

Indexing Tutorial

Database System Technology

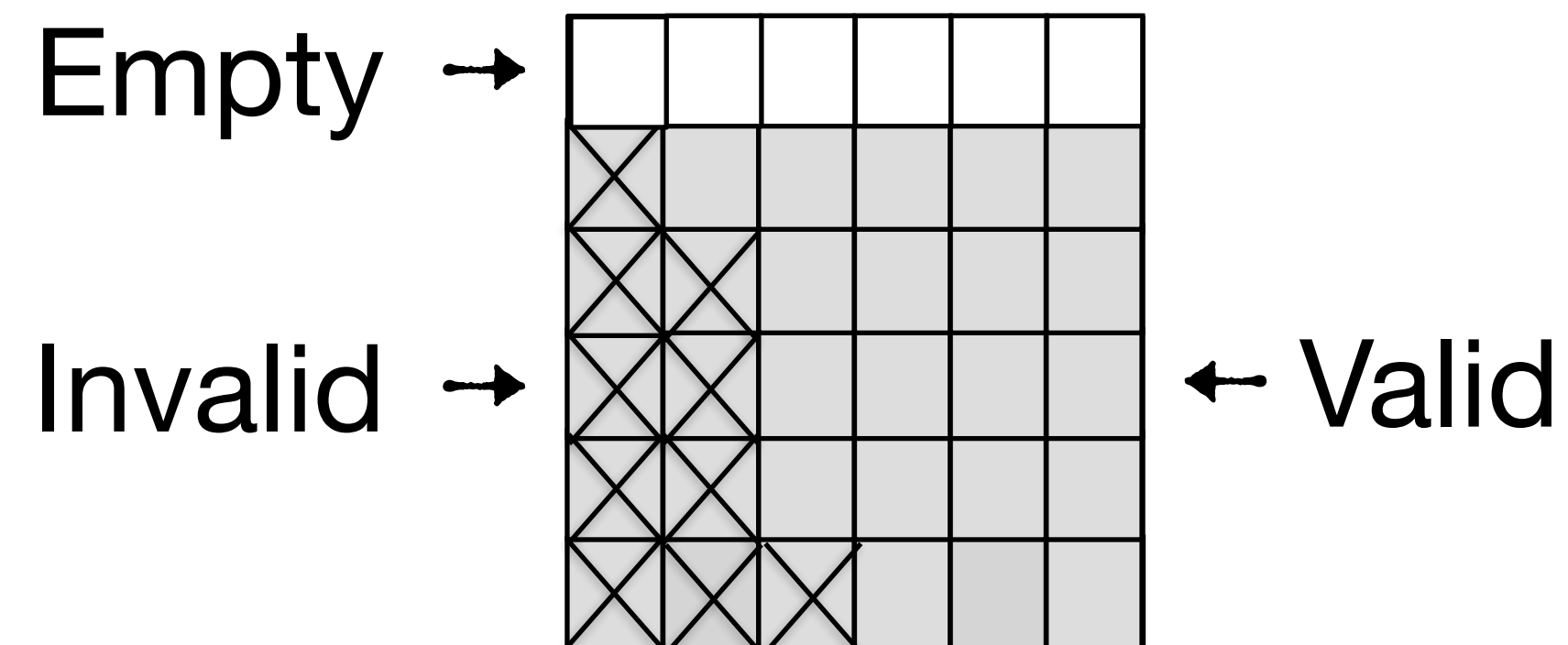
Niv Dayan

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

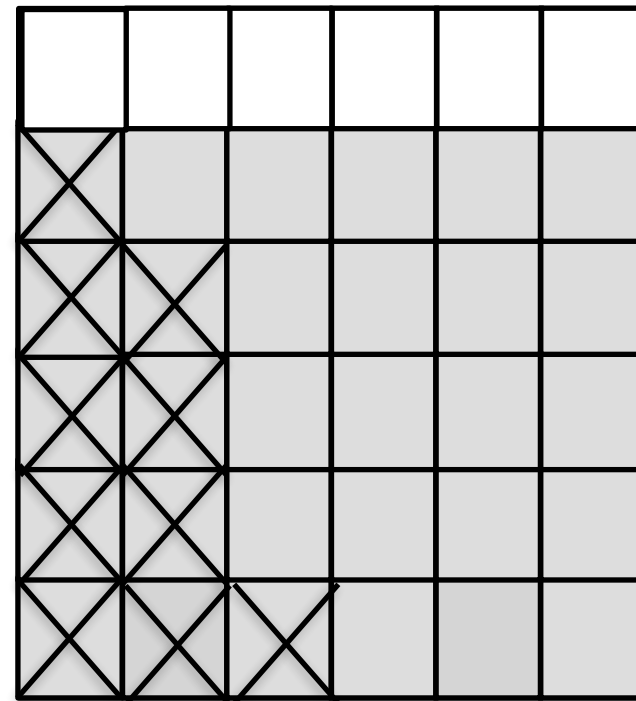
Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P .
The usable capacity is a fraction of L/P of the overall capacity.



Modeling SSD Write-Amp

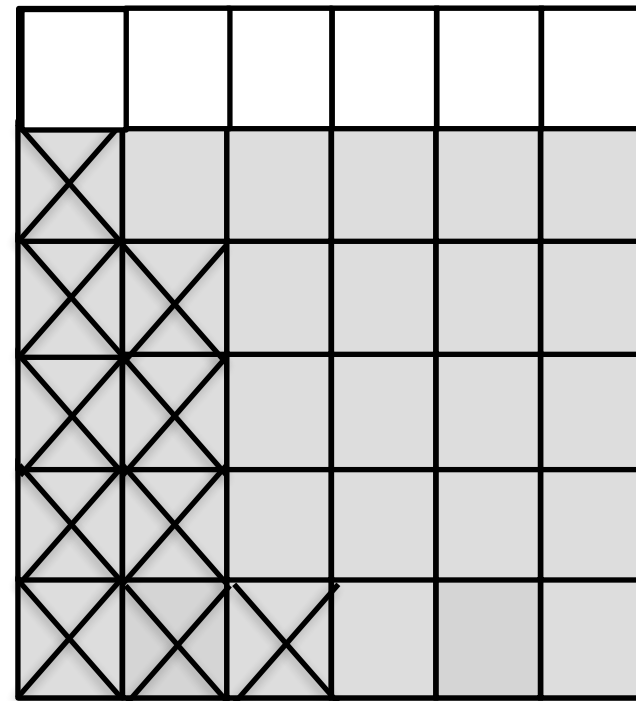
Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.



$$L/P = \frac{\%Valid}{\%Empty + \%Valid + \%Invalid}$$

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.



