Incremental Redundancy to Reduce Data Retention Errors in Flash-based SSDs

31st International Conference on Massive Storage Systems and Technology

Heejin Park, Jaeho Kim, Jongmoo Choi, Donghee Lee, Sam H. Noh
University of Seoul, Dankook University, Hongik University

Outline

- **Introduction & Approach**
- Estimate Error and Period
- Effective OPS size and Hot/Cold Separation
- Experiments & Conclusion

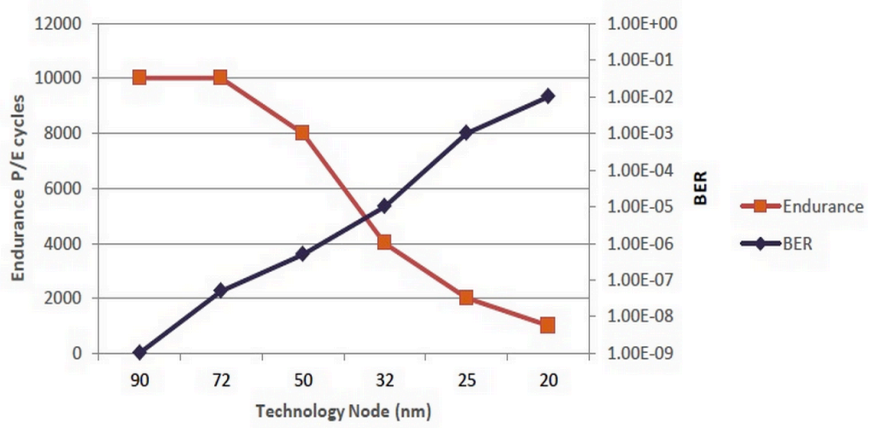
Flash-based SSD

- Flash based SSD is widely used in many areas
  - Computer system
  - Mobile devices

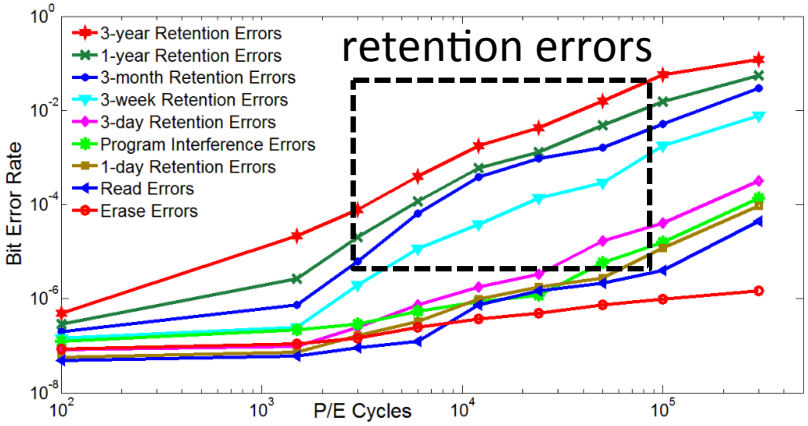
- Why flash memory?
  - Superior performance
  - Low power consumption
  - Shock resistance
  - Small size
  - Light weight

- Multi bit-cell
  - Low price
  - **Reliability issues are being magnified**

Reliability of SSD



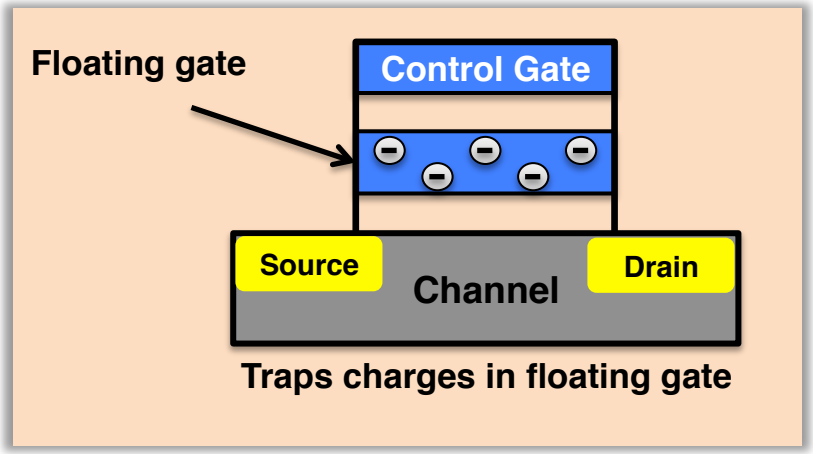
From: Hagop Nazarian and Sylvain Dubois, Crossbar
"The drive for SSDs: What's holding back NAND flash?"



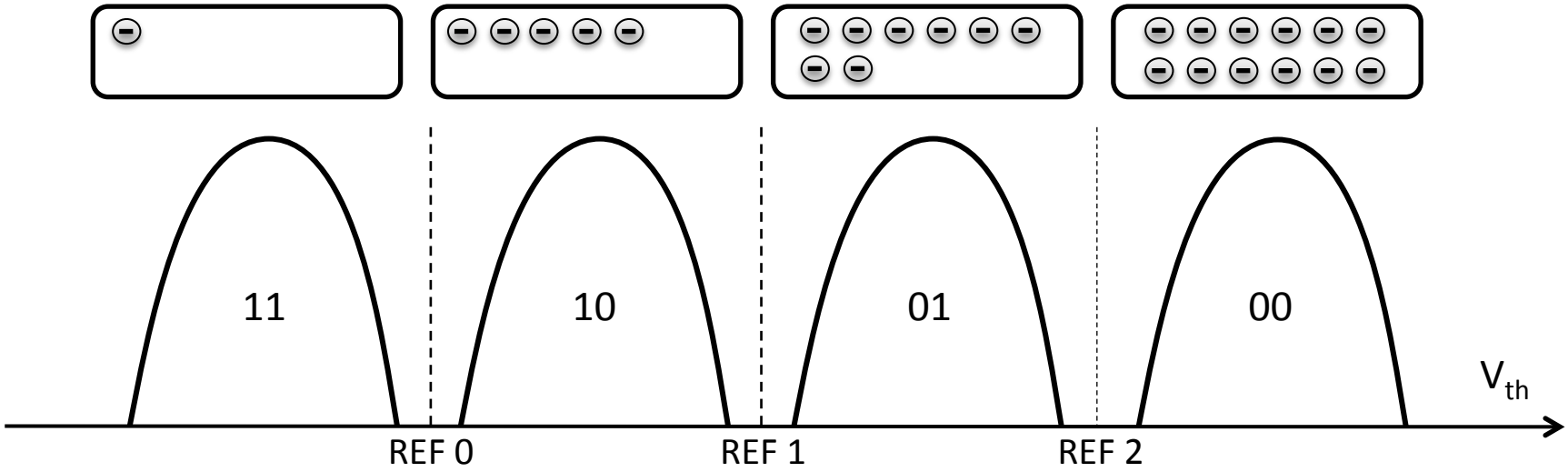
From: Cai, et al, "Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis", DATE' 12

- Manufacturing process is being developed
  - Endurance decreases
  - High bit error rate (short retention period)
  - More serious in MLC & TLC
- Types of error in NAND flash based SSDs
  - Program error
  - Read error
  - **Retention error (dominant error)** ← ***Our work is focused on here***

