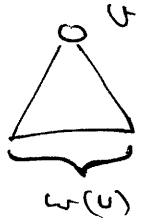
Weight-Balanced B-trees

Def.: Weight $w(v)$ of node v = # of elements in leaves of v's subtree



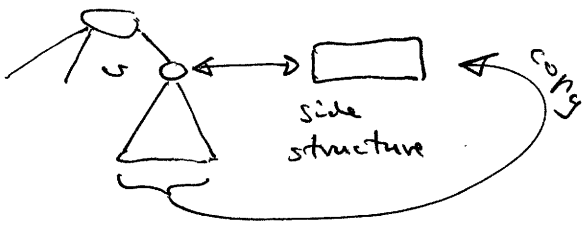
In some applications (mainly in computational geometry), nodes of B-trees have associated structures *) of size $w(v)$. Rebalance at $v \Rightarrow v$'s side structure must be rebuilt. Often takes time linear in side structure size.

If the following is true, we can bound the time spent on rebuilding side structures:

> In between two update operations involving v, at least $\Omega(w(v))$ updates took place in v's subtree †)

Then, the work (of the rebuilding) can be charged to these updates,

[*) Usually, the contents of a side structure is based on contents of the node's subtree:]



leading to low amortized cost.

Unfortunately, B-trees do not fulfill †).

Example:

Keep deleting smallest elm. until root r disappears.
Then keep inserting new smallest elm. until
root r appear again.

[Recall: splits give nodes of degree $b/2$]

Rebalancing at root happens with intervals of $\Theta((b/2)^{height})$ updates (as this is size of expanding/retracting part). But size of r (and of its right child) is $\Theta(b^{height})$.

Difference is factor $(\frac{1}{2})^{height} \neq \Theta(1)$.

New variant of B-trees : Weight-Balanced B-trees

Def: Let l be level of node (leaves have level 1, levels increase upwards). Let b be a parameter, $b \geq 8$.

+ All leaves are on same level.
+ All nodes v except root:

$$\frac{1}{4} \cdot b^l \leq w(v) \leq b^l$$

+ Root: has at least two children, and:

$$w(\text{root}) \leq b^l$$

[Note: Lars Arge's notes have a separate parameter k for weights of leaves, and let level of leaves be 0.]

Height: Let h = height (= level of root - 1)

$$N = w(\text{root}) \geq \underbrace{2}_{\text{at least two children}} \cdot \underbrace{\frac{1}{4} \cdot b^h}_{\text{min weight of root's children}}$$

$$\Updownarrow$$

$$\log_b(2N) \geq h$$

Fanout of nodes:

Max is $\frac{b^l}{\frac{1}{4} b^{l-1}} = 4 \cdot b$

Min is $\frac{\frac{1}{4} \cdot b^l}{b^{l-1}} = \frac{1}{4} \cdot b$

Choose $b = \Theta(B) \Rightarrow \Theta(1)$ blocks per node.

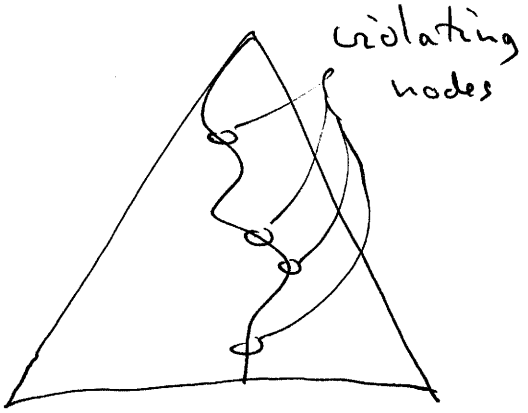
Search: as usual in B-trees.

Updates: Search + update in leaf + rebalance.

