Information in Storage Devices

049063 – EE Department, Technion

LECTURE 5: WEAR LEVELING AND PERFORMANCE

Summary

- WA reduces with amount of spare
- LRU GC: simple
- Greedy GC: optimal
- Greedy approaches LRU with large N_p
- Another advantage of LRU:

Wear leveling

The Fundamental Property of Flash Storage

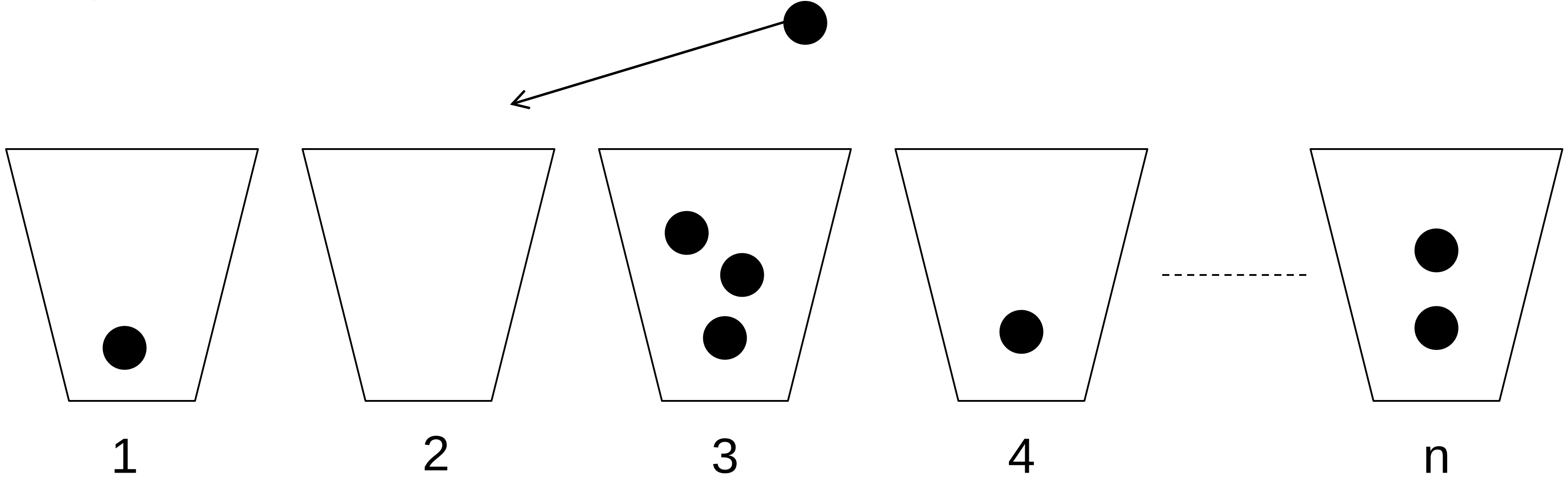
Theorem:

For a device with #E_units=n, after a workload of nt uniform writes, the probability that there exists an E_unit with more than log n dirty P_units tends to 0 as n →∞.

Balls and Bins:

n bins, M balls

1/n: probability of ball falling in bin i



Balls-and-Bins Max Occupancy

Implication:

- No block with “very many” dirty pages.
- t on average, log n max.
- Greedy not fundamentally better than random choice

Proof:

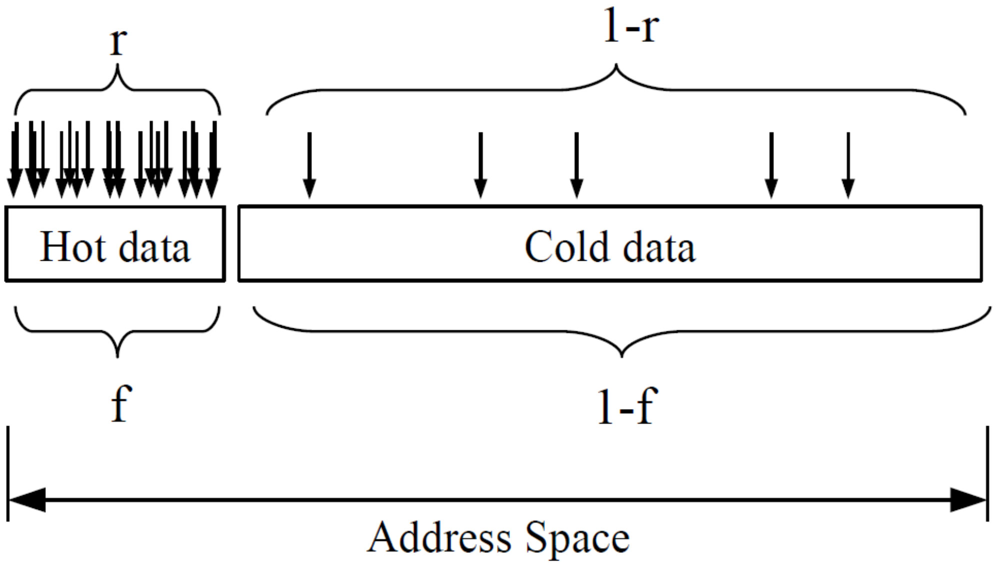
Pr[M balls or more in bin i] < ?

Pr[M balls or more in any bin] < ?

Pr[log n balls or more in any bin] → 0

Beyond Uniform Workloads

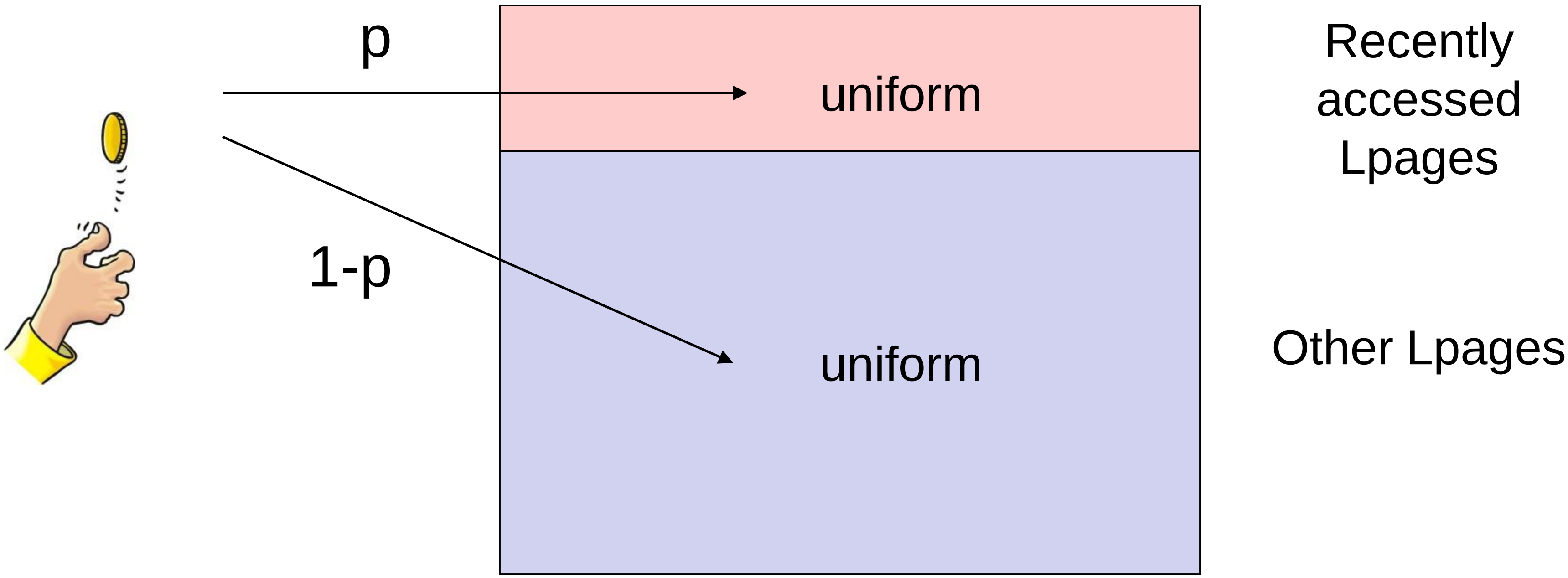
1. Hot/Cold logical addresses



- Can separate hot and cold to two independent mapping layers with same over-provisioning factor
- Same A as with uniform
- Can do better. How?