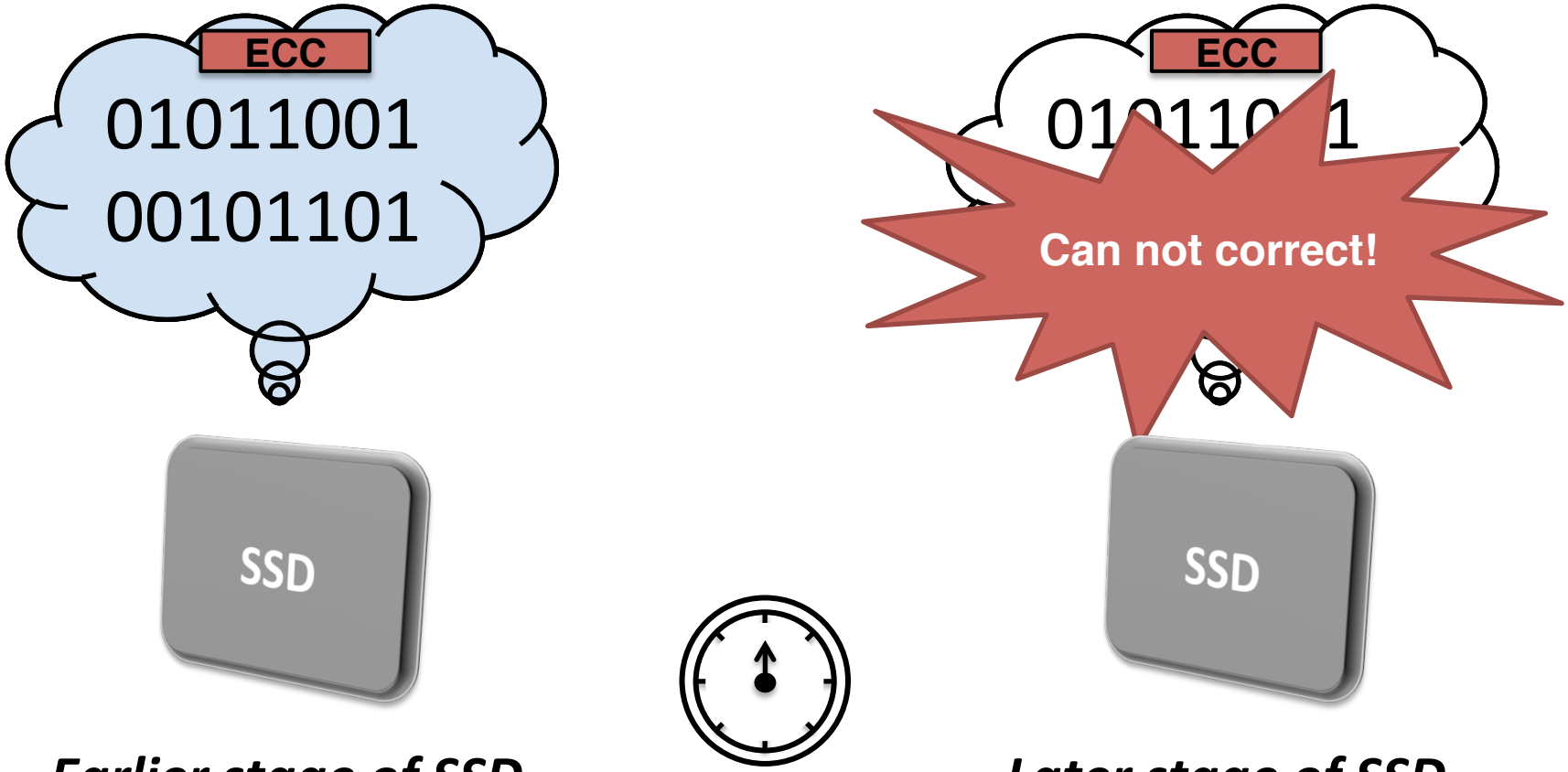
Cause of Retention Error



- **Retention error**
  - Errors due to long data retention period
  - Leaking of injected electrons
  - Shifting of threshold voltages



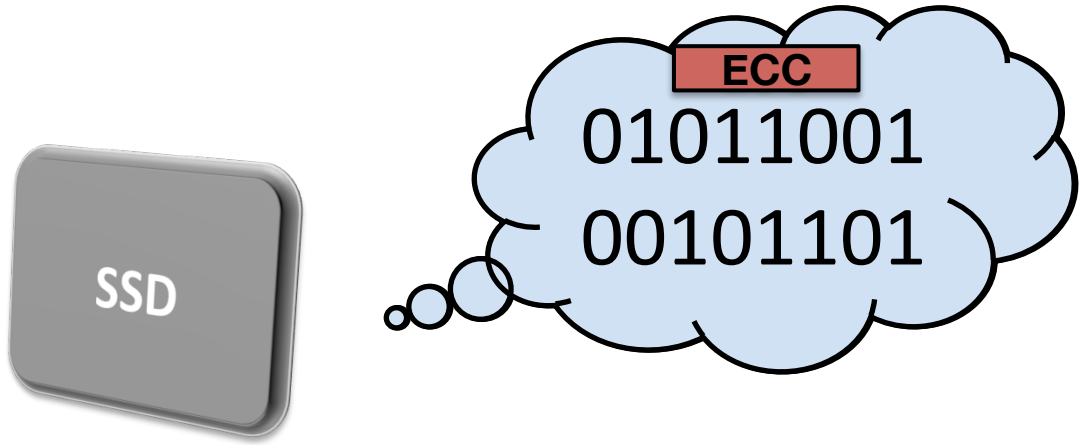
Problem of Retention Error



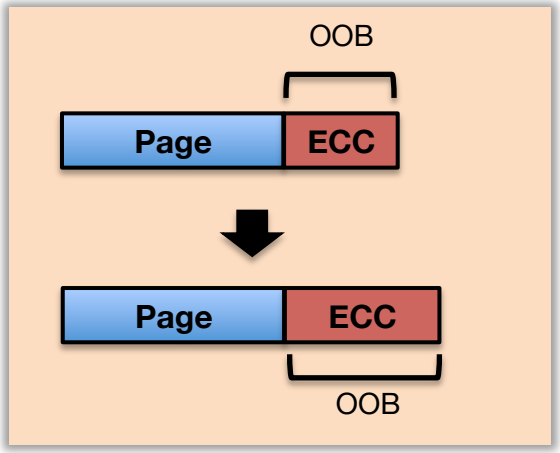
Earlier stage of SSD

Later stage of SSD

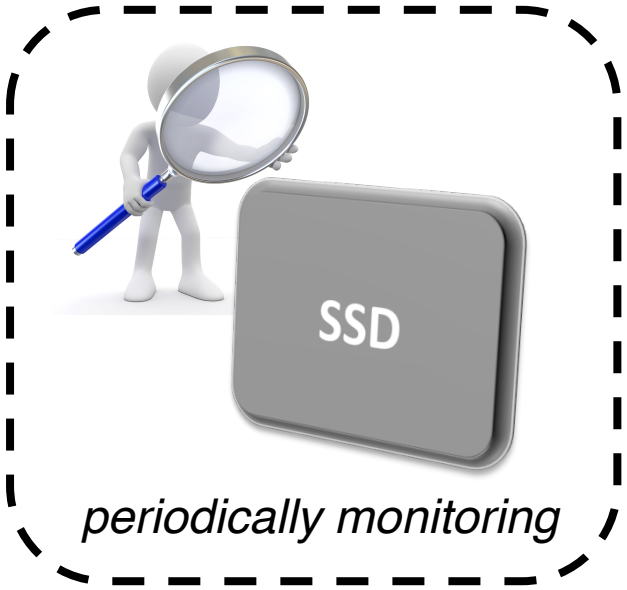
Existing Solutions

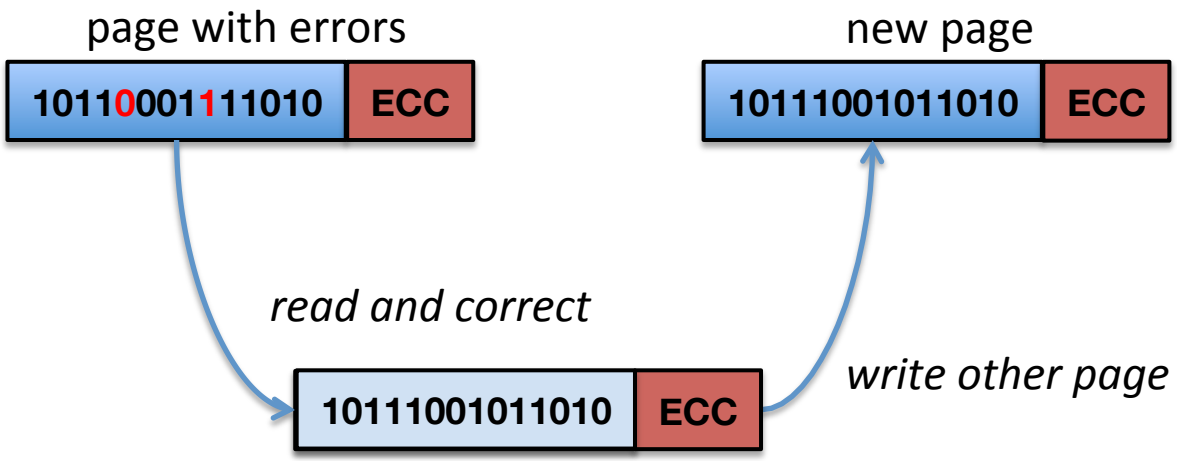


- **Strong ECC**
  - ***Reinforce the error correction capability***
    - Require considerable OOB space
    - The errors in cold data can't be corrected
    - Hurts performance and energy efficiency



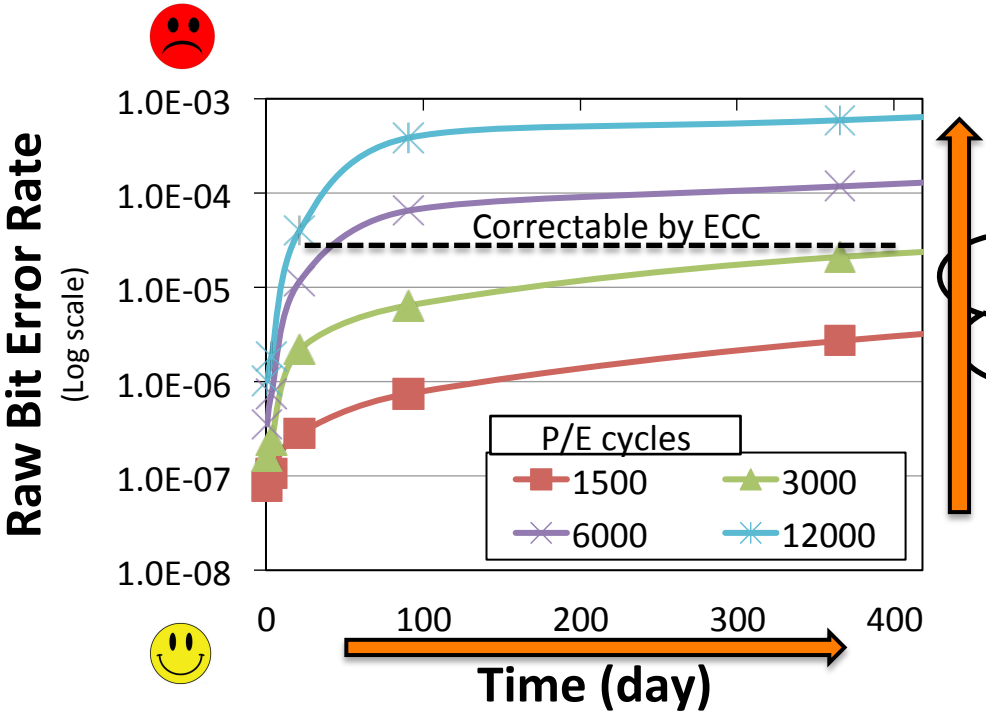
Existing Solutions (Cont.)



