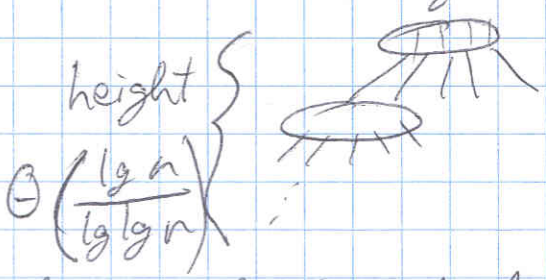
Fusion trees

6.897
L 3.4
Feb. 19, 2003

Model (transdichotomous RAM):
- each element is a word (can be manip. in $O(1)$)
- know (assume) word $\geq \lg n$ bits or so
  but may be much larger!

Main idea

- start with static (as usual)
- initial goal: $O(\lg n / \lg\lg n)$ per op. (!) , $O(n)$ space
- B-tree with branching factor $B = (\lg n)^{1/5}$



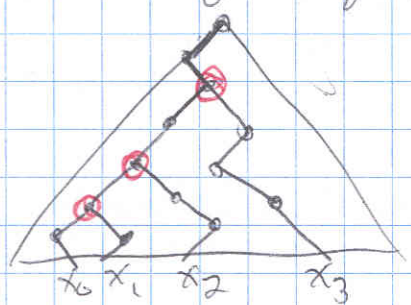
- key for search (single node): how to determine where q fits among $(\lg n)^{1/5}$ words-length keys in $O(1)$ time?

Without preprocessing, can't even afford to look at the input ($(\lg n)^{1/5}$ words) in $O(1)$ time!

What matters in a node?

L 4.1

- suppose keys in a node are $x_0 < x_1 < \cdots < x_{k-1}$, $k = \Theta(B)$
- view keys as binary strings / paths in binary tree



- # branch nodes $= k-1$, on $\leq k-1$ different levels

$\Rightarrow < k$ bits suffice to distinguish the x_i's.

Call them $b_0 < b_1 < \cdots < b_{r-1}$ $\quad r < k = \Theta(B)$.