p-Local Workloads

2. Time-locality with parameter p



p

uniform

Recently accessed Lpages

1-p

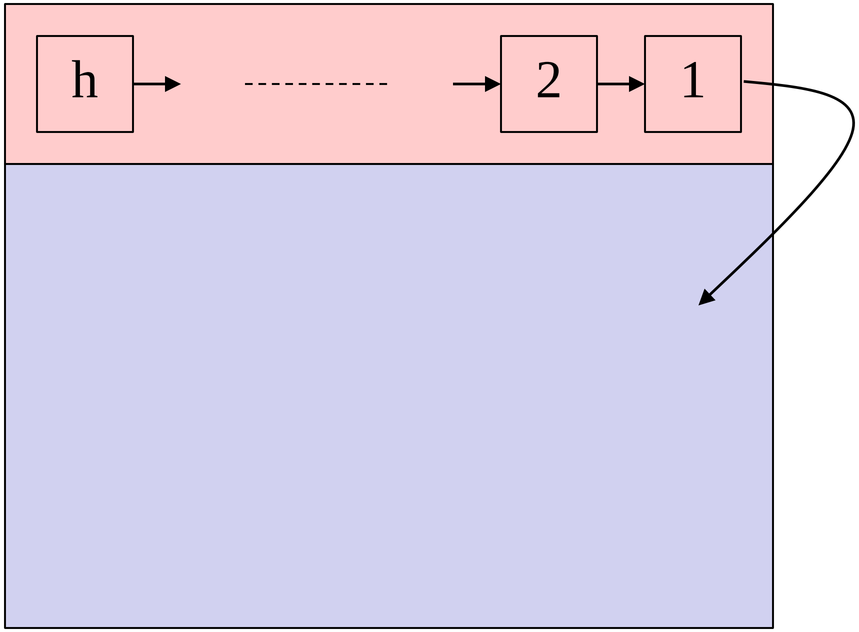
uniform

Other Lpages

p-Local Workloads

2. Aging parameter h

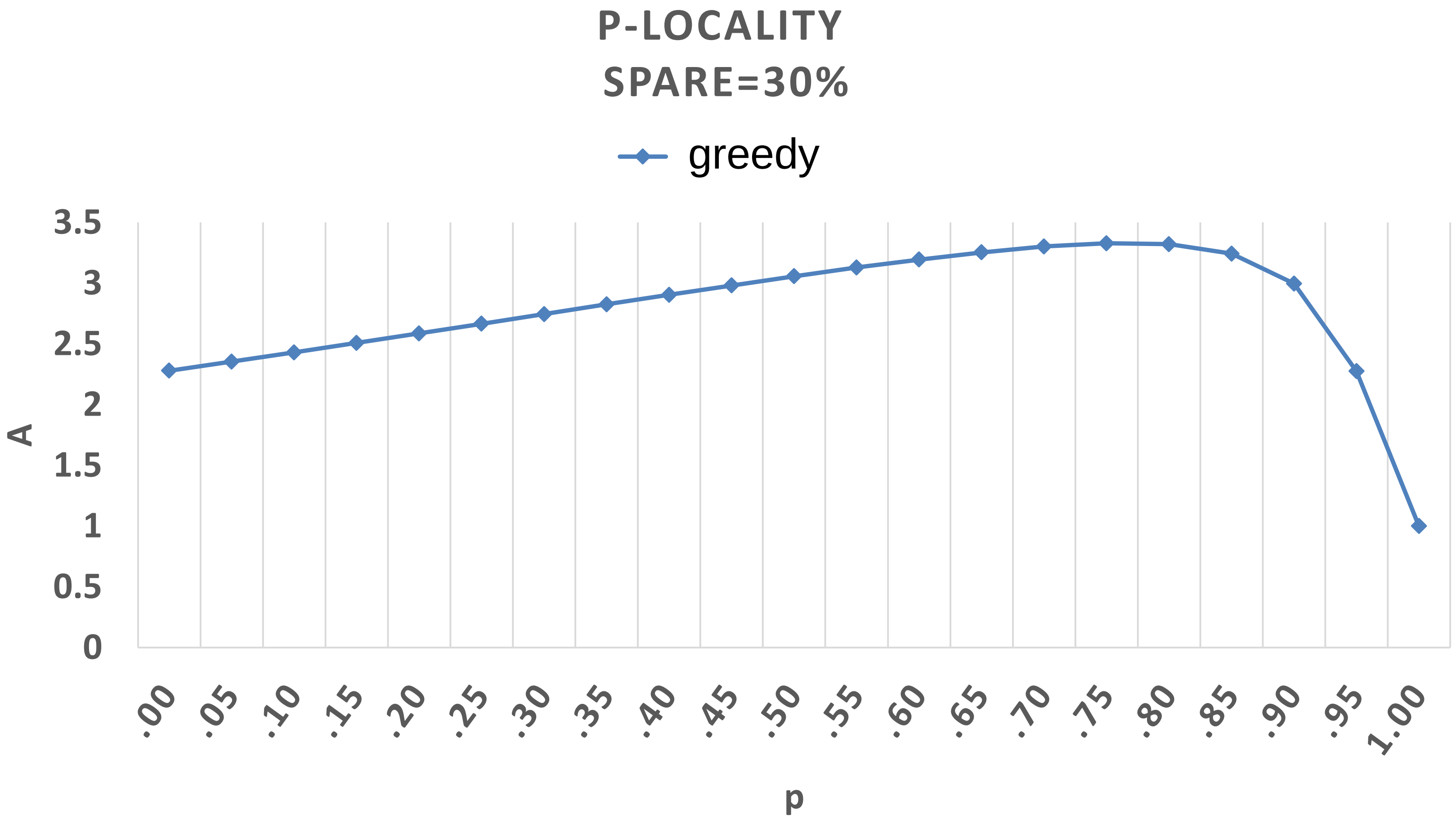
User writes:
1,2,…,h,h+1,…



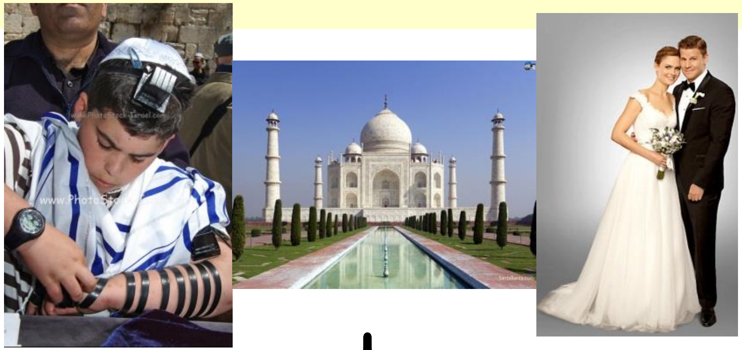
Recently accessed Lpages

Other Lpages

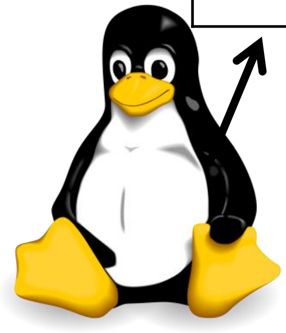
Write Amplification with p-Locality



Wear Leveling

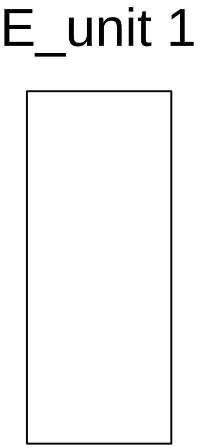


0	4	8	12
1	5	9	13
2	6	10	14
3	7	11	15



Erase Count

N_p=1, T=U=1



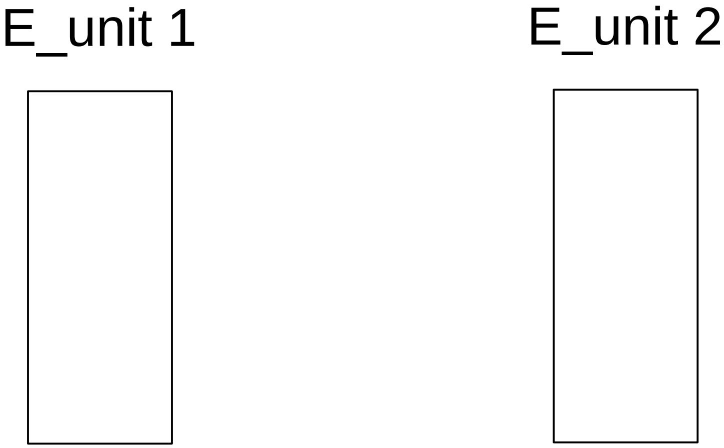
E_unit erase count limit: K

Total number of writes before end of life = 1 + K

$$\underbrace{H\ H\ H\ \ldots\ H}_{\text{K updates}}$$

Uneven Wear