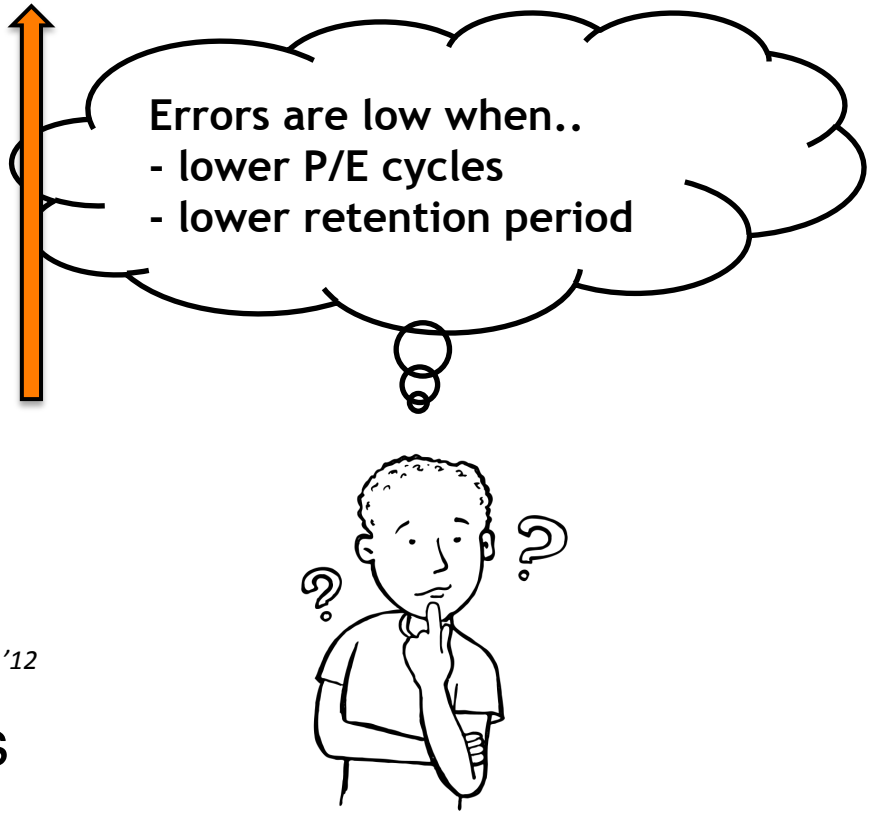


- **Data scrubbing**
  - ***Rewrite data after reading and correcting by using ECC***
    - Extra reads, writes, and erase operations
    - Hurts performance and lifetime

Observation

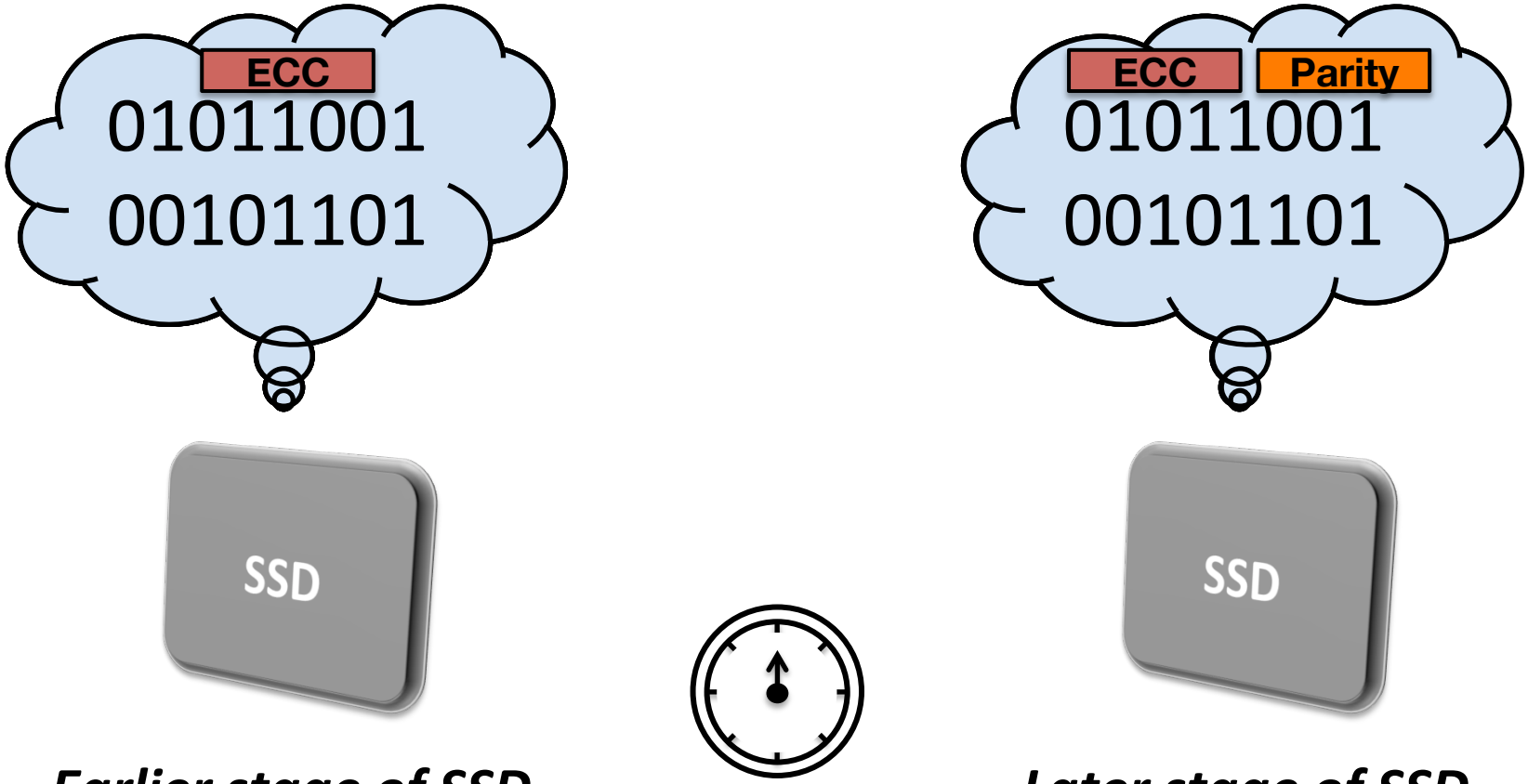


From : Cai, et al. "Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis," in *Proc. DATE '12*



- Retention errors increases as
  - **P/E cycles** increases
  - **Time** goes on

