# 07-B1

## BST Application

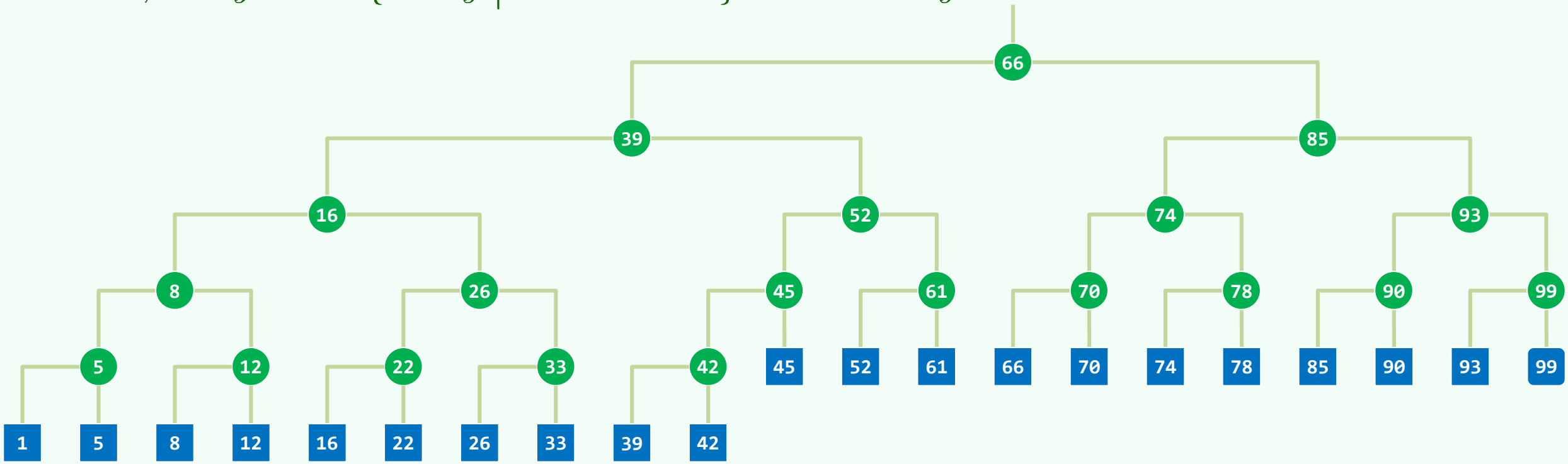
### kd-Tree: 1D

邓俊辉

deng@tsinghua.edu.cn

# Structure: A Complete (Balanced) BST

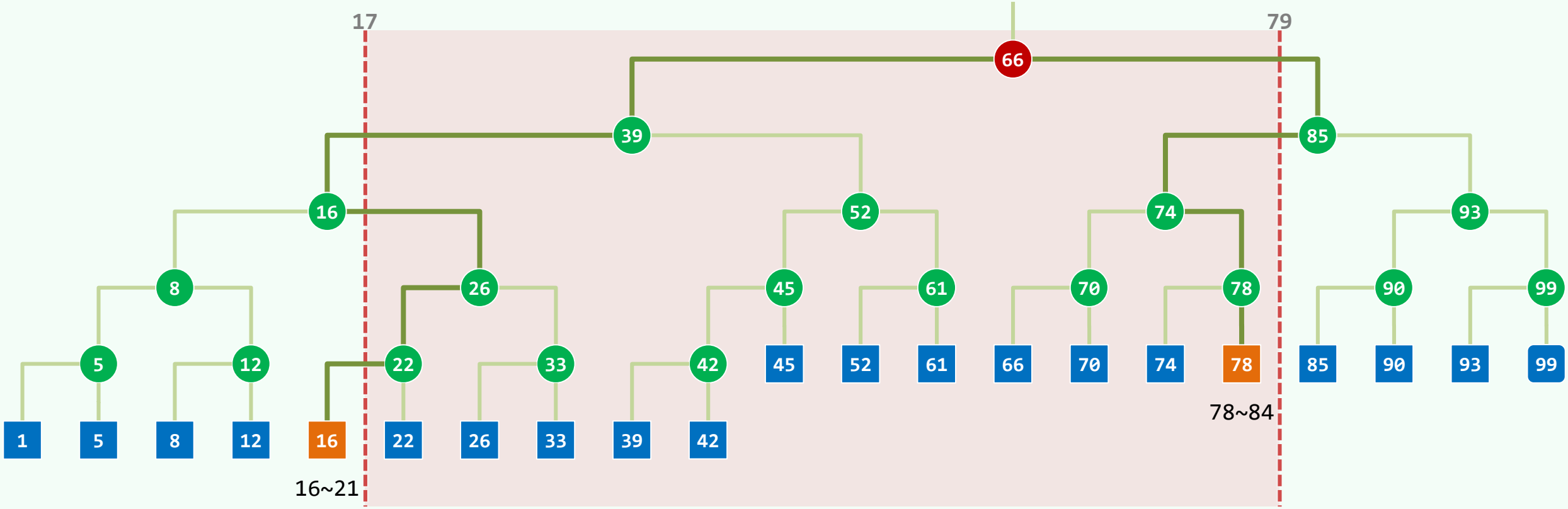
- $\forall v,\ v.key = \min\{\, u.key \mid u \in v.rTree \,\} = v.succ.key$




- $\forall u \in v.lTree,\ u.key < v.key$ $\qquad$ $\forall u \in v.rTree,\ u.key \geq v.key$
- **search(x) : returns the MAXIMUM key not greater than x**

# Lowest Common Ancestor

- Consider, as an example, the query for [17, 79] ...