$N_p=1$, T=U=2



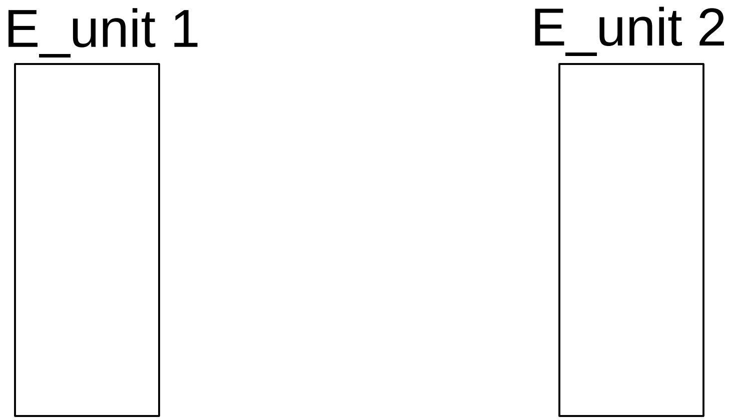
E_unit erase count limit: K

Total number of writes before end of life = 2 + K

C1 $\underbrace{\text{H2 H2 H2 … H2}}_{\text{K times}}$ → used only K erases out of the total 2K

Leveling the Wear

$N_p=1$, T=U=2



Can we get to 2K+2 writes? **Almost…**

Example logical write sequence:

$$C\ \underbrace{H\ H\ H\ \dots\ H}_{2K\ \text{times}}$$

Physical write sequence:

$$C1\ \underbrace{H2\ H2\ H2\ \dots\ H2}_{K-1}\ \underset{\substack{\uparrow \\ \text{copy}}}{\boxed{C2}}\ \underbrace{H1\ H1\ H1\ \dots\ H1}_{K} \longrightarrow \text{total writes} = 2K+1$$

Wear-Leveling Algorithm

Theorem:

For T=U=n, N_p=1, any sequence of nK write requests can be fulfilled.

Algorithm:

(1) Copy logical unit L into E_unit i if

wear(i) + **remaining_writes**(L) = K-1

Proof idea:

(*) Every L is copied at most once. → overhead up to n
(*) All unused wear can be claimed by (1) operations

On-Line Wear Leveling

- Previous algorithm required knowledge of full access sequence (off line).
- What if wear-leveling is required for on-line accesses?

On-Line Wear-Leveling Algorithm

Algorithm OL1:

E1	E2	E3	···	En
L1	L2	L3	…	Ln

(1) Assign Li to Ei, for all i
(2) Li request → write in Ei

Trivial algorithm! guarantees only K updates for any n.

Theorem: Algorithm OL1 is optimal ☹

Proof:

Host can always issue a request to argmax[wear(Ei)]

On-line wear leveling **not** possible without over-provisioning

On-Line + Over-Provisioning

Algorithm OL2:

E1	E2	E3	⋯	Em	Em+1	⋯	En
L1	L2	L3	…	Lm			

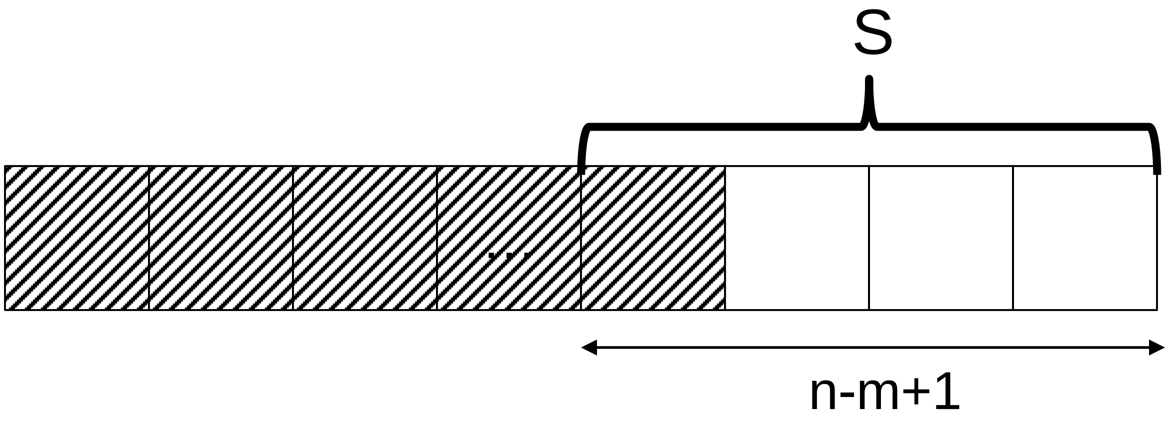
(1) Assign Li to Ei, for all i
(2) Li request → write to unused Ej with lowest wear

Algorithm OL2 guarantees $(n-m+1)K$ updates
(OL1 special case n=m)

Theorem: Algorithm OL2 is optimal

Optimality Proof

Proof:



Adversarial host:

(1) Fix a set of S=n-m+1 physical E_units

(2) Issue write requests only to Li’s in S

Idea: There is always at least one Li allocated in S, hence $\sum_{j \in S} wear(E_j)$ grows by at least 1 every write

↓

Total updates at most |S|K=(n-m+1)K

Performance of Algorithms

Let A be an algorithm with inputs taken from χ. Denote the **performance** of A on input $x \in \chi$ by

Q[A(x)]

Example: wear lifetime

$$Q[A(x)] = \max[t: \forall j \in \{1, \ldots, n\}\; wear_t(j) \leq K]$$

$$x = [Req_1, Req_2, \ldots, Req_t, \ldots]$$

Worst-Case Performance

1) Worst-case performance

$$Q_{wc}[A] = \ min_{x \in \chi} Q[A(x)]$$

Every $x \in \chi$ gives $Q[A(x)] \geq Q_{wc}[A]$.

Average-Case Performance

2) Average-case performance

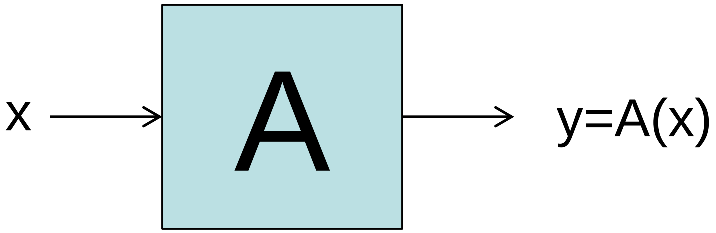
Let $x \sim D(\chi)$, *D* is a probability distribution on the inputs.

$$Q_{av}[A] = E_D[Q[A(x)]]$$

Often $D = U$ is the uniform distribution.

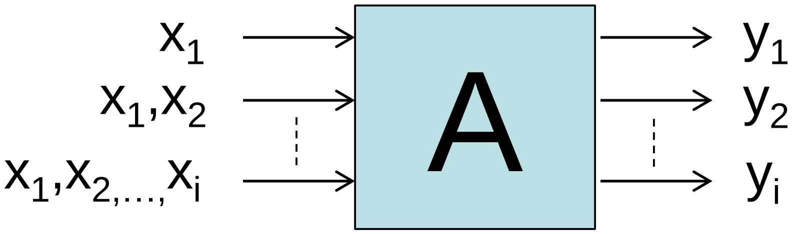
Offline vs. Online

1) Offline performance



A emits outputs only when the full x is given.