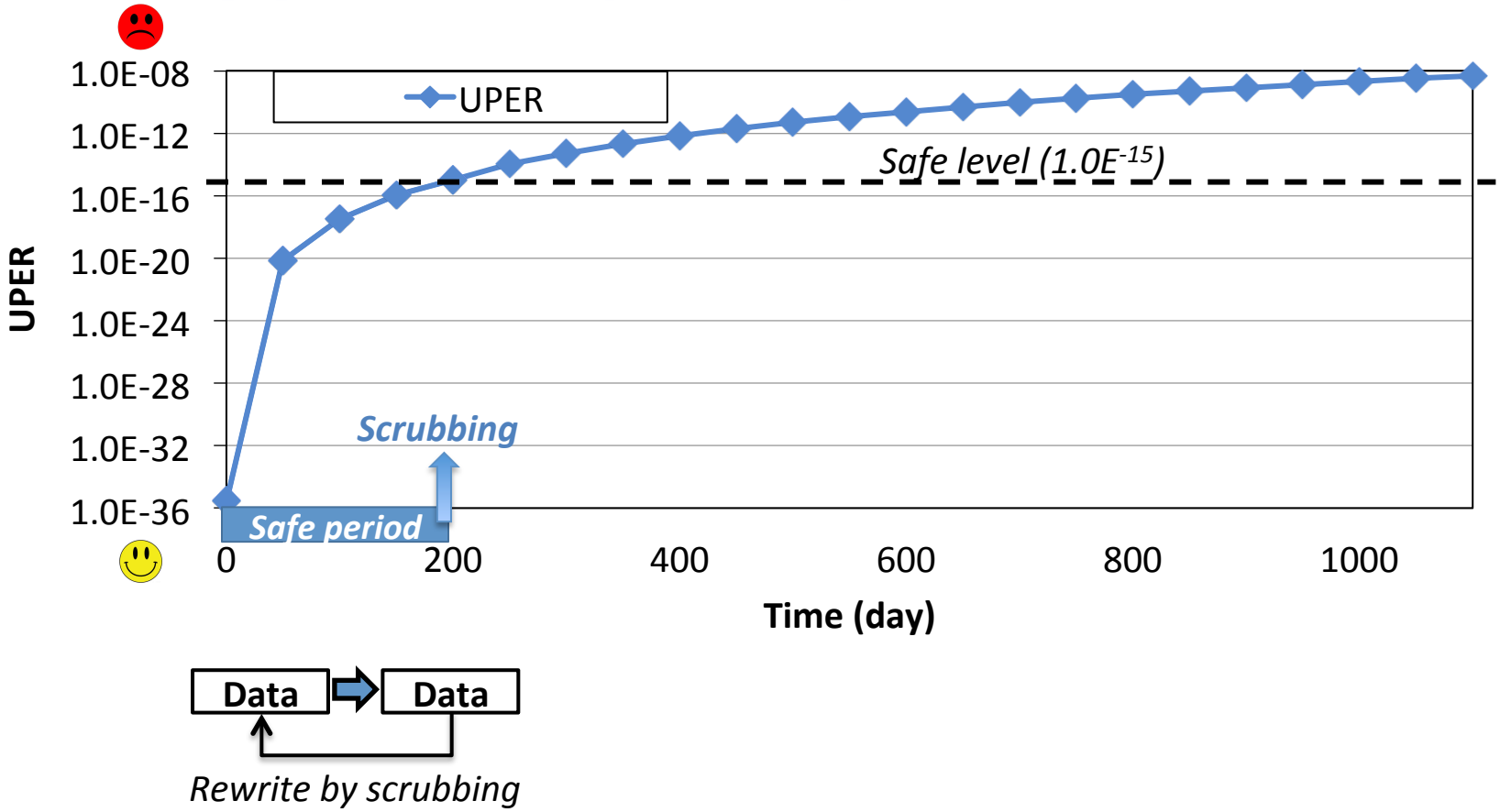
Our Approach



Earlier stage of SSD ***Later stage of SSD***

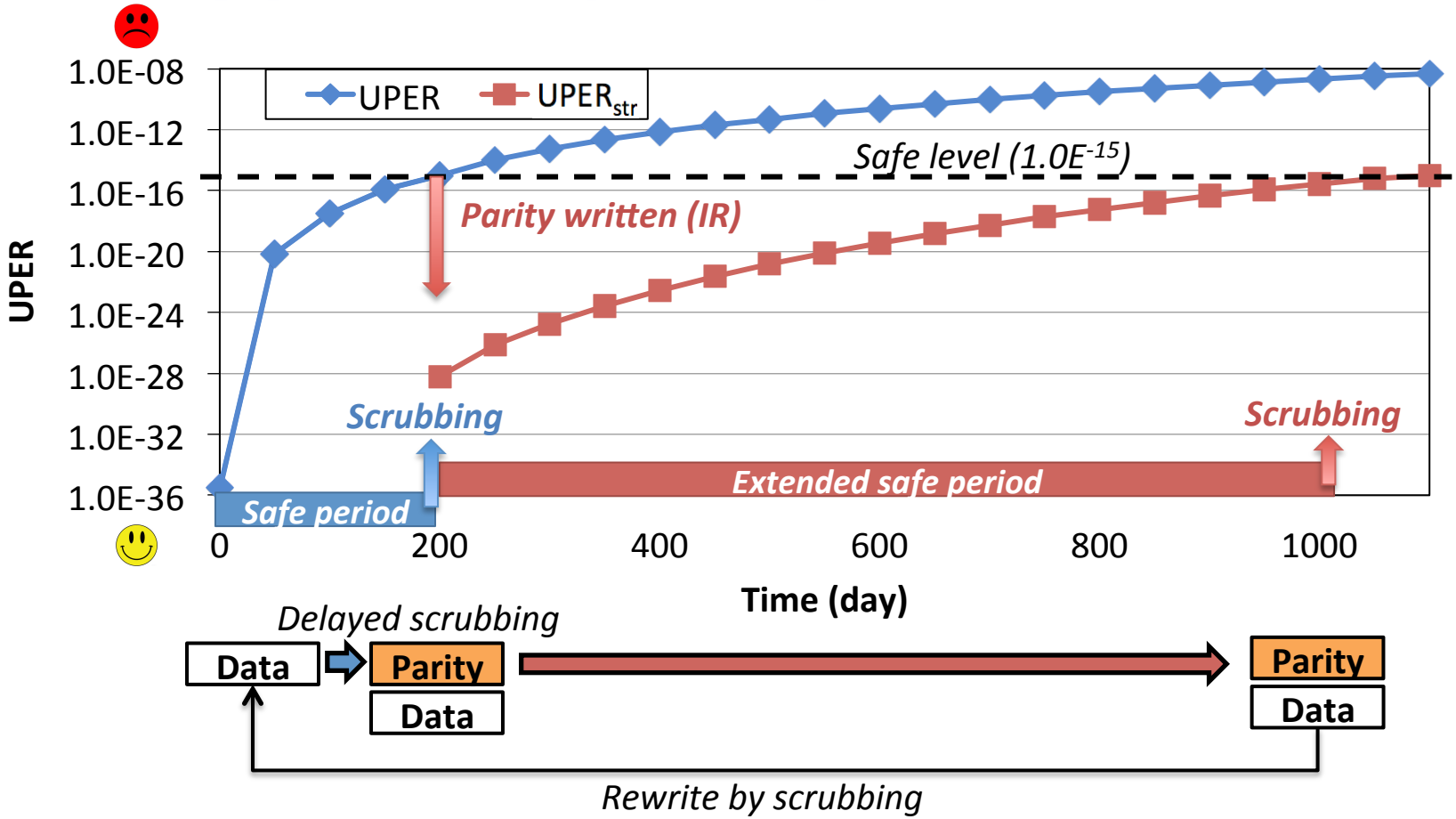
- ***Write parity considering P/E cycles & retention period***

Our Approach (Cont.)



- ***Safe level*** *: error rate level which is considered unsafe (1.0E-15)*
- ***Safe period*** *: duration of the error rate under safe level*

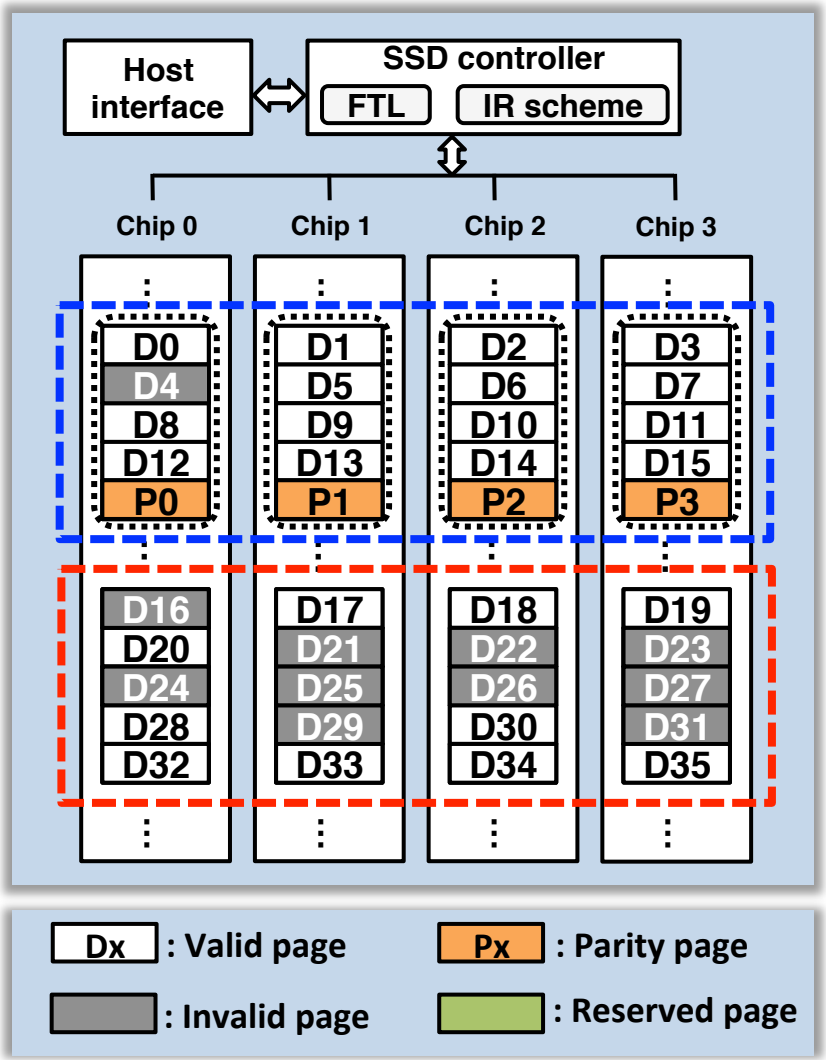
Our Approach (Cont.)