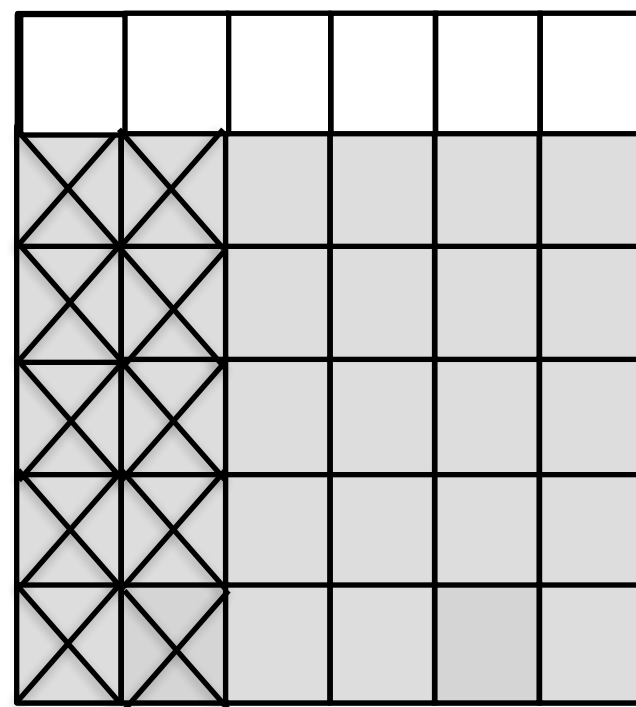
$$L/P = \frac{\%Valid}{\%Valid + \%Invalid}$$

Modeling SSD Write-Amp

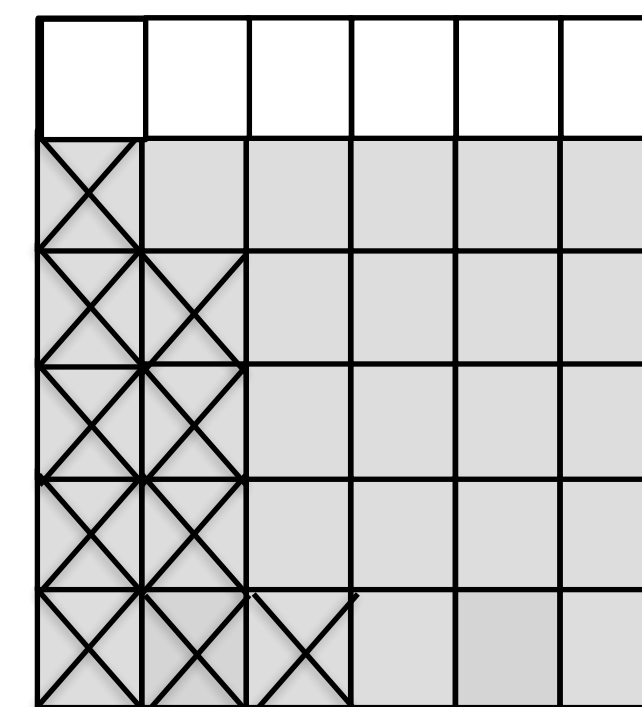
Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.

Worst case



Non-Worst Case



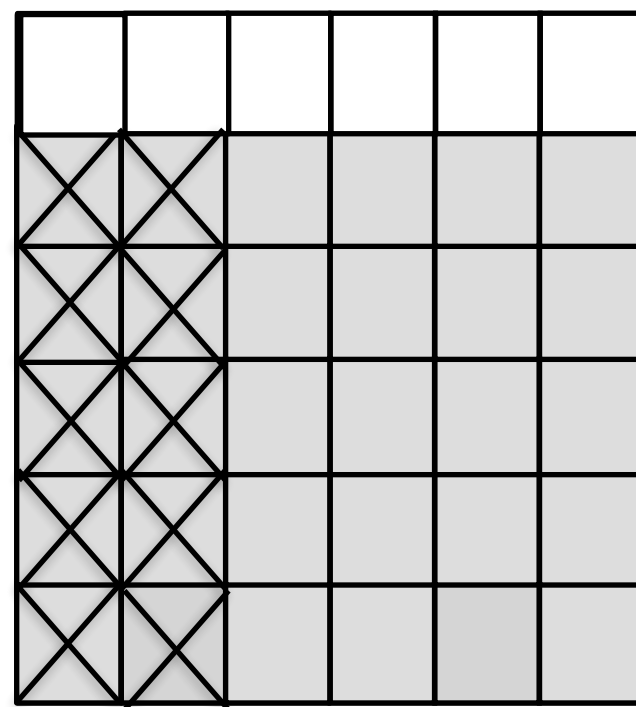
← Best target

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.

