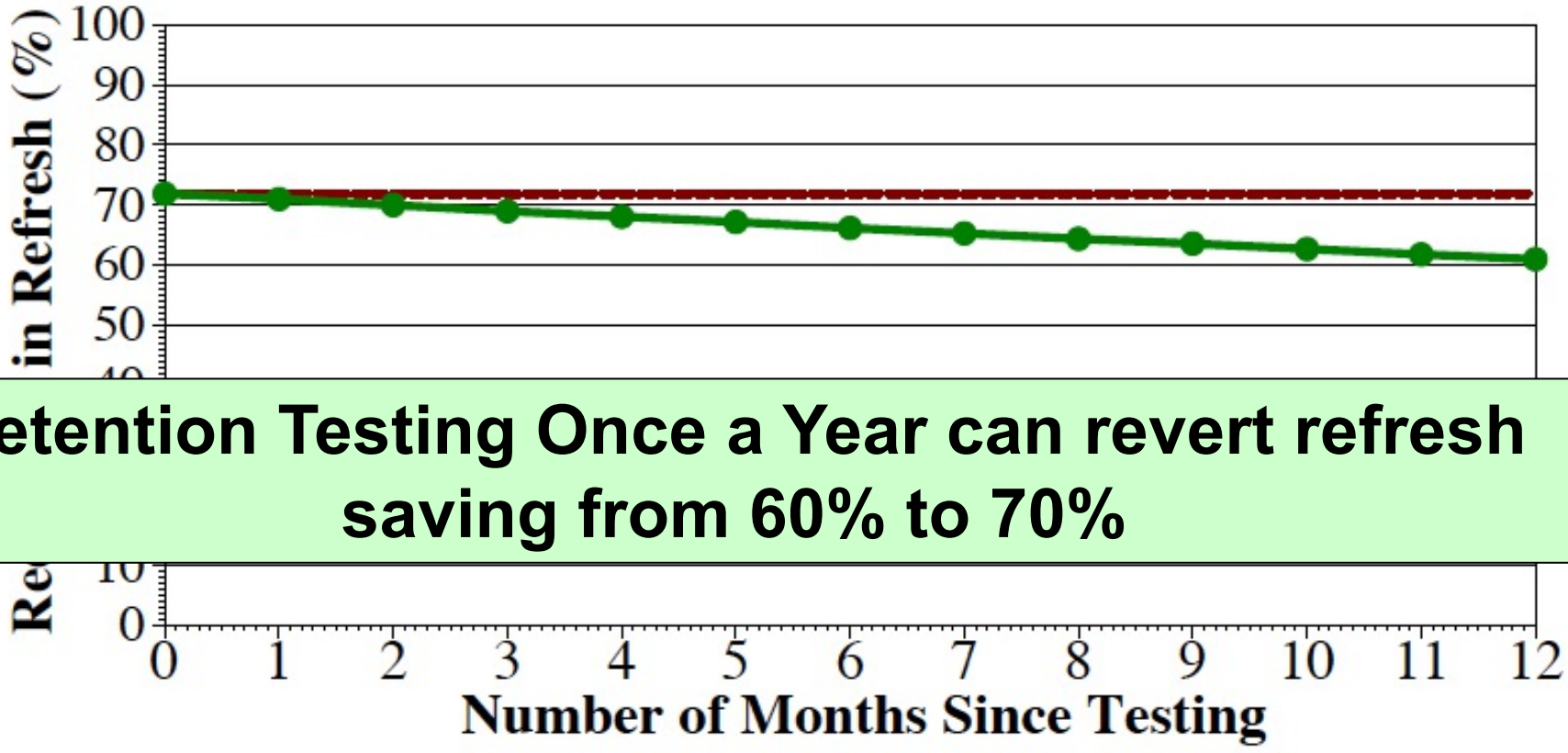
- Summary

RESULTS: REFRESH SAVINGS



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

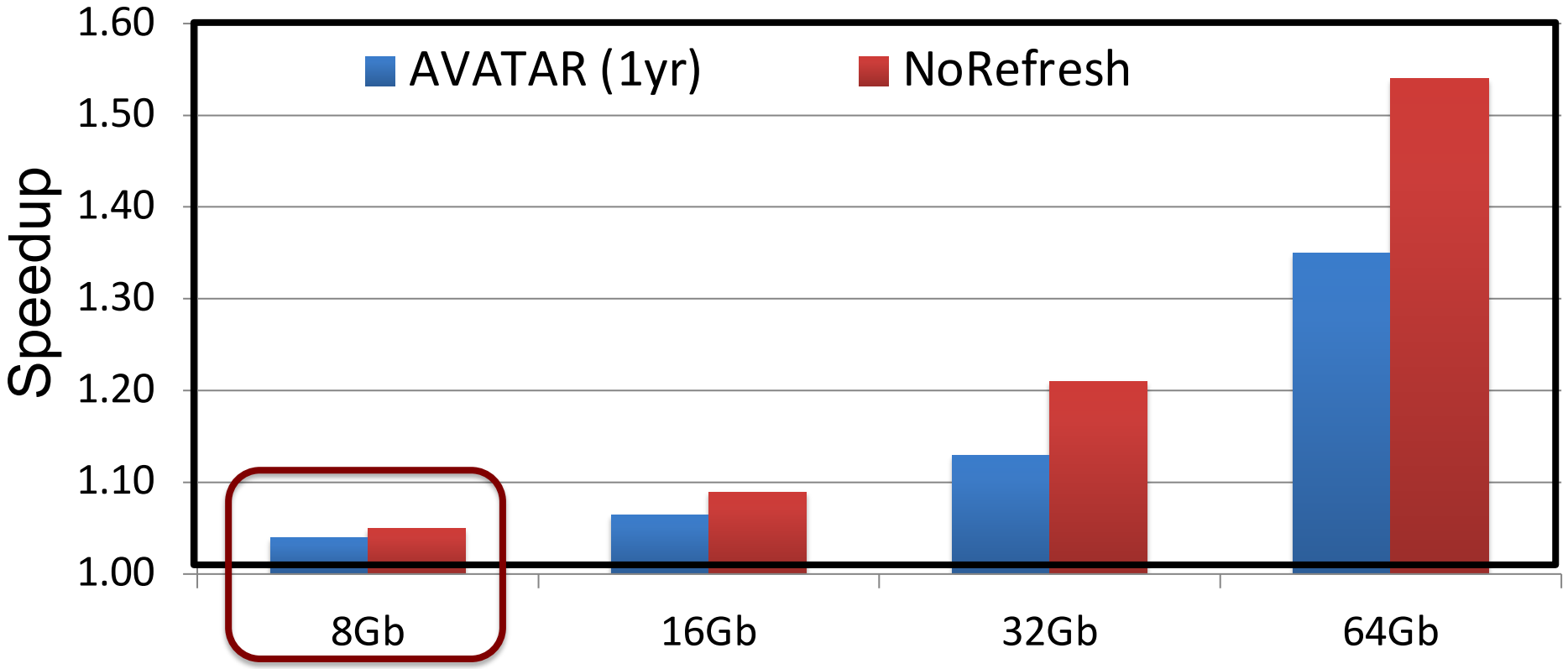
RESULTS: REFRESH SAVINGS



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

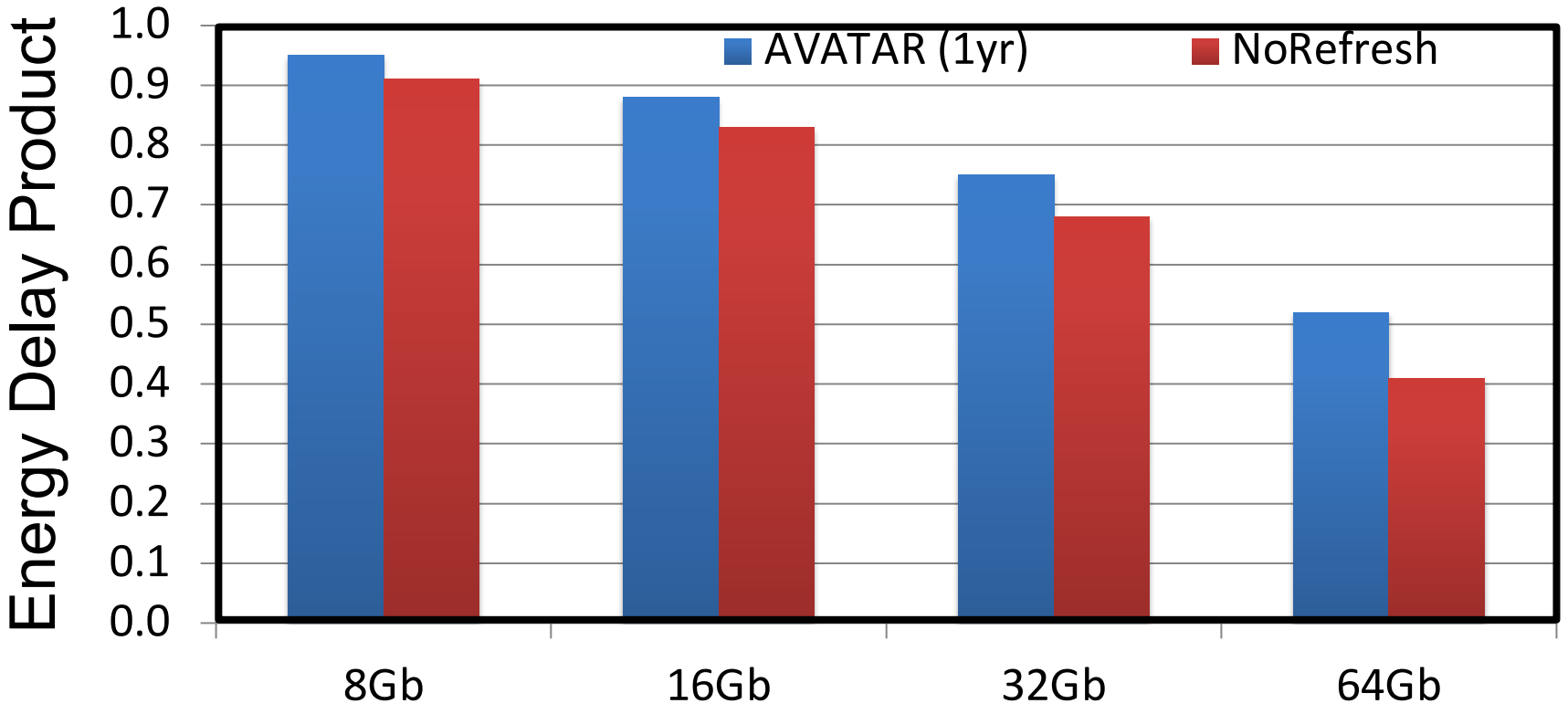
AVATAR reduces refresh by 60%-70%, similar to multi rate refresh but with VRT tolerance

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

OUTLINE

- Background
- VRT: mechanism, measurement, model
- Can’t we fix VRT by simply using ECC DIMM?
- AVATAR
- Results

- Summary

SUMMARY

Multirate refresh ➔ retention profiling to reduce refresh

Variable Retention Time ➔ errors with multirate refresh

✓Architecture model of VRT based on experiments
✓We show ECC DIMM alone is not enough
✓AVATAR (upgrade refresh rate of row on ECC error)

AVATAR increase the time to failure from 0.5 years to 500 years and incurs the same storage as ECC DIMM