

TODAY: memory hierarchies II

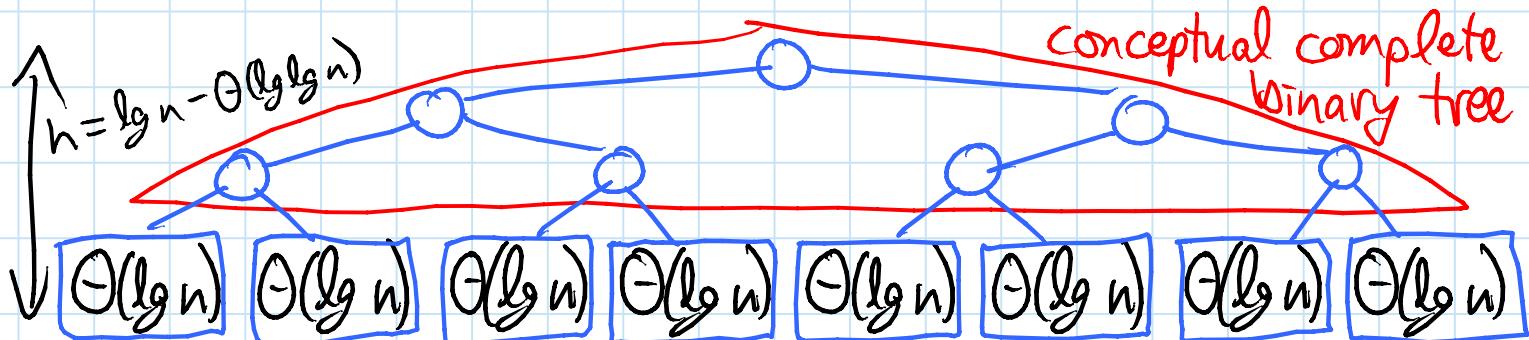
- ordered file maintenance (for B-tree in L20)
- list labeling (for persistence in L19)
- cache-oblivious priority queue

Ordered file maintenance: [Itai, Konheim, Rotem - ICALP 1981;
Bender, Demaine, Farach-Colton - FOCS 2000]

Goal: store N elements in specified order in an array of size $O(N)$ with gaps of size $O(1)$
 \Rightarrow scanning K consecutive elts. costs $O(\lceil \frac{K}{B} \rceil)$ mem.trans.
 subject to elt. deletion & insertion between 2 elts.
 by re-arranging elts. in array interval of $O(\lg^2 N)$ amortized elts., via $O(1)$ interleaved scans
 \Rightarrow costs $O(\frac{\log^2 N}{B})$ amortized memory transfers

Idea: upon updating element, ensure locally not too dense/sparse by redistributing elements in surrounding interval

- intervals defined by nodes in complete binary tree on $O(\lg n)$ -size chunks of array:



Update:

- update leaf node ($\Theta(\lg n)$ chunk) containing elt.
- walk up tree until reach node within threshold
- $\text{density}(\text{node}) = \frac{\# \text{elts. in interval below node}}{\# \text{array slots in that interval}}$
- density thresholds depend on depth d of node:
 - density $\geq \frac{1}{2} - \frac{1}{4} \frac{d}{h} \in [\frac{1}{4}, \frac{1}{2}]$ (not too sparse)
 - density $\leq \frac{3}{4} + \frac{1}{4} \frac{d}{h} \in [\frac{3}{4}, 1]$ (not too dense)
 - stricter at top of the tree (larger interval)
- evenly rebalance descendant elts. in node's interval

Analysis:

- thresholds get tighter as we go up
- \Rightarrow rebalancing a node puts children far within threshold:
 $|\text{density} - \text{threshold}| = \frac{1}{4h} = \Theta\left(\frac{1}{\lg N}\right)$
- \Rightarrow before this node is rebalanced again, must have $\Omega\left(\frac{\text{capacity}}{\lg N}\right)$ updates to bring child out of threshold
 $\Omega(1)$ because leaves have size $\Theta(\lg N)$
- \Rightarrow amortized rebuilding caused by update below a node
 $= \Theta(\lg N)$

- each leaf is below $h = \Theta(\lg N)$ ancestors
- \Rightarrow amortized rebuilding/update = $\Theta(\lg^2 N)$

Worst-case bounds possible [Willard - I&C 1992;

Bender, Cole, Demaine, Farach-Colton, Zito - ESA 2002]

Conjecture : $\Omega(\lg^2 N)$ necessary

List labeling: closely related problem

maintain explicit integer label in each node in a linked list, subject to insert/delete node here, such that labels are monotone at all times