Worst case



SSD WA Approx

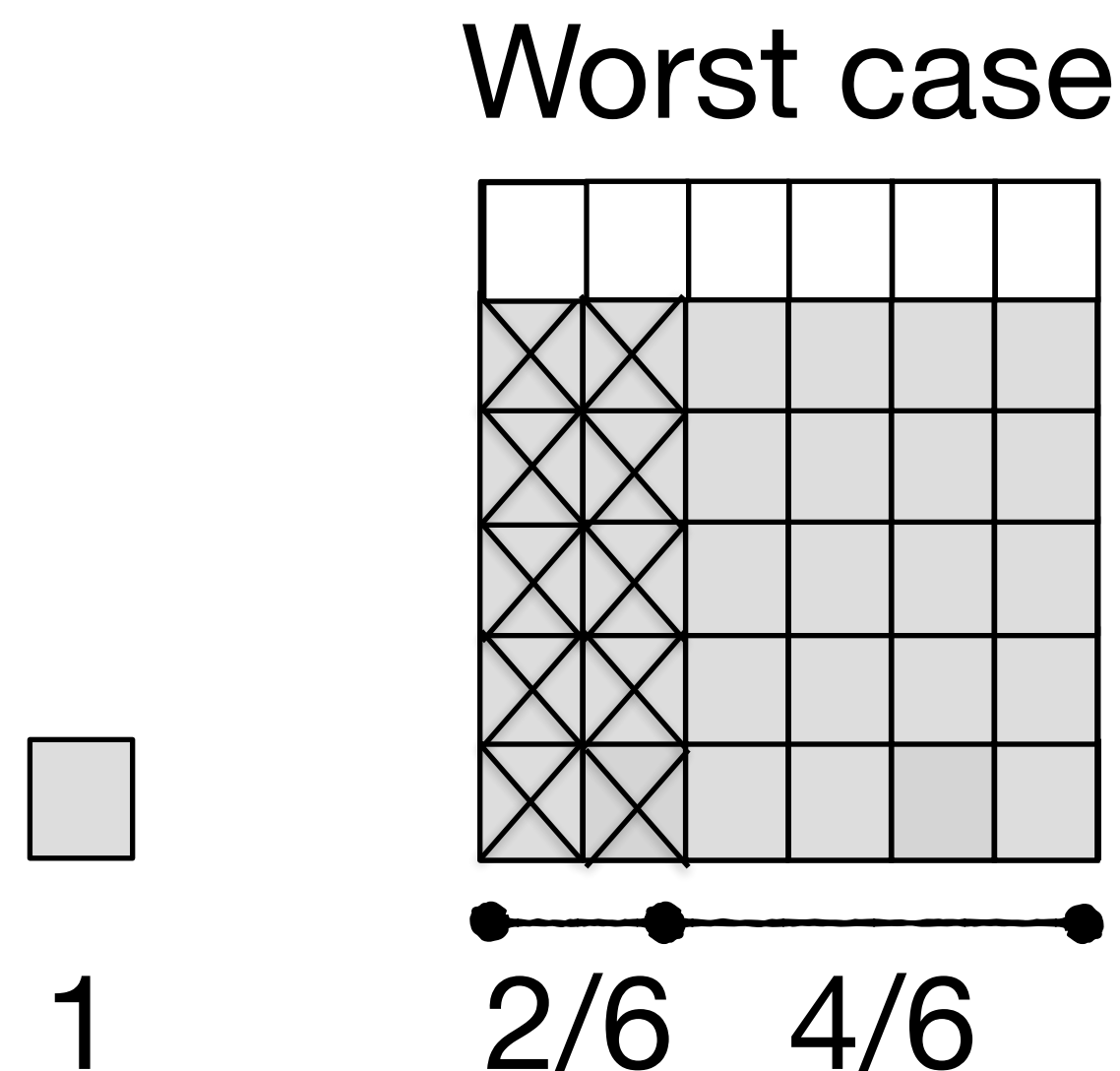
$$1 + \frac{x}{1 - x}$$

x is avg. fraction of valid pages in the garbage-collected block

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.



SSD WA Approx

$$1 + \frac{x}{1 - x}$$

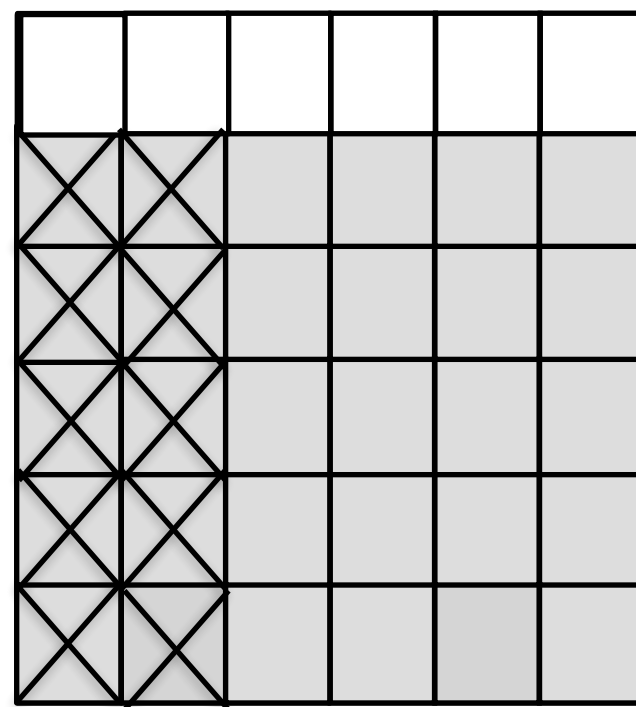
x is avg. fraction of valid pages in the garbage-collected block

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.

Worst case



$$1 + \frac{4/6}{2/6}$$

SSD WA Approx

$$1 + \frac{x}{1 - x}$$

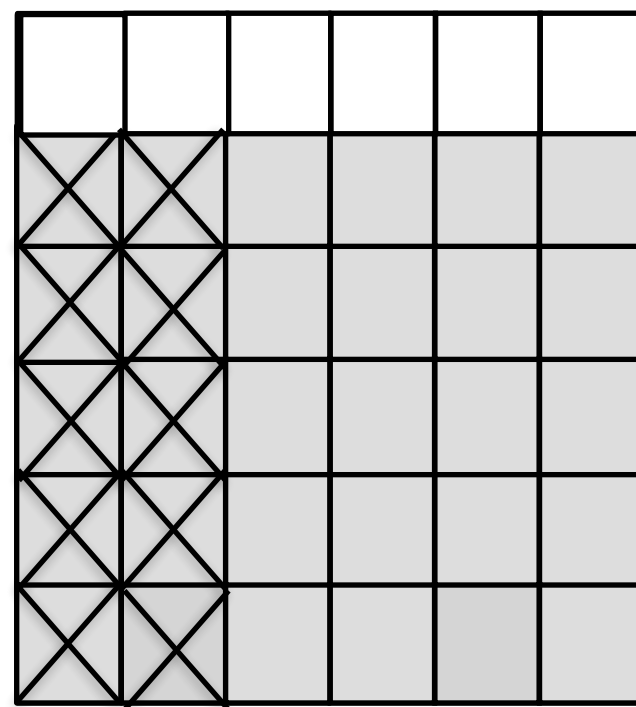
x is avg. fraction of valid pages in the garbage-collected block

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.

