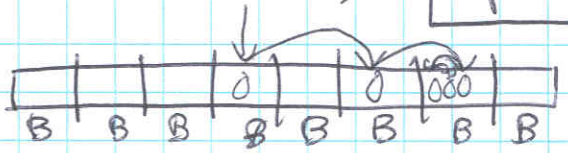
Binary Search / Static Search Trees

6.897
L16.2
Apr. 23, 2003

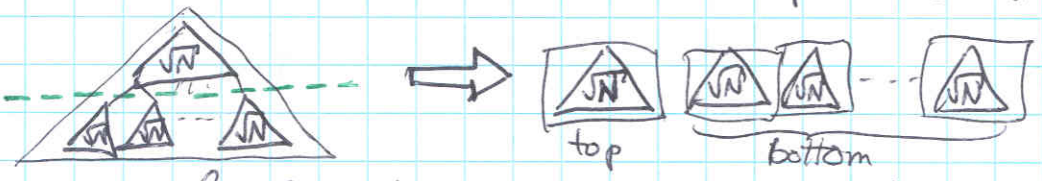
Naïve solution: binary search (doesn't use B or M).

- cost: $\lg(N/B) = \lg N - \lg B$
- want: $\log_B N$. like B-trees



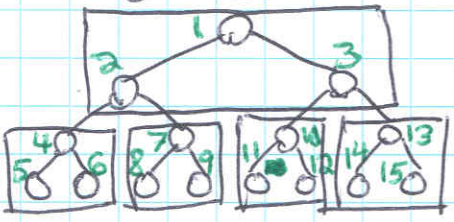
van Emde Boas layout: [Prokop 1999] [Bender, Demaine, Farach-Colton-FOCS 2000]

- store N elements in-order in complete binary tree on N nodes
- carve tree at middle level ⇝ $\sqrt{N}+1$ pieces of $\sqrt{N}$ each



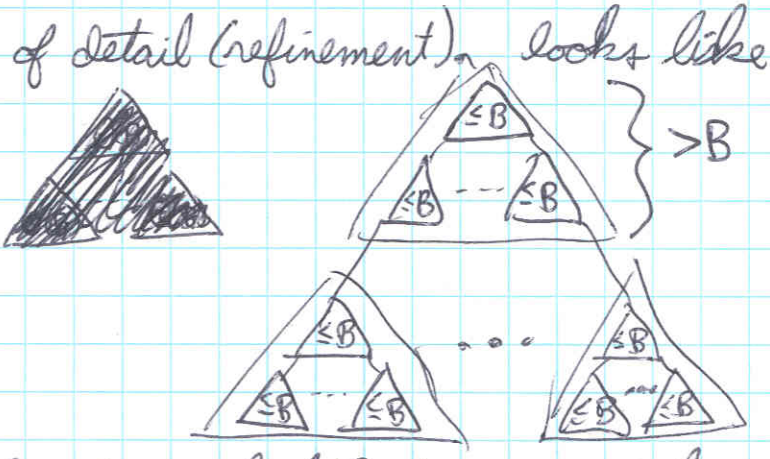
- recursively layout pieces, concatenate them together

e.g.:



Analysis:

- at some level of detail (refinement), looks like:



- we're cutting height in half & stopping at height $\le \lg B$.
- so pieces might have height as low as $\frac{\lg B}{2} \Rightarrow \sqrt{B}$ nodes but $\Theta(\lg B)$
- # pieces visited on root-to-leaf path $= \Theta\left(\frac{\lg N}{\lg B}\right) = \Theta(\log_B N)$
- each piece occupies ≤ 2 blocks (alignment)

Theorem: cost of search $\le 4 \log_B N + 2$.

- generalizes to height $\neq 2^k$ & const. node degrees ≥ 2.