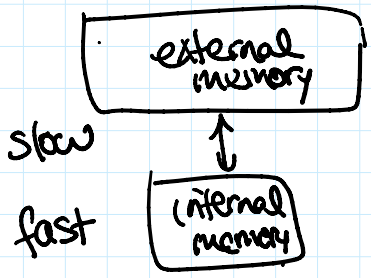
November 17, 2023 10:44 AM

External-Memory Data Structures

EM Model



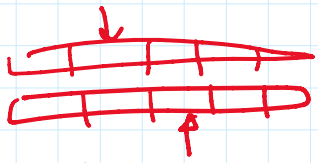
M = internal memory size.

B = block size

cost = # I/O ops (read/write a block)

assume $M \geq B^{1+\varepsilon}$

e.g. min $O(N/B)$ cost.

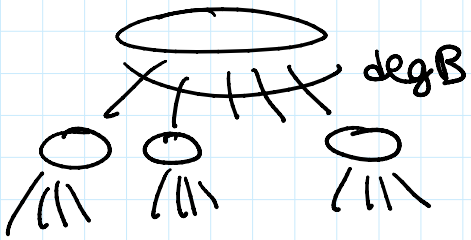


Sort: heapsort X
mergesort $O\left(\frac{N}{B}\log_2 N\right)$

Ex Pred Search

Method 1: B-tree

(or $O(\frac{N}{B})$ blocks)



space $O(N)$

query cost $\boxed{O(\log_B N)}$

$\left(Q(N) \leq Q\left(\frac{N}{B}\right) + \underbrace{O(1)}_{\text{I/O}}\right)$

dynamic $\deg\left(\frac{B}{4}, B\right) \Rightarrow$ update cost $\boxed{O(\log_B N)}$

better?

e.g. if we use B-trees to sort,

$\Rightarrow O(N \log_B N)$

but standard mergesort $O(\frac{N}{B} \log_2 N)$

best sorting algm. $O(\frac{N}{B} \log_{M/B} N)$

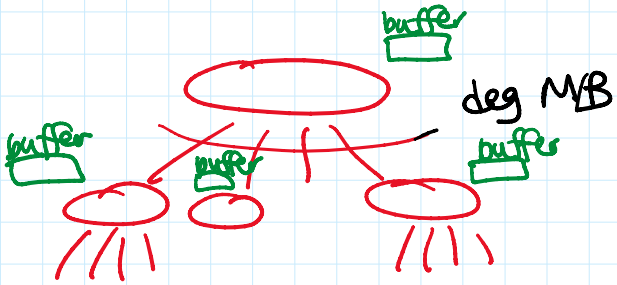
Method 2: Buffer Tree (Arge '03)

update cost amort. $O(\frac{1}{B} \cdot \log_{M/B} N)$ ← sub-const.

i.e. N updates take $O(\frac{N}{B} \log_{M/B} N)$ total cost.

idea - be lazy!

for each node, keep a buffer of up to M not-yet-processed updates



Update algm:

put request in root buffer

when buffer is full,

empty it by pushing to children

↳ need $O(\frac{M}{B})$ I/Os

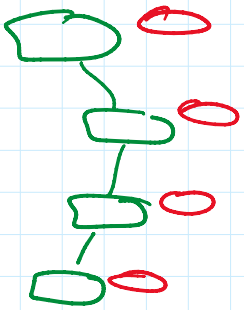
but done once after every M updates per level

$\Rightarrow O(\frac{1}{B})$ amort. per level

$\Rightarrow \boxed{O(\frac{1}{B} \cdot \log_{M/B} N)}$

$\Rightarrow \boxed{O\left(\frac{1}{B} \cdot \log_{M/B} N\right)}$

Query alg'm:



empty buffers along search path

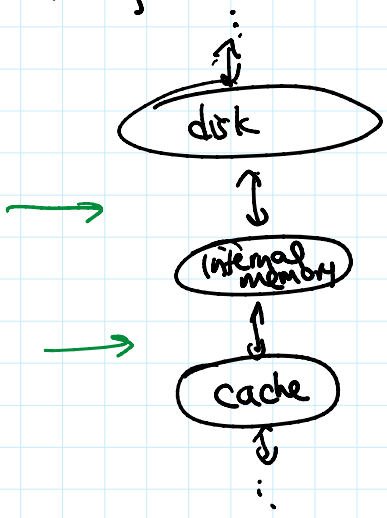
amort. $O\left(\log_{M/B} N \cdot \log_B \frac{M}{B}\right)$

↑ search among $\frac{M}{B}$ children

$= \boxed{O(\log_B N)}$

Rmk. other problems e.g. point location in 2D ... $O(\log_B N)$ --

Rmk. works for other levels of memory hierarchy



Cache-Oblivious Model:

algms not given value of B.

Just assume reasonable paging strategy (e.g. LRU)

Pros: all levels of memory hierarchy simultaneously!

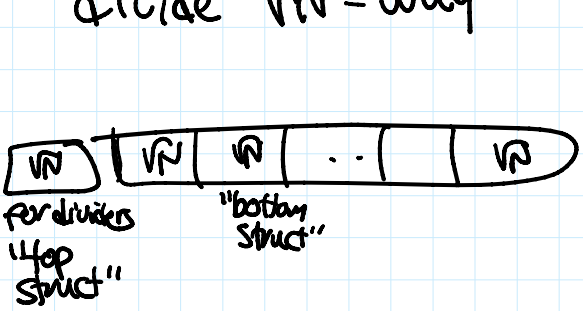
e.g. min in merge $O(N/B)$ cost

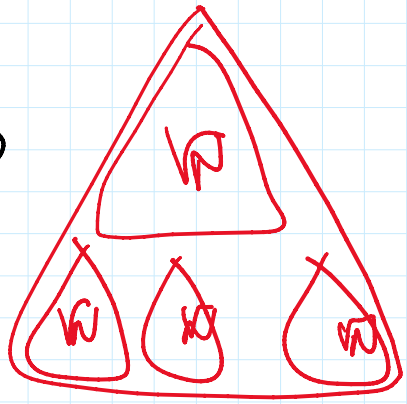
Sort $O(\frac{N}{B} \log_{\frac{M}{B}} N)$

Ex Static ↑ Pred Search

idea - van Emde Boas recursion

divide $\sqrt{N}$-way





query:
1. recurse top
2. recurse in 1 bottom struct.

$\Rightarrow$ 2 recursive calls: bad?

query cost:

$$Q(N) = \begin{cases} 2Q(\sqrt{N}) + O(1) & \text{if } N > B \\ O(1) & \text{if } N \leq B \end{cases}$$

↑ alg'm not need to know when to switch

expand ℓ levels $\Rightarrow$

$$Q(N) \leq 2^{\ell}\left(Q(N^{\frac{1}{2^{\ell}}}) + O(1)\right)$$

$$\leq O\left(\frac{\log N}{\log B}\right) \cdot O(1)$$

$N^{\frac{1}{2^{\ell}}} = B$

Set ℓ s.t. $N^{\frac{1}{2^\ell}} = B$.

$\frac{1}{2^\ell} \log N = \log B$

$2^\ell = \frac{\log N}{\log B}$

$\leq O\left(\frac{\log N}{\log B}\right) \cdot O(1)$

$= \boxed{O(\log_B N)}$

Rmk - can be made dynamic (Bender, Demaine, Farach-Colton ∞)

point location, ...