Worst case



$$1 + \frac{4}{2} = 3$$

SSD WA Approx

$$1 + \frac{x}{1 - x}$$

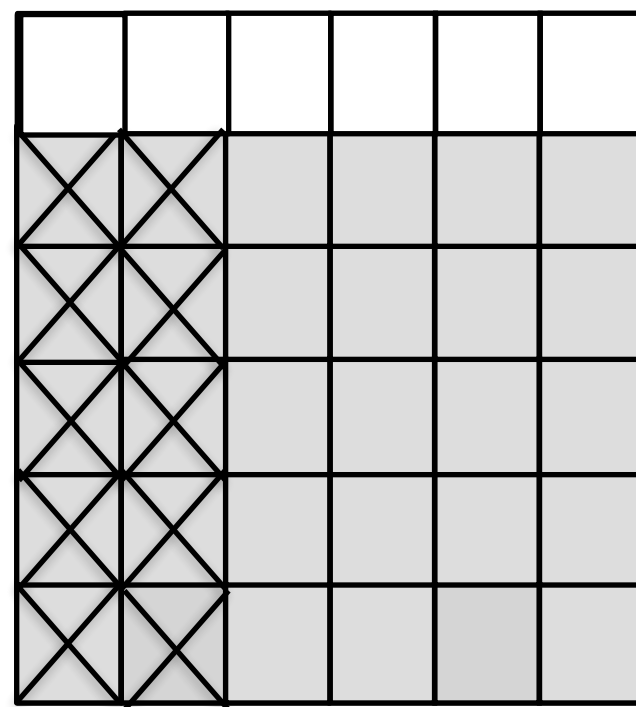
x is avg. fraction of valid pages in the garbage-collected block

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.

Worst case



SSD WA Approx

$$1 + \frac{x}{1 - x}$$

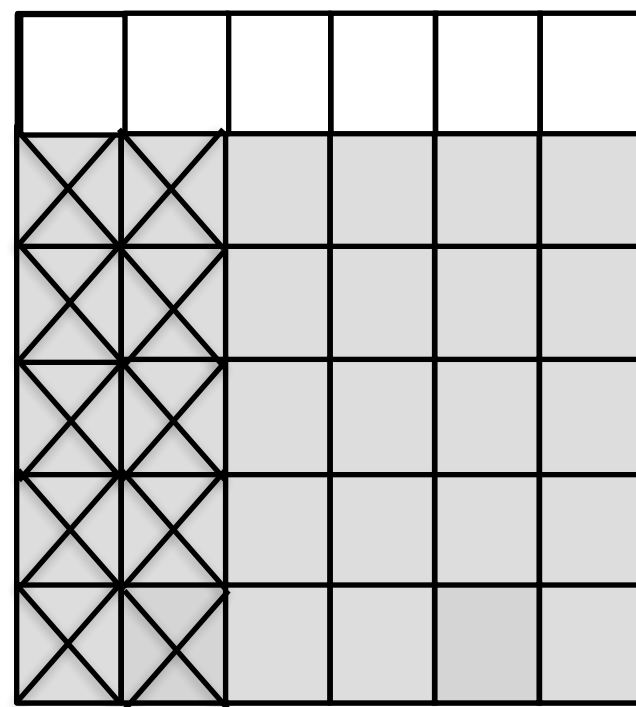
In worst case: $x = \text{valid} / \text{total} = 4/6 = L/P$

Modeling SSD Write-Amp

Consider an SSD with logical address space size L and physical capacity P . The usable capacity is a fraction of L/P of the overall capacity.

Worst case write-amplification (WA) occurs when each erase-unit has same number of invalid pages.