Unbalance possible on nodes on search path:

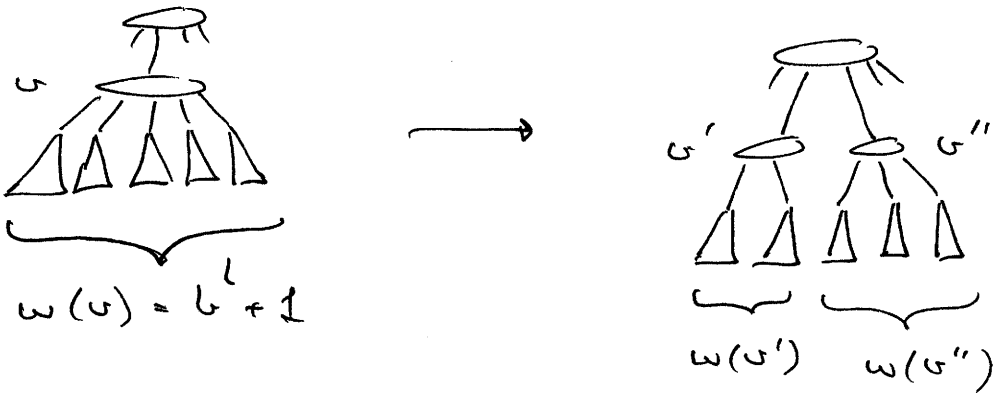
After inserts: $w(v) = b^l + 1$

After updates: $w(v) = \frac{1}{4} \cdot b^l - 1$

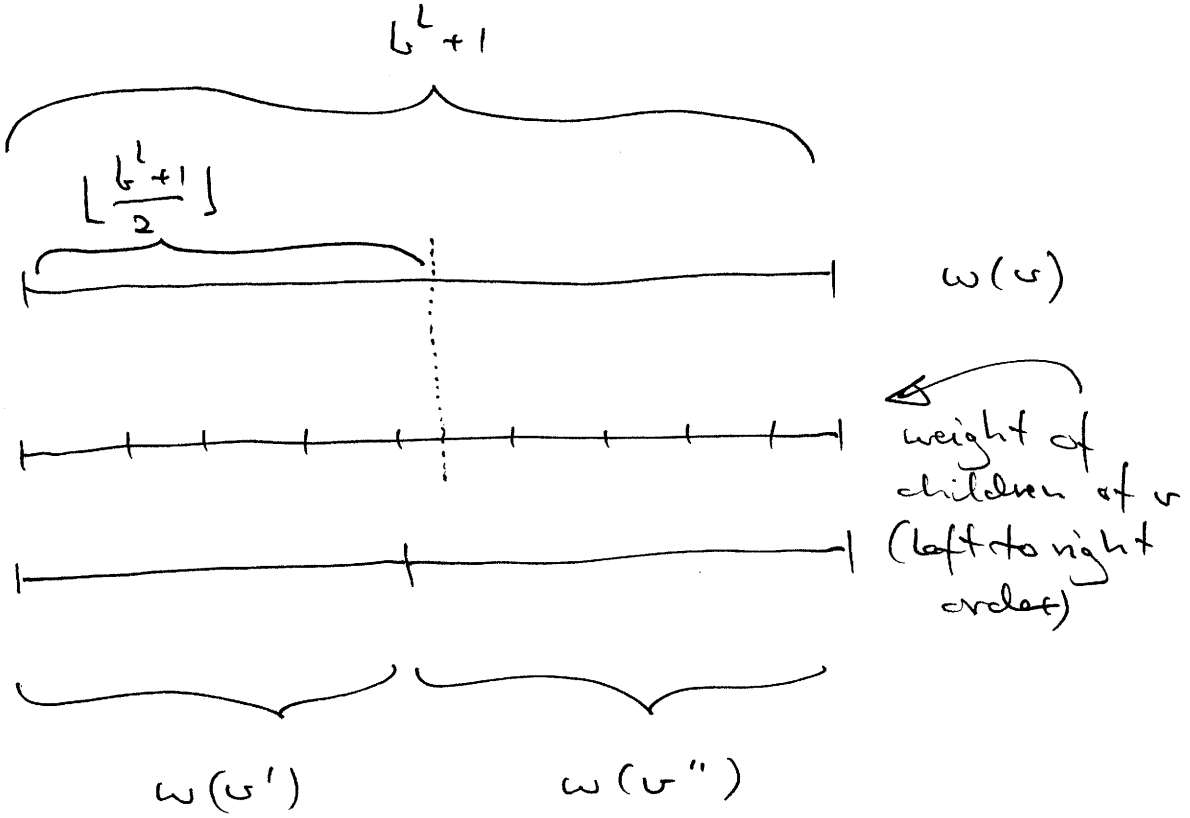


Resolve/rebalance
bottom-up

SPLIT :