- **Incremental Redundancy (IR)**
  - When error rate reached at safe level, write parities dynamically
  - Incrementally reinforce the data recovery capability

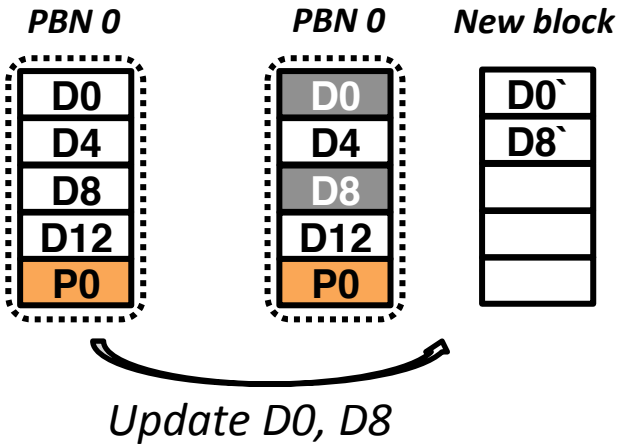

How to apply parity?

IR with Dynamic Vertical Stripe

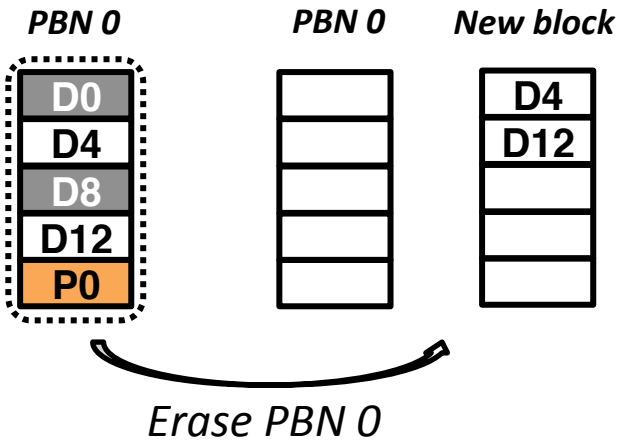


- **Construct stripe dynamically**
  - Reserve some pages
- **Reserved Pages**
  - **Hot Block**
    - Handle write requests
  - **Cold Block (stripe)**
    - Write parities when safe period expires
    - Reinforce error correct capability
- **Low overhead**
  - Parity update cost
  - GC cost

Operation of Vertical Stripe



- **Write operation**
  - Invalidate old data
  - New data is written to new page
  - Old data still exist until GC operation
  - *Parity does not need to be updated*

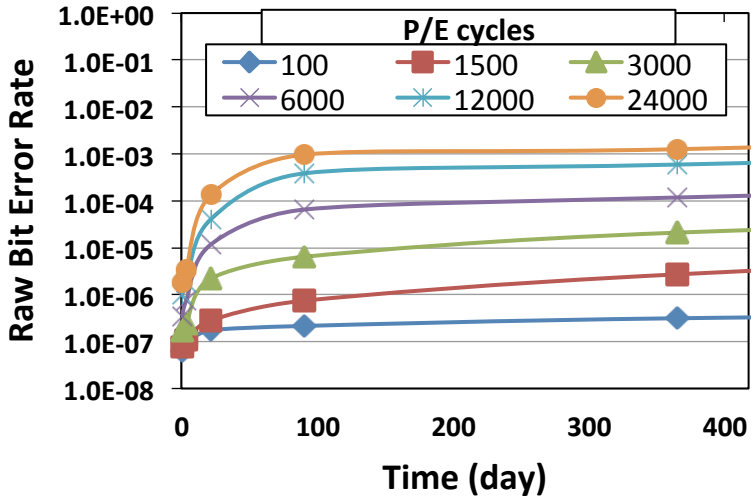


- **GC operation**
  - Unit of victim is block
  - Valid data are migrated
  - Parity & invalid data disappear
  - *Can be done as general GC*
  - *Does not need to reconstruct stripe*

Outline

- Introduction & Approach
- **Estimate Error and Period**
- Effective OPS size and Hot/Cold Separation
- Experiments & Conclusion

Analysis of Error Rate



- **Estimate raw bit error rate**
  - Key factors of aggravating retention error
    P/E cycles ***Time (day)***
  - Make formula form via curve fitting

$$RBER(c,d) = d_r(c) \cdot d$$

dr : deterioration rate function
c : P/E cycles d : day

- **Compare error rate before and after striping**
- **U**ncorrectable **P**age **E**rror **R**ate

$$UPER(n,k) = 1 - CPER(n,k)$$

*- CPER : **C**orrectable **P**age **E**rror **R**ate*
*- CSER : **C**orrectable **S**tripe **E**rror **R**ate*

- **U**ncorrectable **P**age **E**rror **R**ate with stripe

$$UPER_{str}(N,P,c,d) = \frac{1 - CSER_{str}(N,P,c,d)}{N}$$

Safe Period & CG Period

- **Safe period**
  - *Safe period (T_{safe}) : duration of the error rate under safe level*
  - *Extended safe period (T_{esafe}) : safe period with parity*

- RBER threshold makes UPER = safe level(1.0E^{-15})

- **GC period**
  - T_{gc} : duration until data will be selected as victim of GC

P_u : update probability
u : utilization

- Write parity when
  - Error rate ***exceeds safe level*** before garbage collection
  - Data retention period ***reaches at safe period***

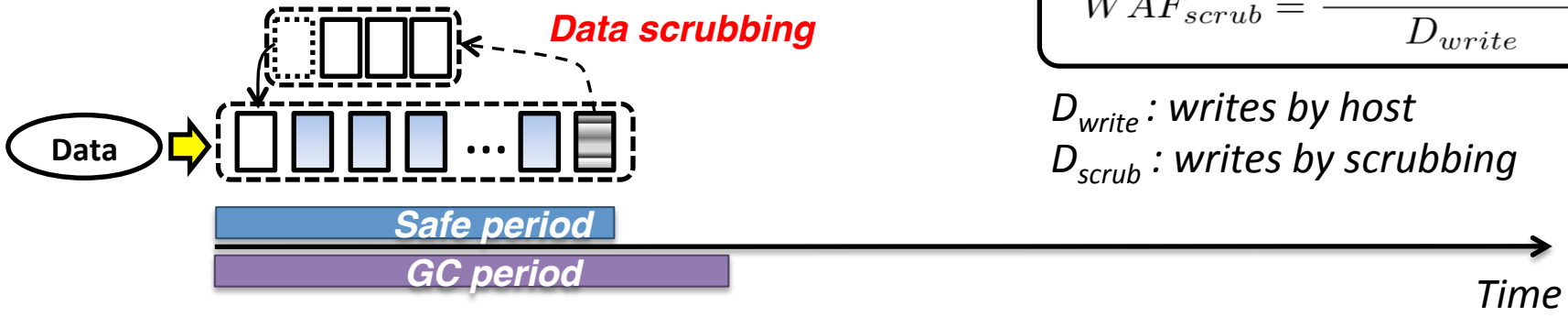
Safe Period vs. GC Period

- **GC period < safe period**

$$WAF_{GC} = \frac{1}{1-u}$$

u : utilization of block

- **GC period > safe period**



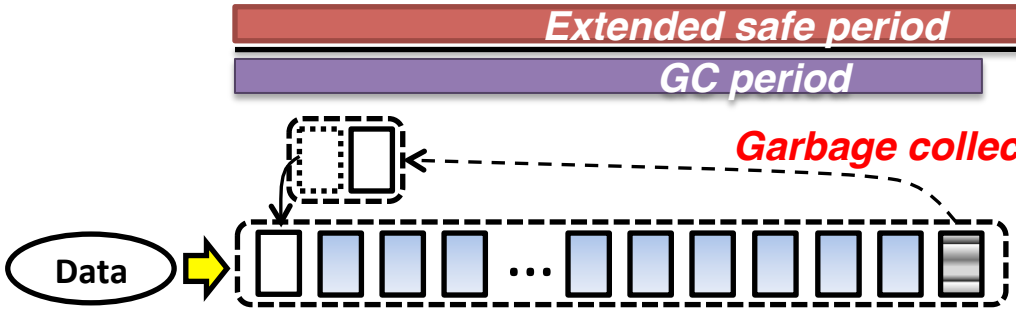
$$WAF_{scrub} = \frac{D_{write} + D_{scrub}}{D_{write}}$$