Worst case



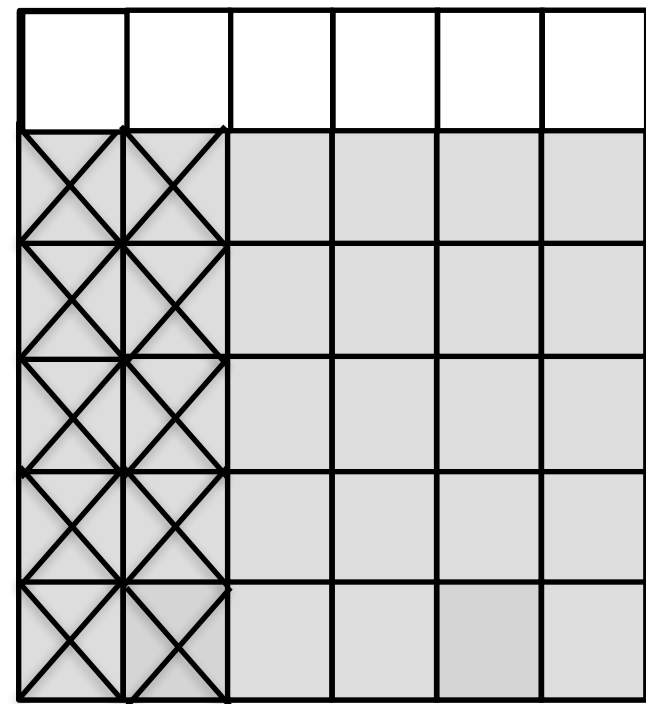
SSD WA Approx

$$1 + \frac{L/P}{1 - L/P}$$

In worst case: $x = \text{valid} / \text{total} = 4/6 = L/P$

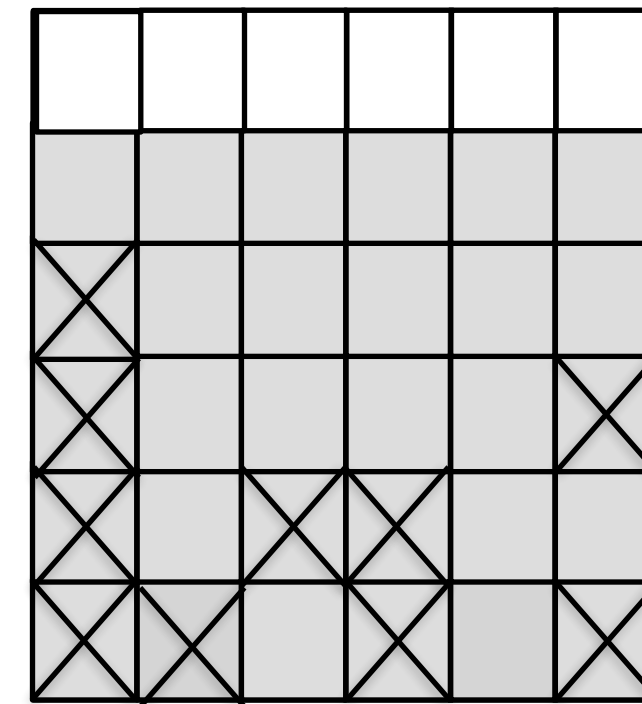
Modeling SSD Write-Amp

But the worst-case hardly
ever happens in practice



$$1 + \frac{L/P}{1 - L/P}$$

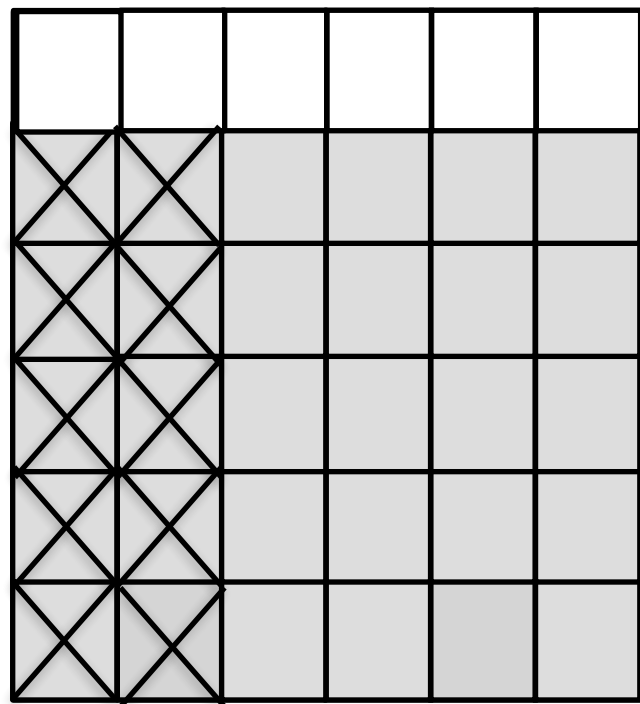
In practice, erase units written
longer ago have more invalid
pages, so GC is cheaper



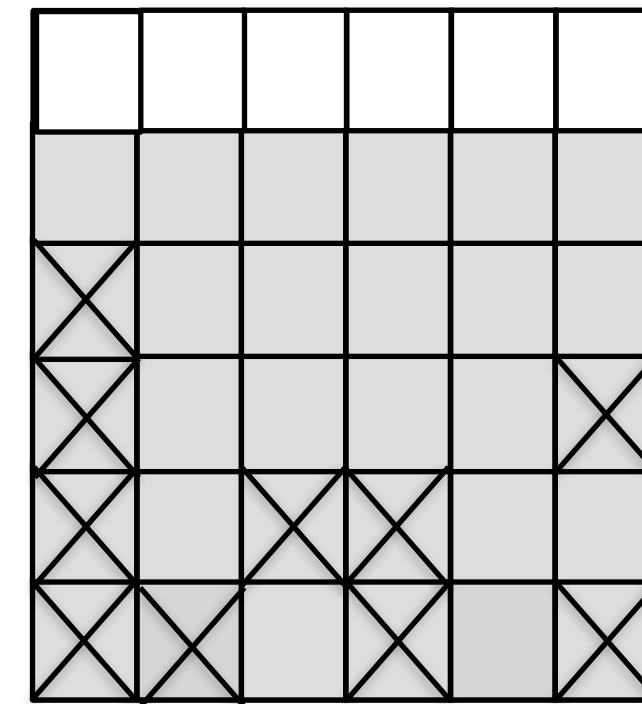
Older

Modeling SSD Write-Amp

Cost model assuming uniformly
randomly distributed writes?



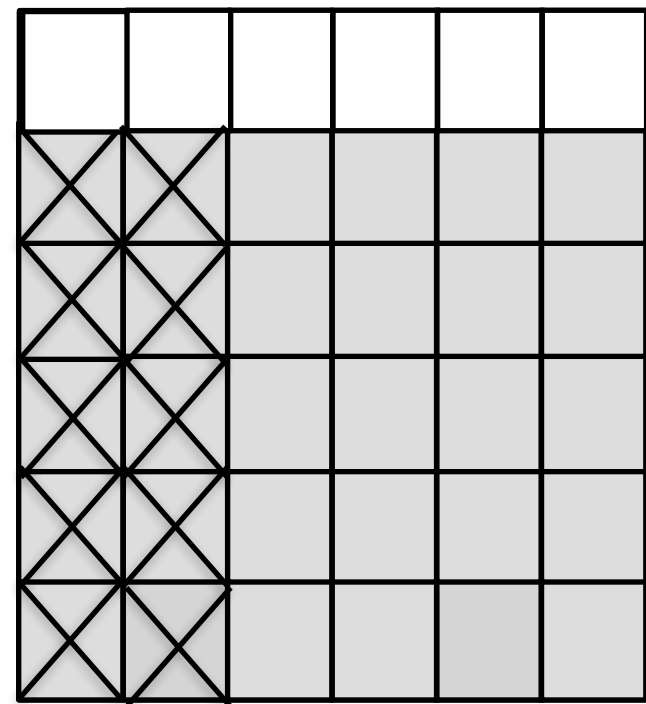
$$1 + \frac{L/P}{1 - L/P}$$



Older

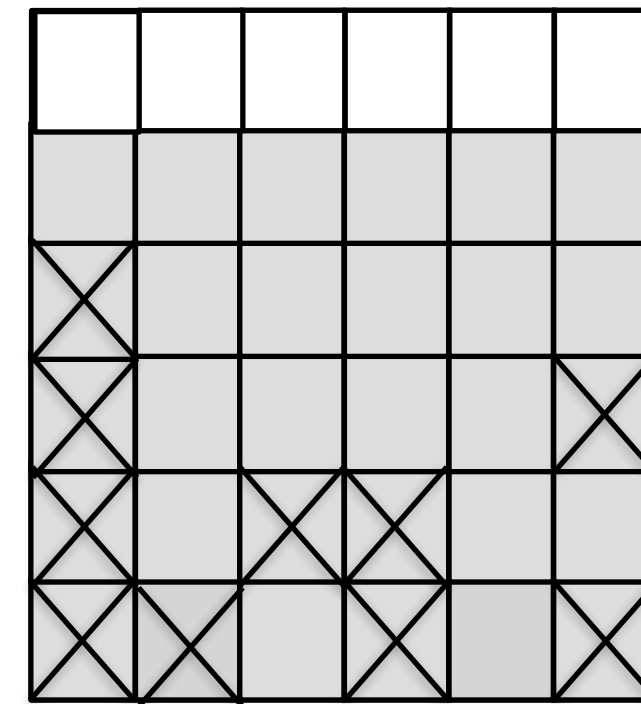
Modeling SSD Write-Amp

Cost model assuming uniformly
randomly distributed writes?



