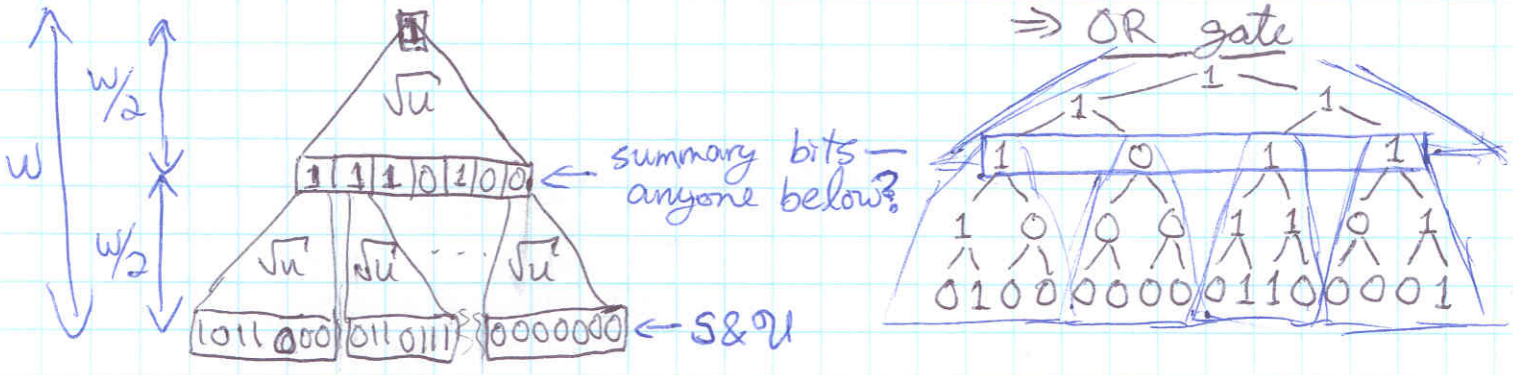
6.897
L 9.4
Mar. 3, 2005

Successor(x, S):
- if $x < S.\min$: return $S.\min$
- x is at position $\text{low}(x)$ in $S[\text{high}(x)]$
- if $\text{low}(x) < S[\text{high}(x)].\max$ ← key: augmentation
  then: return $\text{high}(x) \cdot \sqrt{|S|} + \text{Successor}(\text{low}(x), S[\text{high}(x)])$
  else: $i \leftarrow \text{Successor}(\text{high}(x), S.\text{summary})$
  return $i \cdot \sqrt{|S|} + S[i].\min$

Insert(x, S):
- if $S.\min = \text{none}$ then $S.\min \leftarrow x$, stop
- if $S[\text{high}(x)]$ is empty $(.\min = \text{none})$
  then: $S[\text{high}(x)].\min \leftarrow \text{low}(x)$ ← key: not stored recursively
  $\text{Insert}(\text{high}(x), S.\text{summary})$
  else: $\text{Insert}(\text{low}(x), S[\text{high}(x)])$

Analysis:
- key: one recursive call on word half the size
  ⇒ $O(\lg w)$
- Predecessor & Delete similar

Expand recursion: tree view



- can't afford to update all these bits ~
  but the ones really needed by Successor are there
- not storing min recursively avoids a lot of these updates