D_{write} : writes by host
D_{scrub} : writes by scrubbing

IR can be used to extend safe period

Extended Safe Period vs. GC Period

- **GC period < extended safe period**

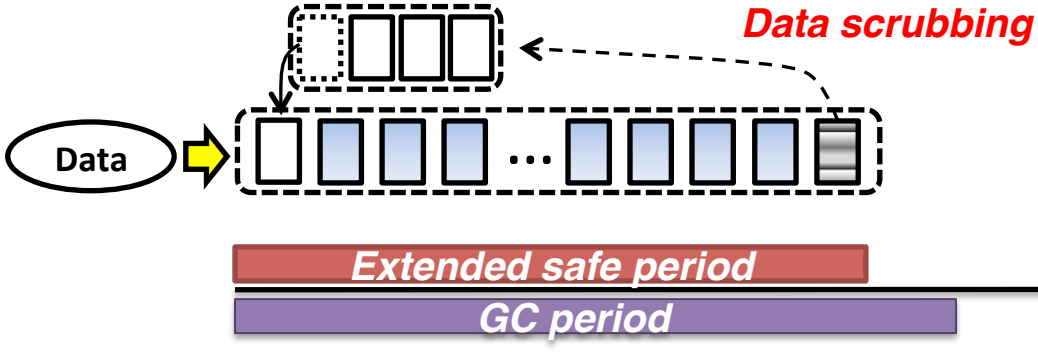


$$WAF_{GC} = \frac{1}{1-u}$$

u : utilization of block
considering space of parities

- **GC period > extended safe period**

$$WAF_{scrub} = \frac{D_{write} + D_{scrub} + D_{parity}}{D_{write}}$$

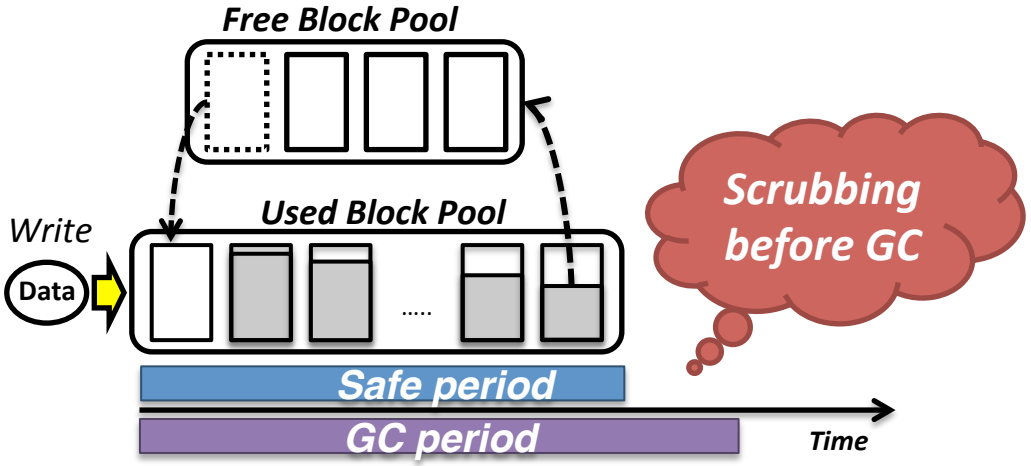


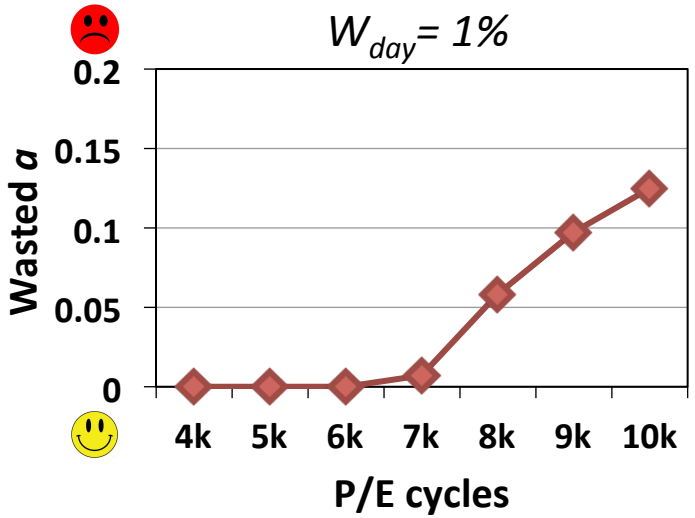
D_{write} : *writes by host*
D_{scrub} : *writes by scrubbing*
D_{parity} : *parity writes*

Outline

- Introduction & Approach
- Estimate Error and Period
- **Effective OPS size and Hot/Cold Separation**
- Experiments & Conclusion

Wasted OPS



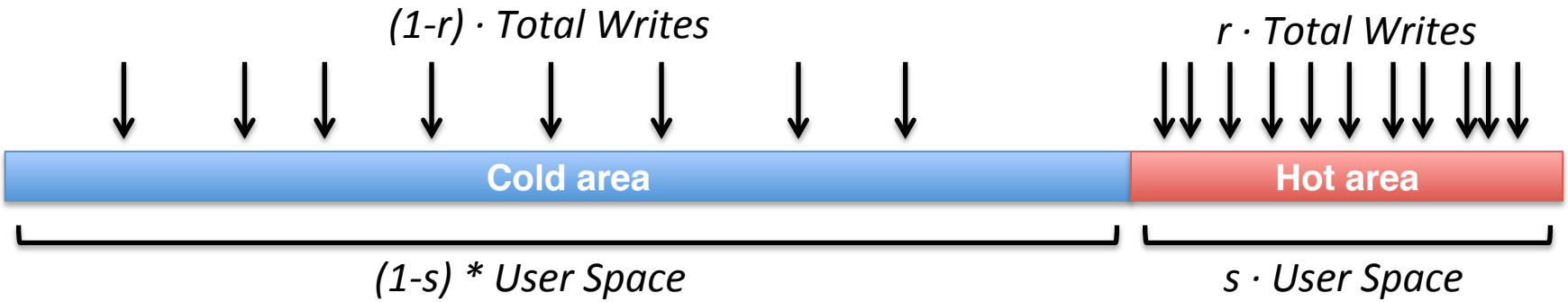


- OPS (**O**ver **P**rovisioning **S**pace)
  - The rest of total space except user space
- GC period > safe period (or extended safe period)
  - Data scrubbing generates free space before GC operation
  - OPS can not be fully used

$$a_{effective} = \frac{ln(u_{scrub})}{(1 - u_{scrub})} - 1$$

- *In case of Hot/Cold Separation, wasted OPS can be reassigned*

Configuration of Hot/Cold

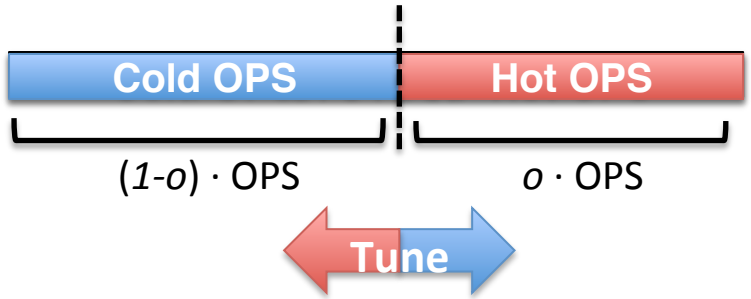


OPS needs to be separated into 2 parts for hot and cold

We initially set optimal *o* which is found by Desnoyers

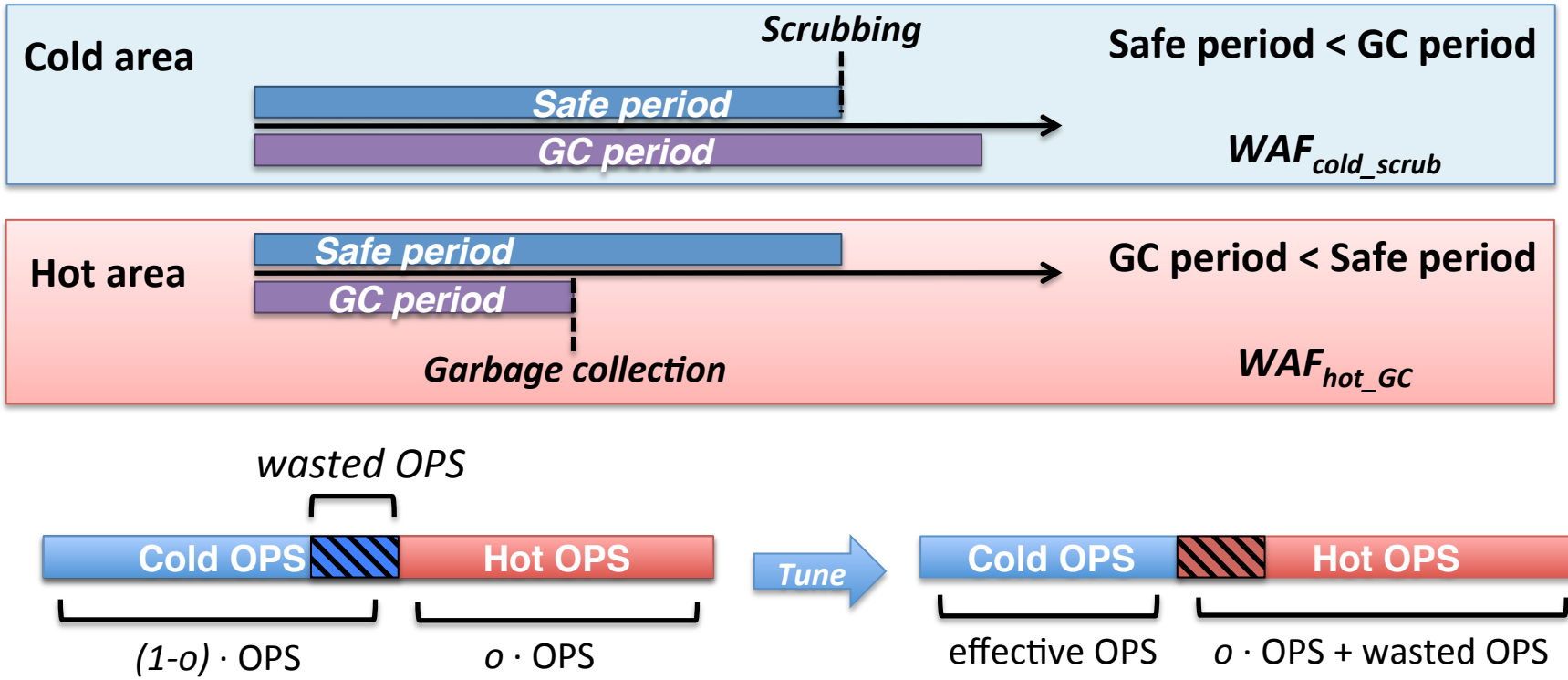
From: Desnoyers, "Analytic Models of SSD write Performance," ACM Transactions on Storage