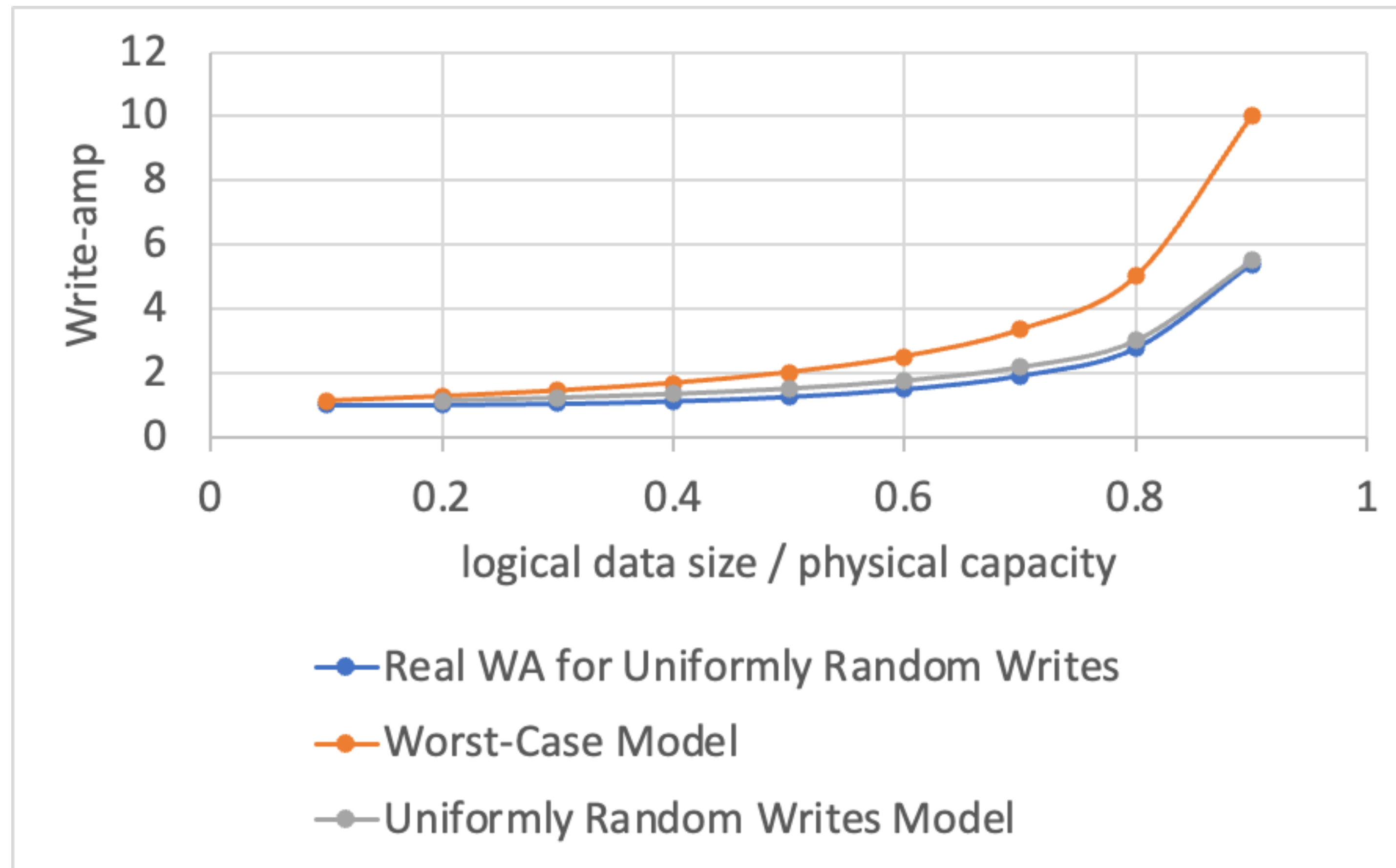
$$1 + \frac{L/P}{1 - L/P}$$

$$1 + \frac{1}{2} \cdot \frac{L/P}{1 - L/P}$$



Older
↓

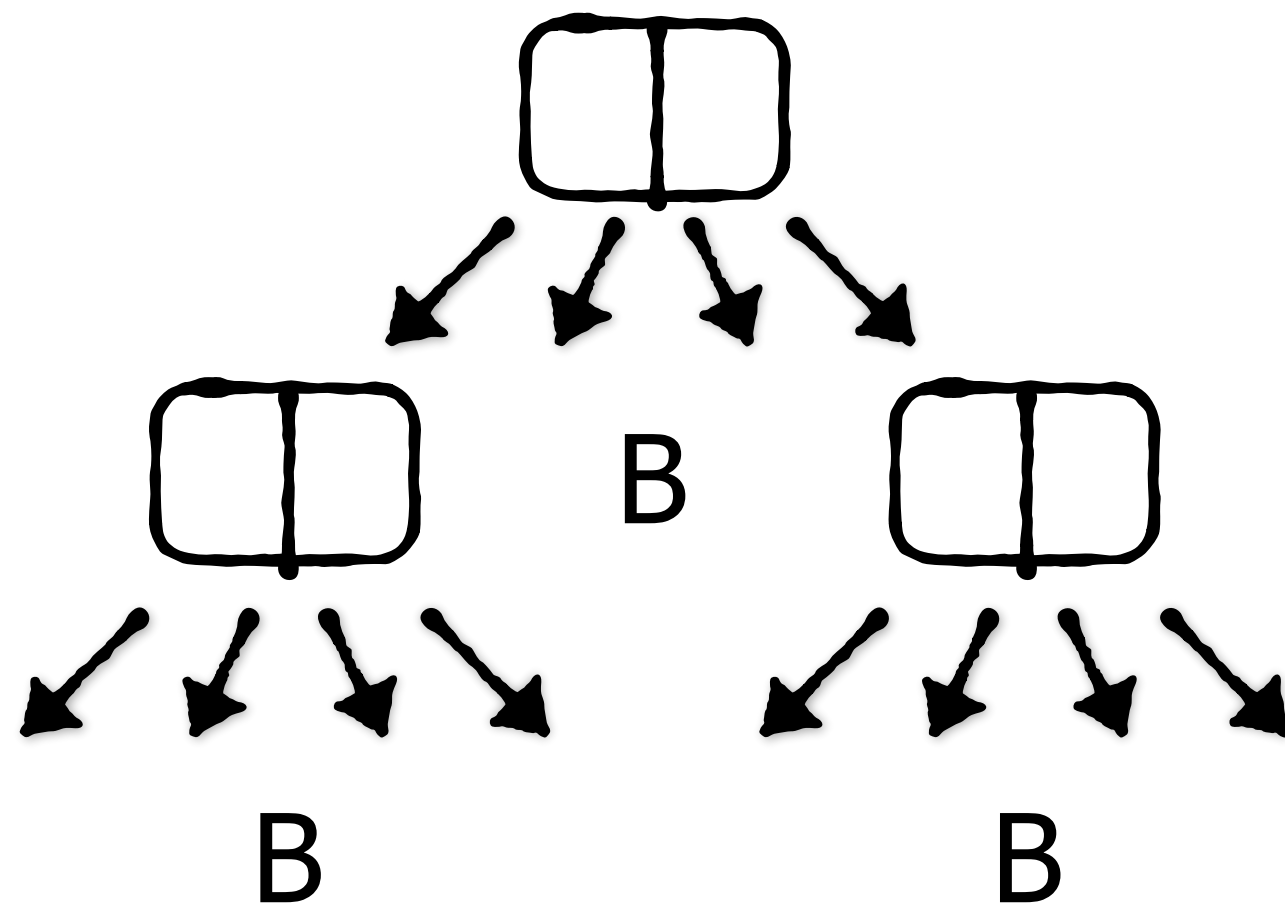
Modeling SSD Write-Amp



$$1 + \frac{L/P}{1 - L/P}$$
$$1 + \frac{1}{2} \cdot \frac{L/P}{1 - L/P}$$

Question 1

Consider a B-tree subject to uniformly randomly distributed updates. There are 100 entries per page. The b-tree occupies 70% of the physical SSD capacity, while the rest is over-provisioned. What write-amplification would you expect? *A back-of-the-envelope calculation is enough.*



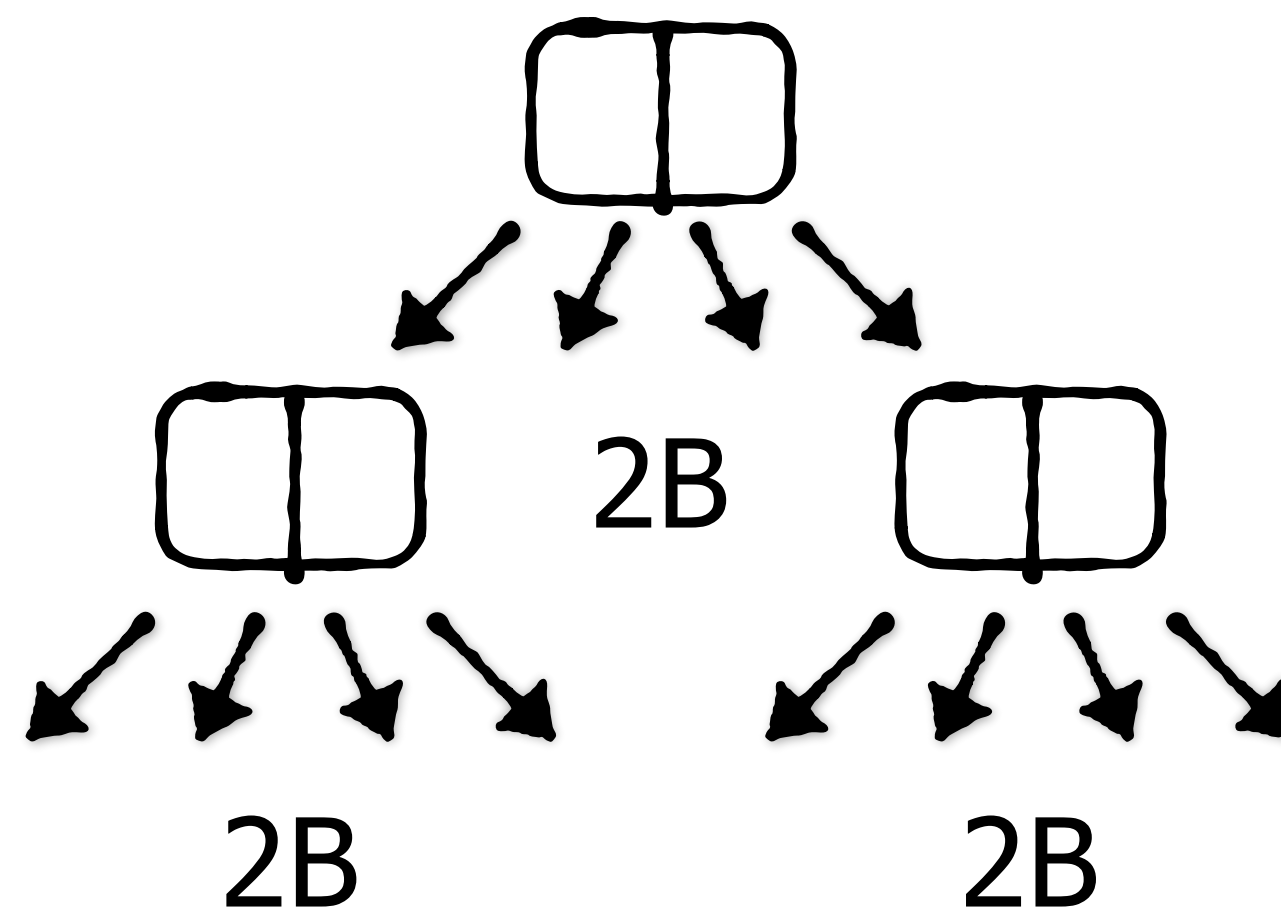
Question 1

Consider a B-tree subject to uniformly randomly distributed updates. There are 100 entries per page. The b-tree occupies 70% of the physical SSD capacity, while the rest is over-provisioned. What write-amplification would you expect? *A back-of-the-envelope calculation is enough.*

Now suppose the workload exhibits skew (some entries are more likely to be updated). Which mechanism of a database allows us to reduce write-amplification in this case, and why?

Question 2

We have a B-tree on an SSD. the workload is read-intensive, so we only care about read performance. Each node and each SSD page are 4 KB. Consider making each B-tree node take up two rather than one flash pages (i.e., 8KB rather than 4KB). This can make the tree shallower. Is this a good idea?



