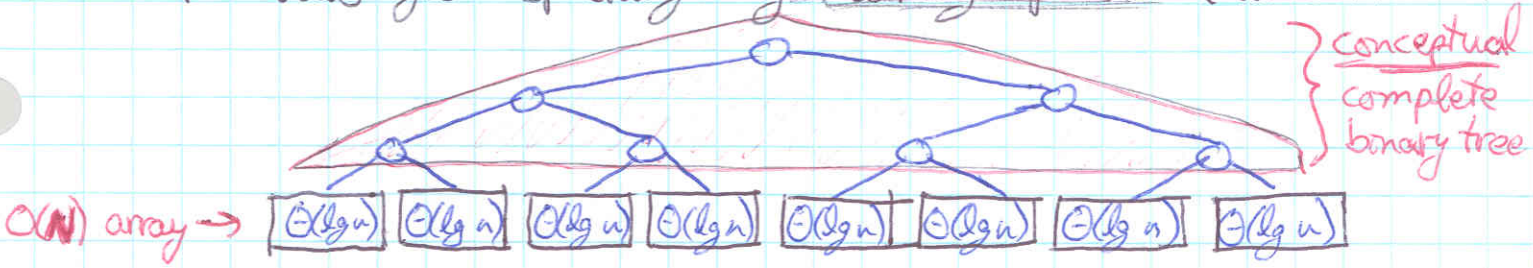
Ordered file maintenance

[Itai, Konheim, Rodeh - ICALP 1981]
[Bender, Demaine, Farach-Colton - FOCS 2000]

- goal: store N elements in specified order in array of size $O(N)$ with gaps of size $O(1)$
  $\Rightarrow$ scanning K consecutive elements takes $O(1 + K/B)$ memory transfers
  & support deleting an element / inserting element between 2 given elements by re-arranging elements in array interval of $O(\lg^2 N)$ am.
  $\Rightarrow$ costs $O(\frac{\lg^2 N}{B})$

- rough idea: upon updating an element, ensure locally not crowded/sparse
  - grow an interval around the element until "right" ratio of elements to space
    ↳ not too dense or too sparse
    ↳ depends on size of interval!
  - evenly redistribute elements in that interval

- intervals grow specially: by walking up tree (node ↔ interval)



  - leaves of tree represent $\lg N$ consec. slots in array
    $\Rightarrow$ minimum interval length $= \lg N$
  - parent extends interval by factor of 2 in some direction

- density(node) $= \frac{\#\text{ elements in interval}}{\#\text{ slots in interval}}$

- density constraints depend on depth d of node (interval length):
  - density $\geq \frac{1}{2} - \frac{1}{4}\frac{d}{h} \in [1/4, 1/2]$ ~ not too empty
  - density $\leq \frac{3}{4} + \frac{1}{4}\frac{d}{h} \in [3/4, 1]$ ~ not too full
  } density thresholds

- constraints are strongest at low depth d = long intervals
- constraints are not hard ~ allow violation
  ~ fix when violation "seen" locally around update