$$WAF = r \times WAF_{hot\_GC} + (1 - r) \times WAF_{cold\_GC}$$



Then tune the *o* as effective ops size which is suited to safe period

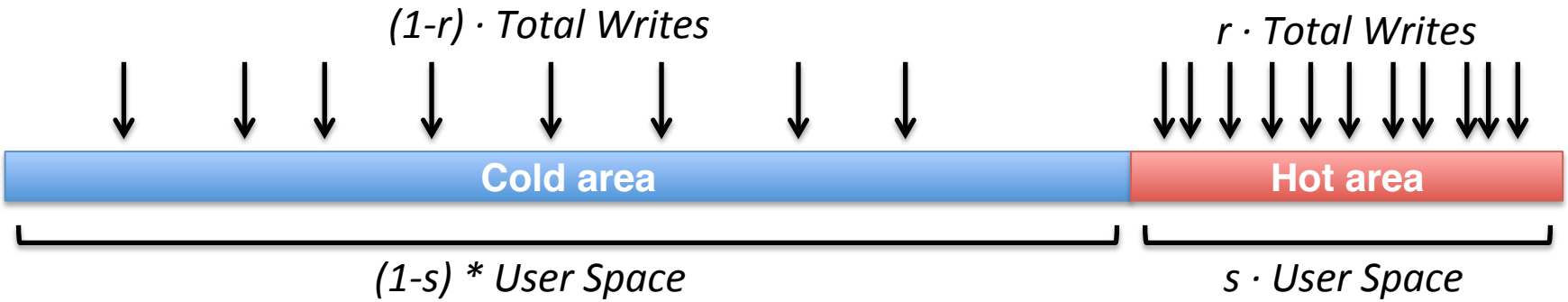
OPS Tuning



- **If scrubbing occurred one area**
  - Wasted OPS can reassigned the other area

$$WAF = r \times WAF_{hot\_GC}^{tune} + (1 - r) \times WAF_{cold\_scrub}$$

Definition of WAF



Hot	Cold	WAF
GC	GC	$WAF = r \times WAF_{hot\_GC} + (1-r) \times WAF_{cold\_GC}$
GC	Scrub	$WAF = r \times WAF^{tune}_{hot\_GC} + (1-r) \times WAF_{cold\_scrub}$
Scrub	GC	$WAF = r \times WAF_{hot\_scrub} + (1-r) \times WAF^{tune}_{cold\_GC}$
Scrub	Scrub	$WAF = r \times WAF_{hot\_scrub} + (1-r) \times WAF_{cold\_scrub}$

Outline

- Introduction & Approach
- Estimate Error and Period
- Effective OPS size and Hot/Cold Separation
- **Experiments & Conclusion**

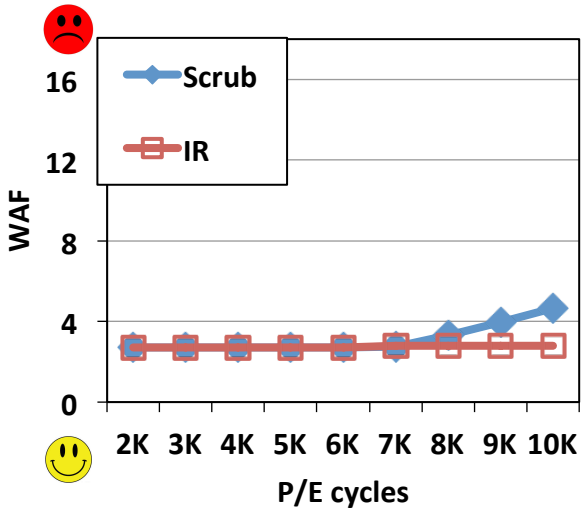
Experiment Setup

- **Page-mapping FTL simulator**
  - With **data scrubbing**

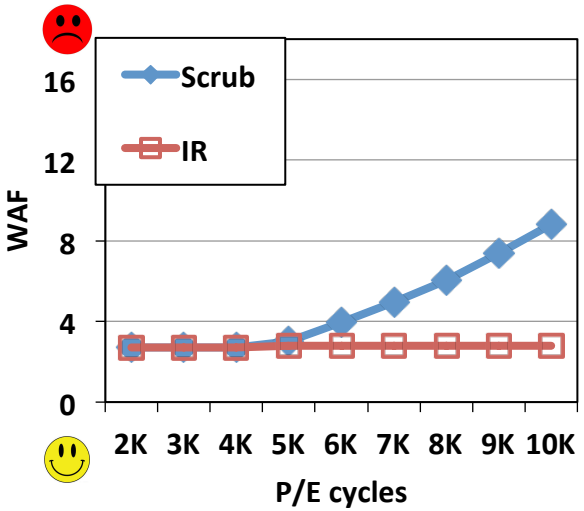
- **Workloads**
  - Synthetic workload (Random & Hot/Cold)
  - Real workload from *MSR* cambridge
  - W_{day} means the ratio of updated data per day in user data space.

- **Setting**
  - SSD capacity = 128GB, user space = 80% (initialized)
  - Page size = 4k
  - Stripe size = PPB (Page Per Block) = 128
  - ECC (correct 8 bits / detect 16 bits per codeword)
  - Page is composed of 8 codewords
  - GC policy = LRW (Least Recently Written)

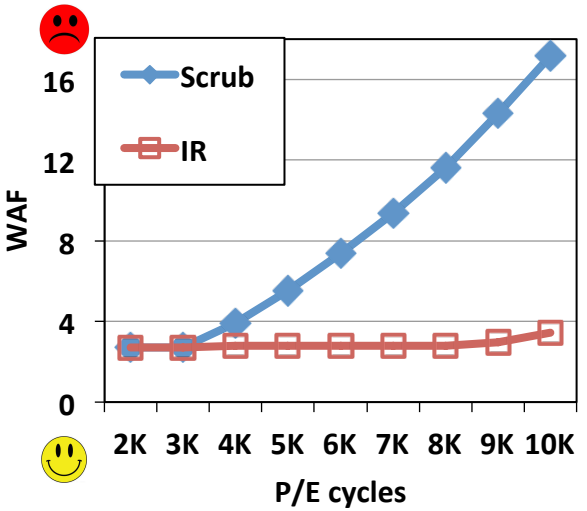
Random workload



(W_{day} = 1%)



(W_{day} = 0.5%)



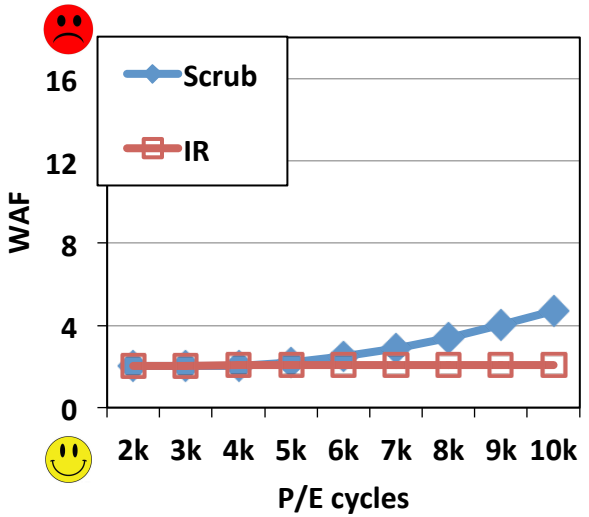
(W_{day} = 0.25%)