Question 3

Consider a table with columns A, B and C. Suppose we employ buffered insertions at a cost of $O(1/B)$ each.

A B C



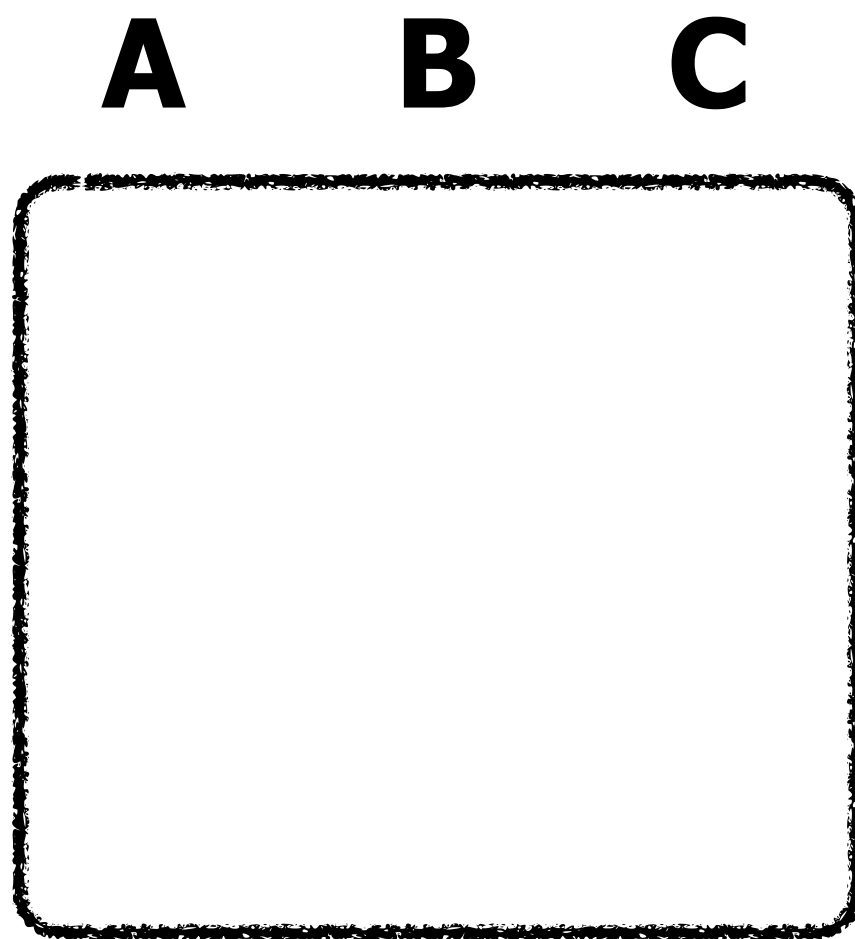
Workload:

50% Select * from table where A = "... " Returns one row

50% Insert (, ,)

Question 3

Consider a table with columns A, B and C. Suppose we employ buffered insertions at a cost of $O(1/B)$ each.



Workload:

50% Select * from table where A = "... " Returns one row

50% Insert (, ,)

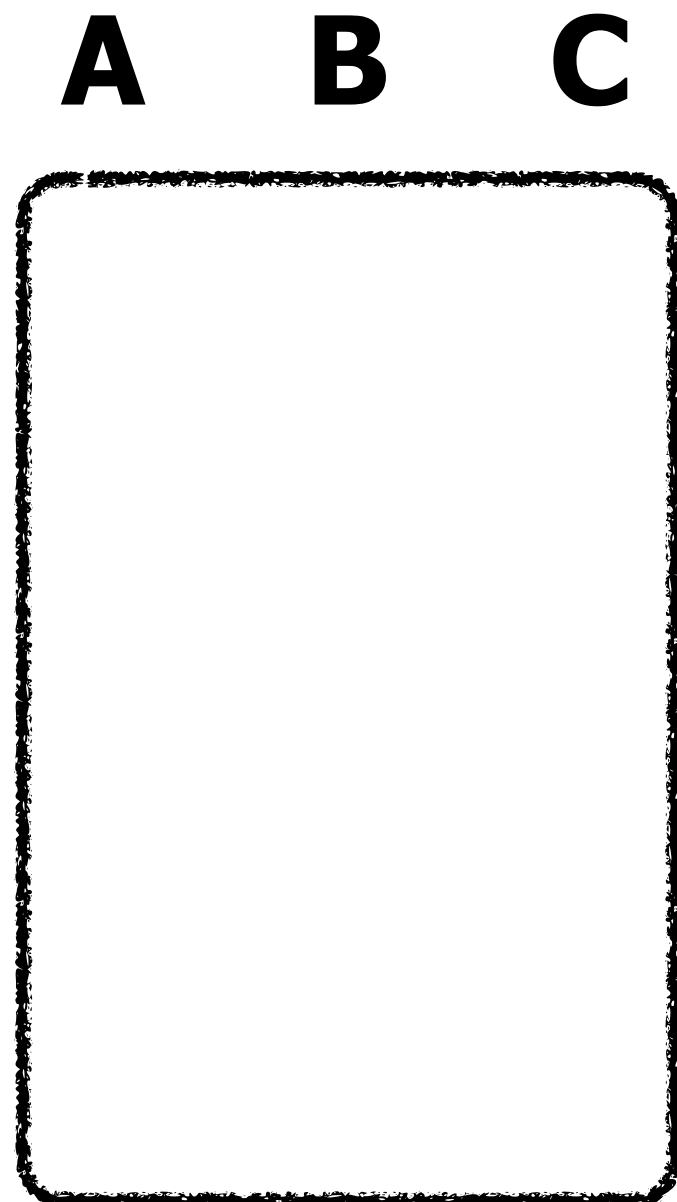
Should we employ a B-tree index on any of the columns?
Estimate the overall I/O cost of both operations with & without it.
(Back-of-the-envelope reasoning is sufficient)

Question 4

A table has columns A, B and C. The workload consists of two query types:

50% Select A from table where A = "..."/> Returns one row

50% Select * from table where B > x and B < y Returns avg. 10 rows



How should we index this table? B-tree or hash table? Clustered vs. unclustered? Estimate worst-case I/O cost with your plan for each query with these indexes assuming $N=2^{40}$ and $B=2^{10}$