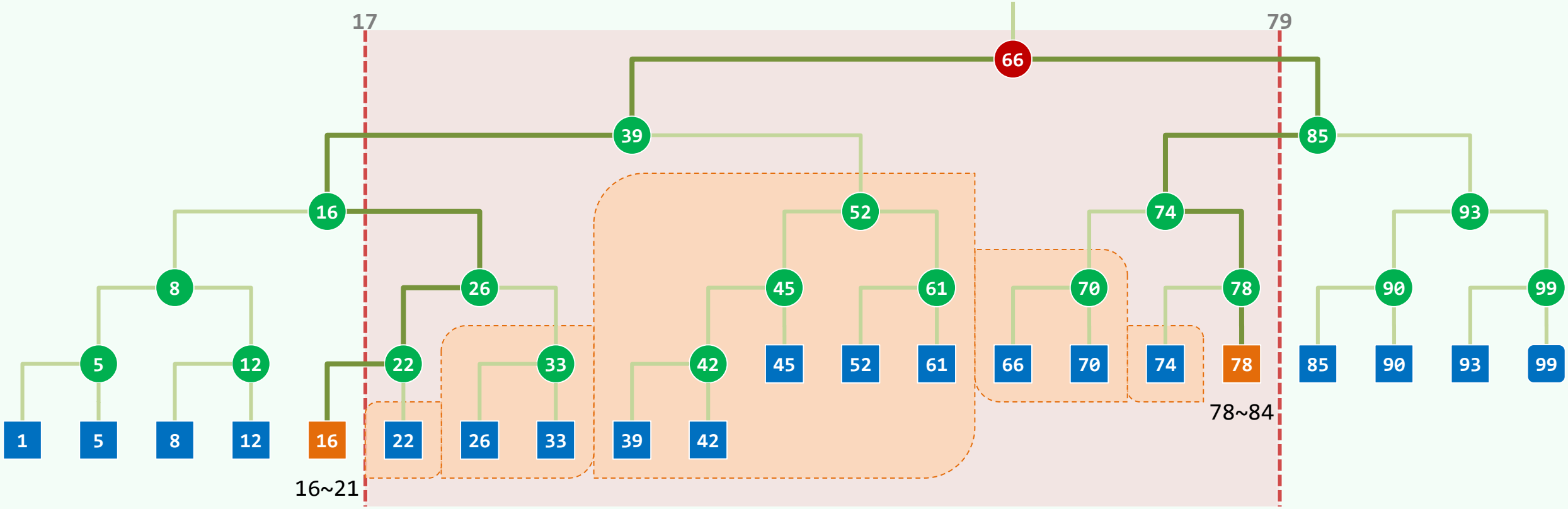



- Do search(17) = 16 (might rejected) and search(79) = 78 (must accepted)

- Consider LCA(16, 78) = 66 ...

# Union of $\mathcal{O}$(logn) Disjoint Subtrees

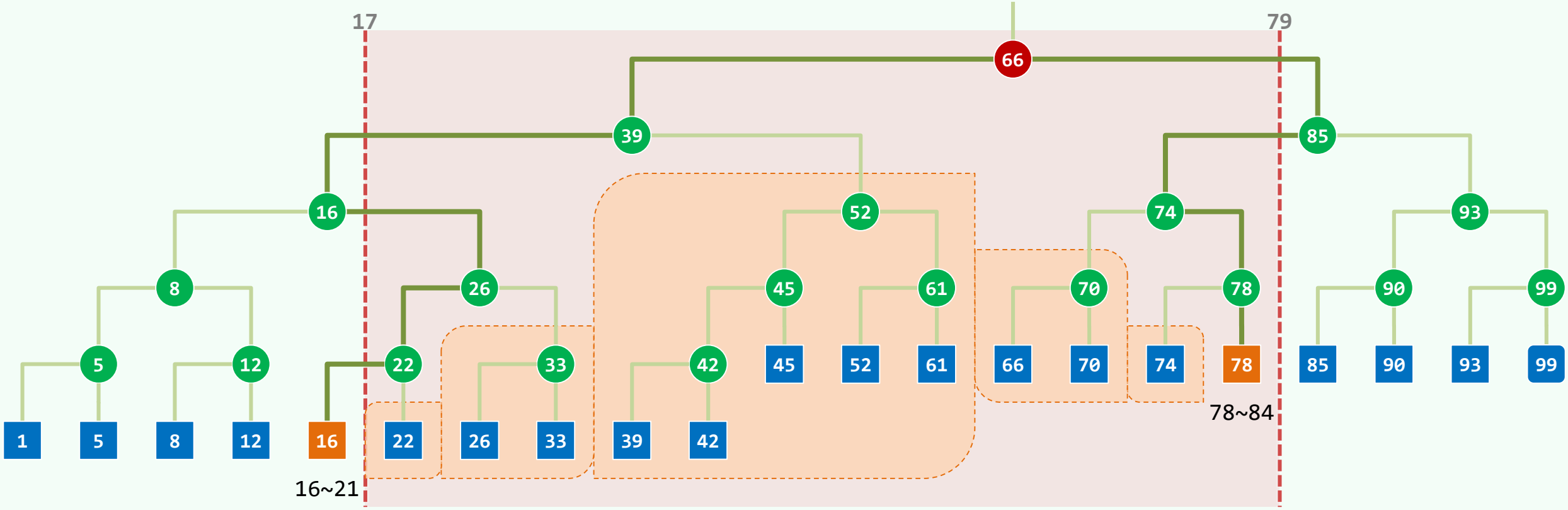
❖ Starting from the LCA, traverse path(16) and path(78) once more resp.