Split such that $w(v')$ and $w(v'')$ are as even as possible



We have $w(v'') \geq w(v') \geq \lfloor \frac{b^l+1}{2} \rfloor - (b^{l-1} - 1)$

$\geq \lfloor \frac{b^l+1}{2} \rfloor - (b^l \cdot \frac{1}{8} - 1) \geq \frac{b^l}{2} - 1 - b^l \cdot \frac{1}{8} + 1$

$= \frac{3}{8} \cdot b^l$

$(b \geq 8)$

$$w(v') \leq w(v'') \leq$$

$$\left\lceil \frac{b^l+1}{2} \right\rceil + \left(b^{l-1} - 1\right) \leq \frac{b^l+1}{2} + \frac{1}{2} + b^{l-1} - 1$$

$$= \frac{b^l}{2} + b^{l-1} \leq \frac{b^l}{2} + \frac{1}{8} b^l = \frac{5}{8} \cdot b^l$$

$$(b \geq 8)$$

FUSE : $w(v) = \frac{1}{4} b^l - 1$

$w(\text{sibling}) \in \left[\frac{1}{4} b^l ; b^l\right]$

Fuse $\Rightarrow$ new node v'.

$$w(v') \geq \frac{1}{4} \cdot b^l - 1 + \frac{1}{4} b^l = \frac{1}{2} b^l - 1$$

$$\geq \frac{1}{2} b^l - \frac{1}{8} b^l = \frac{3}{8} \cdot b^l$$

$$\begin{pmatrix} b \geq 8 \\ l \geq 1 \end{pmatrix}$$

If $w(v') > \frac{7}{8} b^l$: split new node v', creating a SHARE. Else keep as a FUSE.

SHARE:

New nodes after v' is split: v'' and v'''.

$$w(v''') \geq w(v'') \geq \left\lfloor \frac{\frac{7}{8}b^l + 1}{2} \right\rfloor - \left(\frac{1}{8}b^l - 1\right)$$

$$\geq \frac{7}{16}b^l + \frac{1}{2} - \frac{1}{2} - \left(\frac{1}{8}b^l - 1\right)$$

$$\geq \frac{5}{16} \cdot b^l .$$

$$w(v'') \leq w(v''') \leq \left\lceil \frac{\frac{1}{4}b^l - 1 + b^l}{2} \right\rceil + \left(\frac{1}{8}b^l - 1\right)$$

$$\leq \frac{5}{8}b^l - \frac{1}{2} + \frac{1}{2} + \frac{1}{8}b^l - 1$$

$$\leq \frac{6}{8}b^l = \frac{3}{4}b^l$$

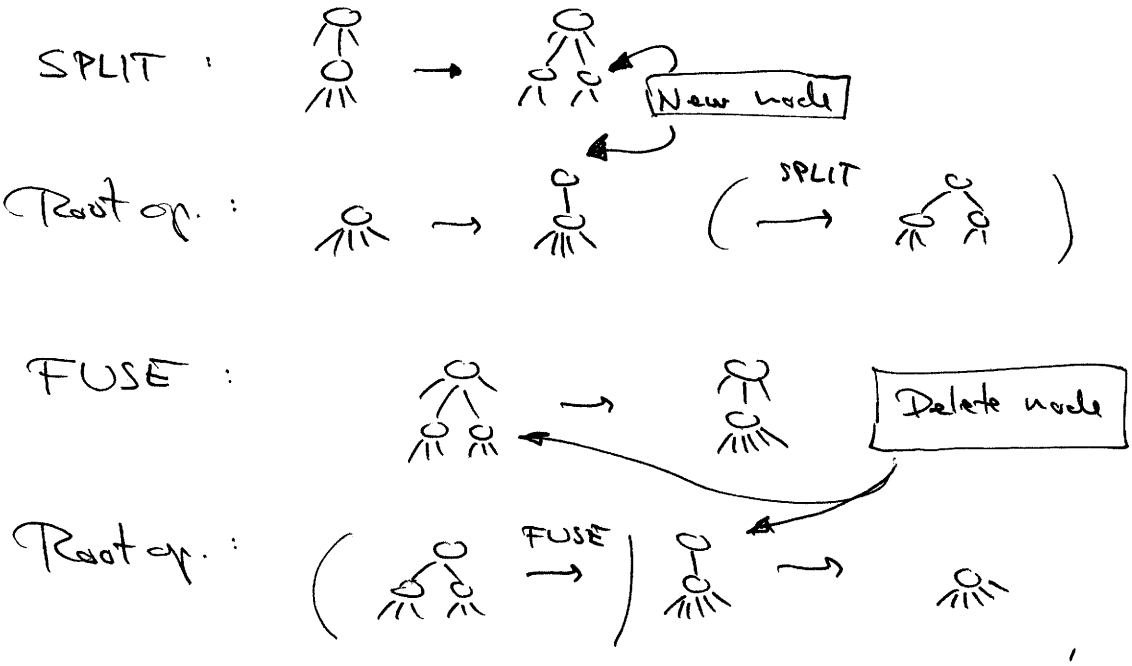
The calculations above show that all the rebalancing operations (SPLIT, FUSE, SHARE) leave all affected nodes [i.e. nodes with changed weight] with a weight in (v', v'', and v''')