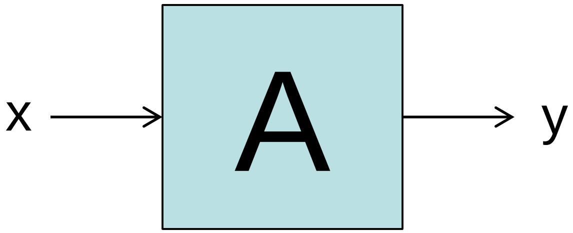
1) Online performance



A emits outputs / commits actions after every x_i.

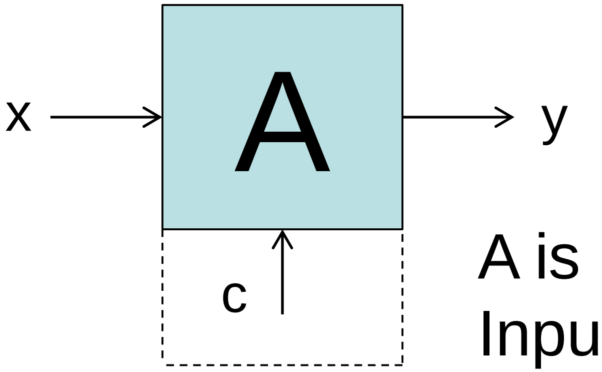
Deterministic vs. Probabilistic

1) Deterministic A



A is known to all.

1) Probabilistic A (also called randomized)

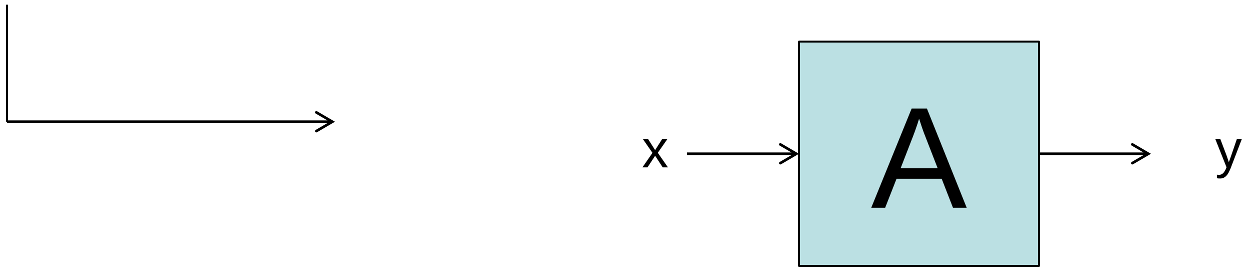


A is known to all.
Input c (random coin flips)
is private.

Adversarial Performance

1) Adversarial input

B knows A and chooses x



$$Q_{adv}[A] = min_{x=B(A)} Q[A(x)]$$

If A is deterministic: $Q_{adv}[A] = Q_{wc}[A]$

If A is randomized: $Q_{adv}[A] \geq Q_{wc}[A]$

$$Q_{adv}[A] = \ min_{x \in \chi} E_C Q[A(x, c)] \qquad Q_{wc}[A] = \ min_{\substack{x \in \chi \\ c \in C}} Q[A(x, c)]$$

Deterministic vs. Randomized Wear Leveling

N_p=1, T=U=N, wear_limit=K

Alg. Det.

Write in **fixed** E_unit

1 erase per L_write

Alg. Rand.

Write in **random** E_unit

2 erases per L_write (with probability 1-1/U)

Worst case	**K**	=	**K**
Adversarial	**K**	<	#ball-pairs, s.t. ≤K in all bins
Sequential	**NK**	>	**NK/2**

Performance Evaluation

#balls s.t. ≤K in all bins

Q:

Given n bins and bound K on number of balls,
how many balls m can throw?

A:

Poisson approximation:

$$lim_{n\to\infty} \Pr(X_n = k) = p(k) \triangleq \frac{e^{-t}t^k}{k!}$$

$$0.001 > \sum_{k=K+1}^{\infty} p(k)$$

What is t?

$t = \frac{m}{n}$ (expectation). Find t and substitute $m = tn$.

Other Topics in Data Placement

- Caching/pre-fetching
  - Move data between fast and slow media to maximize R/W performance
- Compression and data reduction
  - Can improve access and wear performance
  - Challenges to mapping layer
- Workload detection and prediction
  - Tailor placement to workload features
- Security and access control
  - Data privacy, access privileges