(label = index in array)

label space best known time/update

$(1+\varepsilon)n \sim n \lg n$	$\Theta(\lg^2 n)$	- ordered file maintenance
$n^{1+\varepsilon} \sim n^{\Theta(1)}$	$\Theta(\lg n)$	→ 0 via modified threshold: density $\leq \frac{1}{\alpha^2}$, $1 < \alpha \leq 2$ [Dietz, Seiferas, Zhang - SIDMA 2005]
2^n	$\Theta(1)$	- trivial

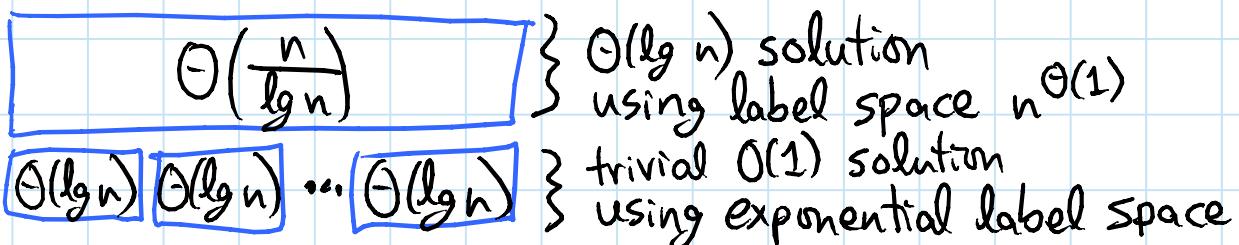
List order maintenance: easier problem, from L19

maintain linked list subject to insert/delete node here

& order query: is node x before node y ?

- $O(1)$ solution via indirection: [Dietz & Sleator - STOC 1987;

Bender, Cole, Demaine, Farach-Colton, Zito - ESA 2002]



- implicit node label = (top label, bottom label)
 $O(\lg n)$ bits

⇒ can compare two labels in $O(1)$ time

- top updates change many implicit labels at once (impossible in list labeling)

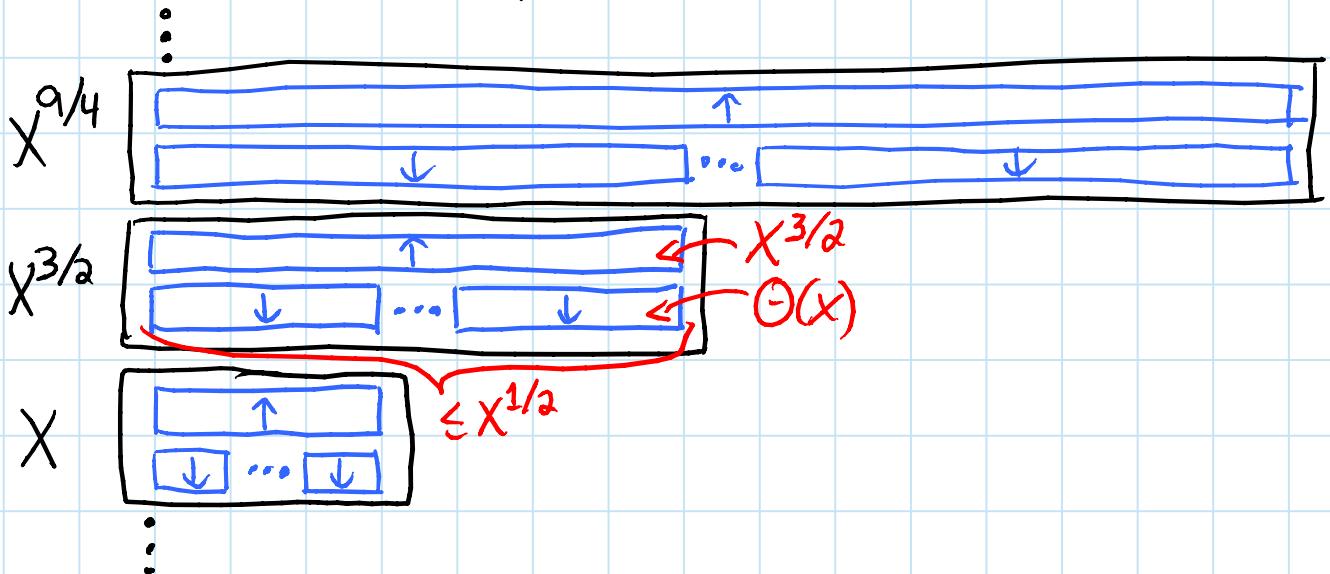
- bottom chunks slow top updates by $\Theta(\lg n)$ factor

⇒ $O(1)$ amortized cost

- worst-case bounds possible [same refs.]