- All R/L-turns along path(16)/path(78) are ignored and

- the R/L subtree at each L/R-turn is reported

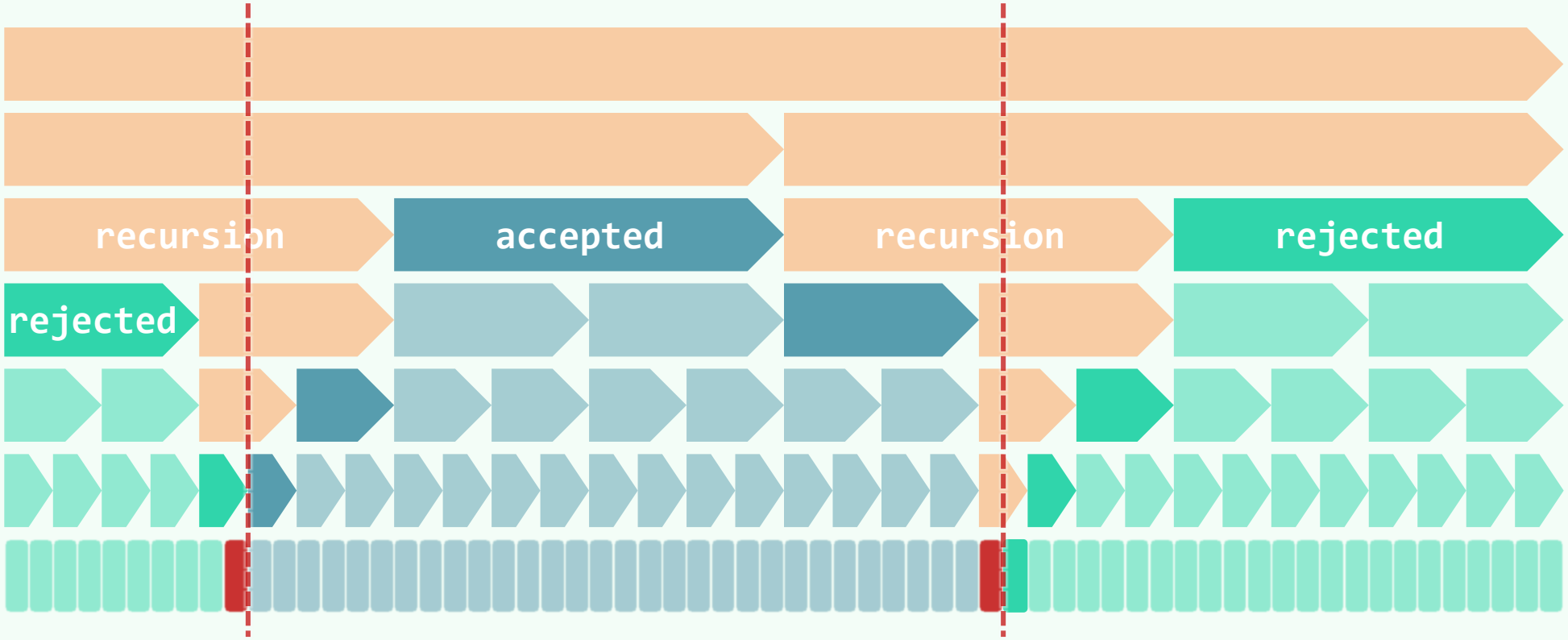
# Complexity




**Query:** $\mathcal{O}(\log n)$ **Preprocessing:** $\mathcal{O}(n \log n)$ **Storage:** $\mathcal{O}(n)$

# Hot Knives Through A Chocolate Cake of Height $O(\log n)$

- Region(u) is enclosed by Region(v) iff u is a descendant of v in the 1d-tree
- Region(u) and Region(v) are disjoint iff neither is the ancestor of the other




- All nodes are partitioned into 3 types: accepted + rejected + recursion