Tree decompositions
- preferred paths
- heavy-light decomp.

* ART/leaf-trimming decomp.
* separator decomp.

Tango trees, link-cut trees
link-cut trees
level-ancestors

Separator theorem on trees: [Jordan 1864-
“Sur les assemblages de lignes”–Journal Reine Angew. Math.]
Any tree on n vertices has a vertex whose
removal disconnects the tree into components of size $\leq \frac{n}{2}$

Proof:
- pick any vertex
- if not done, exactly one component has size $> \frac{n}{2}$
- walk one step into that component
- new component of old vertex +
  other pieces has $< \frac{n}{2}$ vertices
$\Rightarrow$ never go back to old vertex
$\Rightarrow$ terminate $\square$
Separator decomposition:
- apply separator theorem \(\Rightarrow\) root of new tree
- recurse on components \(\Rightarrow\) children subtrees
\[\Rightarrow\text{depth of new tree } = O(lg n)\]

Application: Oracle Search [Aronov et al. - LATIN 2006]
- suppose you're looking for a cat (node) in a tree
- if you ask Oracle about an edge \((u, v)\),
  it tells you which subtree contains cat
- separator decomposition tree lets you find cat in \(O(lg n)\) calls to Oracle (if bounded degree)
- example: maintain set of points subject to finding farthest from query point
**ART decomposition:**

- define **bottom tree** rooted at each maximally high node with $\leq \log n$ leaves below
  $\Rightarrow$ compressed size $O(\log n)$, disjoint

- **top tree** on remaining nodes $\Rightarrow \leq \frac{n}{\log n}$ leaves
  (charge each top leaf to $>\log n$ leaves in subtree)

- recurse in **top tree**
Marked ancestor problem [Alstrup, Husfeldt, Rauhe - FOCS 1995]
- rooted tree - here, static
- each node either marked or unmarked
- updates: mark(v) & unmark(v)
- query: lowest marked ancestor of v

Bounds:
* $O\left(\frac{\log n}{\log \log n}\right)$ query, $O(\log \log n)$ update \today
- $\Omega\left(\frac{\log n}{\log \log n}\right)$ query [chronogram technique]
- $\Omega(\log \log n)$ update, even in path: colored predecessor problem with $u=n$

Application: dynamic method dispatching in OOP
- tree = inheritance among classes
- mark = class implements method X
- query = call to X

OPEN: DAGs from multiple inheritance
Marked ancestor upper bound:
- recursive ART decomposition
  \( \Rightarrow O(\frac{lg n}{\sqrt{\lg n}}) \) levels of recursion

Each bottom tree:
- maintain bitvector of which nonbranching paths have a marked node, ordered by depth
- each node stores bitmask of its ancestor paths
  \( \Rightarrow \) in \( O(1) \) time (mask + LSB), find right path
- maintain predecessor DS on each nonbranching path
  \( \Rightarrow O(\lg \lg n) \) time/op.: \( u = n \)
- only used at end of query:
  if no marked ancestor in bottom tree,
  recurse with parent(root) in top tree
  \( \Rightarrow O(\frac{lg n}{\sqrt{\lg n}} + \lg \lg n) \) query
- each node stores recursion depth at which it's in bottom tree
  \( \Rightarrow \) update = predecessor update for path
    + bitvector update for bottom tree
  \( \Rightarrow O(\lg \lg n) \)
Decremental connectivity in a tree: $O(1)$ amortized (assuming all $n-1$ edges deleted)

1. $O(lg n)$ amortized update (or use link-cut/Euler-tour)
   - each node stores explicit component id.
   - delete($v, w$):
     - run DFS from both $v$ & $w$ in parallel
     - stop when one DFS stops $\Rightarrow$ smaller component
     - update all nodes in smaller component to new id.
     - component containing updated node shrinks by $2x$
     $\Rightarrow$ $O(lg n)$ updates to any node
   - can also store mapping from id. to root of component

2. $O(1)$ amortized for path
   - split into chunks of length $lg n \Rightarrow \leq \frac{n}{lg n}$ chunks
   - store each chunk as bitvector
   - use $O(lg n)$ structure to store which chunks have cut
   - query: find right chunk, shift+LSB within chunk

3. $O(1)$ amortized for top tree
   - use $O(lg n)$ structure on $O(\frac{n}{lg n})$ nonbranching paths
   - use path structure on each nonbranching path

4. $O(1)$ amortized for bottom trees $\leq lg n$
   - maintain bitvector of which nonbranching paths have cut
   - preprocess mask for ancestors of each node (1 word)
   - order bits by depth $\Rightarrow$ query = mask+shift+LSB+path
   - use path structure on each nonbranching path

→ NONRECURSIVE ART DECOMPOSITION