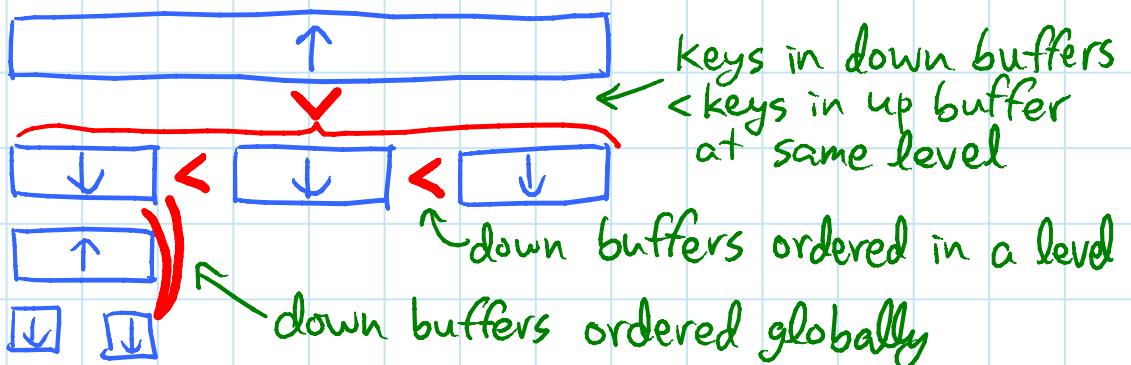
Cache-oblivious priority queue: [Arjen Bender, Demaine, Holland-Minkley, Munro - STOC 2002, SICOMP 2007; Brodal & Fagerberg - ICALP 2002]

- $\lg \lg n$ levels of size $N, N^{2/3}, N^{4/9}, \dots, c=O(1)$
- level $X^{3/2}$ has 1 up buffer of size $X^{3/2}$
 & $\leq X^{1/2}$ down buffers each of size $\Theta(X)$,
 all except first with $\Theta(X)$ elements



- levels stored consecutively, say smallest to largest

Invariants:



Insert:

- ① append to bottom up buffer
- ② swap into bottom down buffers if necessary
- ③ if up buffer overflows: push

Push X elements into level $X^{3/2}$

$\text{all} \geq \text{all elts. in down buffers at level } X \text{ (& below)}$

- ① sort elements
- ② distribute among down buffers (& possibly up buffer):
 - scan elts., visiting down buffers in order
 - when down buffer overflows: split in half, link list
 - when #down buffers overflows: move last to up buffer
 - when up buffer overflows: push it up to $X^{9/4}$

Delete-min:

- ① if bottom down buffers underflow: pull
- ② extract smallest elt. in bottom-left down buffer

Pull X smallest elts from level $X^{3/2}$ (& above)

- ① sort first two down buffers & extract leading elts.
- ② if $< X$: pull $X^{3/2}$ smallest elts. from level $X^{9/4}$ above
 - sort these elts. + up buffer
 - put larger elts. in up buffer (same # as before)
 - extract smallest elts. to get X total smallests
 - split rest into down buffers

- Analysis: push/pull at level $X^{3/2}$. sans recursion.
 costs $O\left(\frac{X}{B} \log_{M/B} \frac{X}{B}\right)$ memory transfers
- assume all levels of size $\leq M$ stay in cache \Rightarrow free
 - tall-cache assumption: $M \geq B^2$ (else change $3/2$)
 - push at level $X^{3/2} > M \geq B^2 \Rightarrow X > B^{4/3} \Rightarrow \frac{X}{B} > 1$
 - sort costs $O\left(\frac{X}{B} \log_{M/B} \frac{X}{B}\right)$ memory transfers
 - distribute costs $O\left(\frac{X}{B} + X^{1/2}\right)$ memory transfers
scan startup each down buffer
 - if $X \geq B^2$ then cost = $O\left(\frac{X}{B}\right)$
 - else: only one such level with $B^{4/3} \leq X \leq B^2$
 can keep 1 block per down buffer in cache:
 $X \leq B^2 \Rightarrow X^{1/2} \leq B \leq \frac{M}{B}$ by tall cache
 so just pay $O\left(\frac{X}{B}\right)$ at this level too
 - pull at level $X^{3/2} > M \geq B^2$:
 - sort costs $O\left(\frac{X}{B} \log_{M/B} \frac{X}{B}\right)$ memory transfers
 - another sort of $X^{3/2}$ elts. only when recursing
 \Rightarrow charge to recursive pull

Totaling:

- X elts. involved in push/pull costing $O\left(\frac{X}{B} \log_{M/B} \frac{X}{B}\right)$
- each elt. goes up & then down (more or less)

G real proof messier

$$\Rightarrow O\left(\frac{1}{B} \sum_X \underbrace{\log_{M/B} \frac{X}{B}}_{\text{exp. geometric}} \right) \text{ amortized cost per element}$$

\hookrightarrow geometric

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

□