Scrub *: conventional scrubbing*
IR *: our proposal*

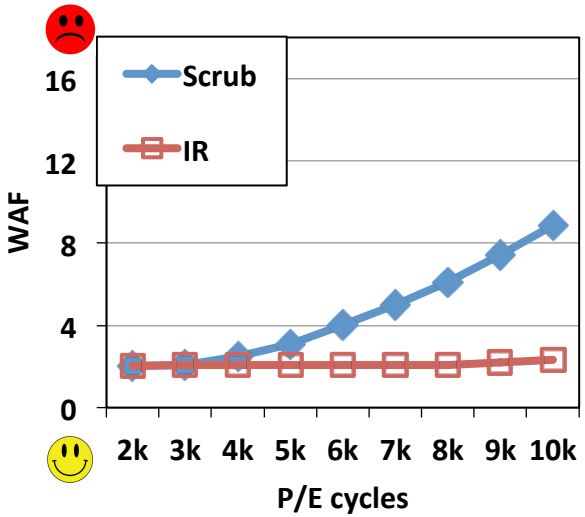
- As data are less updated
  - GC period is more longer
  - The safe period expires on lower P/E cycles (GC period > Safe period)
  - WAF increases on earlier stage

- As P/E cycles increases
  - WAF continuously increases (safe period becomes shorter)

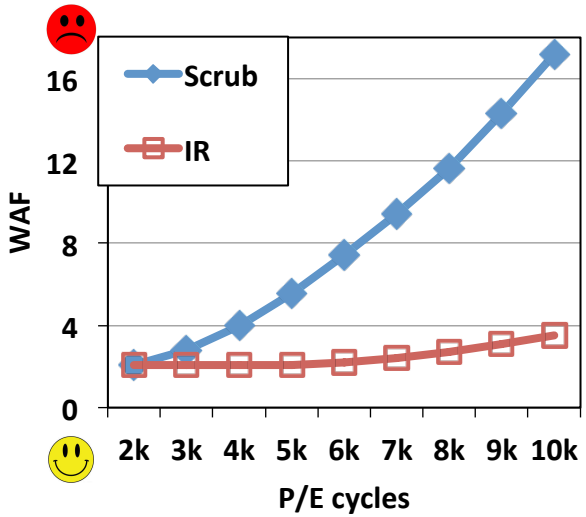
Hot/Cold Workload



(W_{day} = 1%)



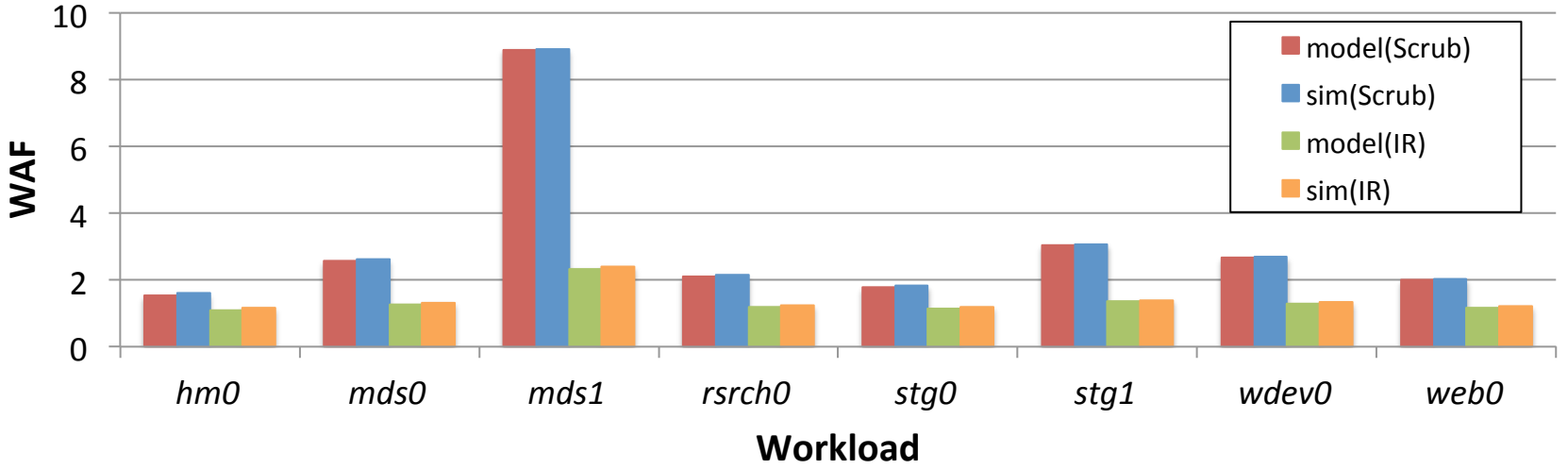
(W_{day} = 0.5%)



(W_{day} = 0.25%)

- Trends are similar to random workload
  - As W_{day} is small, GC period become longer
- WAF starts to soar for Scrubbing, but for IR
- IR can release additional writes by scrubbing

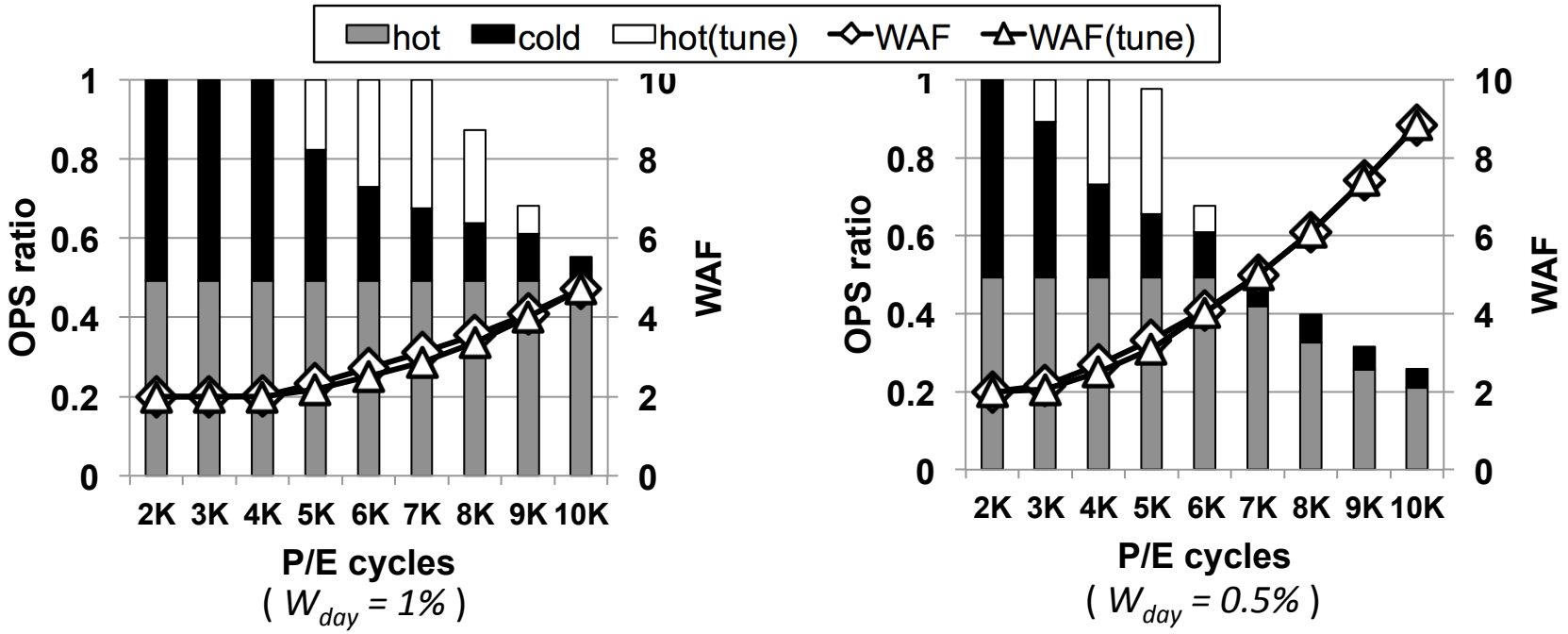
Real Workload



- Configuration
  - 6K P/E cycles
  - Derive parameters (*s, r,* W_{day})
- Results of simulation and model are similar
  - WAF decreases when IR is applied
- Extended safe period
  - Decrease additional writes by scrubbing
  - More opportunities to be refreshed by writes

Name	*s*	r	*Avg.* W_{day}
hm0	0.014	0.990	3.19%
mds0	0.002	0.986	1.09%
mds1	0.002	0.824	0.22%
rsrch0	0.002	0.991	1.54%
stg0	0.004	0.991	2.20%
stg1	0.002	0.979	0.84%
wdev0	0.003	0.991	1.03%
web0	0.006	0.992	1.70%

Effect of OPS tuning



- Scrubbing occurs in cold area optimally assigned OPS wasted
- WAF can be lowered by reassigning wasted OPS
  - Decreases WAF of hot area
- When the scrubbing also occurs in hot area
  - Total OPS cannot fully used

Conclusion

- ***Retention error*** is a serious issue and must be handled

- ***IR*** writes parity dynamically to
  - ***Reinforce*** data correct capability
  - ***Extend*** the safe period of data

- ***IR*** can be a promising solution
  - With ***little overhead*** for striping
  - Data scrubbing operation ***can be delayed***
  - Additional writes by scrubbing ***can be reduced***

Thank you