*) $$\left[\frac{5}{16}b^l, \frac{7}{8}b^l\right]$$

In particular, they are legal nodes.

But more can be deduced:

Note that during the life span of a node, which starts by a SPLIT or root operation, and ends by a FUSE or root operation :



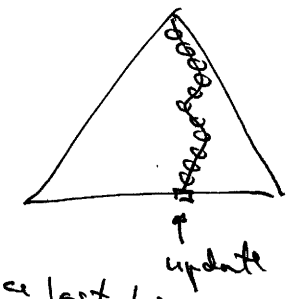
a node can only change weight by :

1) Updates taking place in leaves of its subtree

2) The node participating in a SPLIT, FUSE, or SHARE.

Since each occurrence of 2) leaves the node with a weight in the interval *), there must have been at least $\frac{1}{16} b^l$ instances of 1) before the node can be violating (ie., have weight $b^l + 1$ or $\frac{1}{4} \cdot b^l - 1$) and give rise to a rebalancing operation.

This means that if we charge an update $16 \cdot \frac{1}{b^l}$ for each node on its search path [those nodes where l) happens and weight is changed by ± 1], then a violating node will have saved up [by the charging of the updates passing it] a value of 1. So if rebalance operations cost is $O(1)$, these are covered by the charging scheme. The total charge on an update is at most



(since last time it was violating)

$$16 \cdot \sum_{l=1}^{\infty} \frac{1}{b^l} = \frac{16}{b} \cdot \sum_{l=0}^{\infty} \frac{1}{b^l} = \frac{16}{b} \cdot \underbrace{\frac{1}{1-1/b}}_{O(1)}$$

$$= O\left(\frac{1}{b}\right)$$

Hence, the amortized cost of rebalancing is $O\left(\frac{1}{b}\right)$ pr. update.

[Eg. when having side structures in nodes]

If rebalance operations cost (for each) is instead $O(w(v))$ [or $O\left(\frac{w(v)}{I}\right)$], then we can instead charge an update 16 [or $16 \cdot \frac{1}{I}$] for each node on its search path.

Then a violating node will always have saved up a value of $16 \cdot \frac{1}{16} b^l = b^l = \Theta(w(v))$

[or $\frac{16}{B} \cdot \frac{1}{16} b^l = \Theta\left(\frac{w(v)}{B}\right)$].

The total charge for an update is

$$\sum_{l=1}^{\text{height of tree}} 16 = O(\text{height of tree}) = O(\log_B N)$$

[or $O\left(\frac{1}{B} \cdot \log_B N\right)$],

which is then the amortized cost of rebalancing.

Detail omitted above: a new root is created with weight $b^{l'}+1$, where l' is the level of the new roots children (ie. $l' = l-1$), when created in connection with a SPLIT. Since $b^{l'}+1 = b^l \cdot \frac{1}{b} + 1 = b^l\left(\frac{1}{b} + \frac{1}{b^l}\right) \leq b^l \cdot \frac{2}{b} \leq b^l \cdot \frac{2}{8} = b^l \frac{1}{4}$, $w(\text{new root}) \leq b^l \cdot \frac{1}{4}$.
It may also be created in connection with a FUSE. There we already know $w(\text{new root}) \leq \frac{7}{8} b^l$ by *).

[See figures on page (8) for the possible ways to get a new root].

Since roots can only be violating when too big ($w(\text{root}) = b^l + 1$), the above analysis of at least $\frac{1}{16} b^l$ updates that can be charged still holds. [Recall, rebal. ops. are paid by the charge left on the violating node triggering the rebal. operation.]