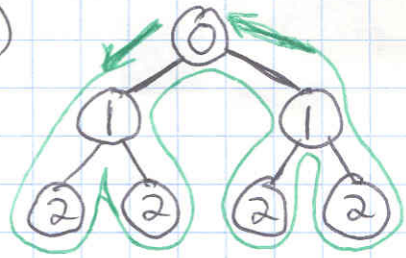
LCA [Bender & Farach-Colton 2000; Berkman et al. 1989] [originally Harel & Tarjan-SICOMP 1984]

① Reduce to range-min query "±1 case"

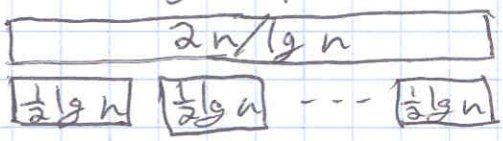
- Take Euler tour of tree (DFS with repeats)
- Store depths of nodes as visited
- LCA(i, j) = (index of) min depth in range i:j



0 1 2 1 2 1 0 1 2 1 2 1 0

Extra property: adjacent values (depths) have difference of ±1.

② Split into groups of $\frac{1}{2}\lg n$



- range min overall = min { range min in leftmost block, range min in rightmost block, range min. of entire blocks in between }

③ Lookup table for blocks

- blocks are invariant under translation (+k on all values) [index of range mins still the same]
- so translate all blocks so that 1st number is 0.
- # distinct block types now $2^{\frac{1}{2}\lg n} = \sqrt{n}$.
- lookup table: (block type, left pos., right pos.) $\mapsto$ range-min pos.
  - block type: $\sqrt{n}$; left pos.: $\frac{1}{2}\lg n$; right pos.: $\frac{1}{2}\lg n$; range-min pos.: $\frac{1}{2}\lg n$
- space = $O(\sqrt{n}\lg^2 n \lg\lg n)$ bits $= o(n)$ bits.

④ Trivial range-min query: $O(n^2)$ space, $O(1)$ time (store all answers)

⑤ Smarter range-min query: $O(n \lg n)$ space, $O(1)$ time

- store answer from any start point ← n
  & with length = power of 2 ← $\lg n$
- $O(n \lg n)$ space
- Any interval of length k can be covered by 2 of length $\lfloor k \rfloor$