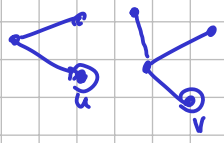
Dynamic Connectivity



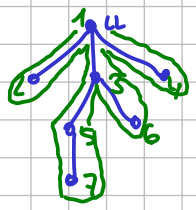
Union-Find with deletions

op: Ins/Del vertex/edge
connectivity queries

goal: $O(\log^c n)$ poly-logarithmic amortized time per op.

For forests:

ET sequences: rooted tree



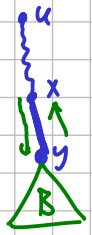
DFS
record vertices visited

1213575363141

length $= 2m+1 = 2n-1 \in \Theta(n)$

Eulerian Tour in the tree with edges doubled

Operations: ① Cut (delete edge) xy, x is parent of y



$uAxyBy\!\!\not{x}Cu$ → $uAxCu$, yBy

$uAxy\!\!\not{x}Cu$ → $uAxCu$, y

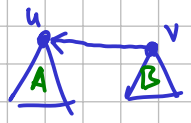
$O(1)$

② Re-rooting

$uAvBu \to vBuAv$

Cutting & pasting sequences

③ Link (joining 2 trees at the roots)

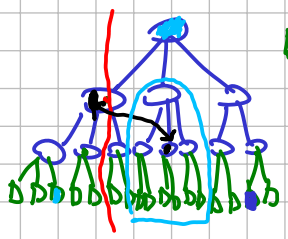


$uAu, vBv \to uAuvBvu$

and adding & removing elements

Sequence → (a,b)-tree (assume constant a,b)

$O(b \cdot \log n)$



keys in internal nodes ~ spaces between ~ edges
vertices in seq.

external nodes ~ vertices of the sequence

$O(\log n)$ time:
Insert, Delete, Cut, Join
↓
operations of Sequences

↳ we also keep:

for every vertex: one occurrence in the seq. "active"
↳ pointer to ext. node

for every edge: both occurrences in the seq
↳ pointer to int. node & key for 1 occurrence
pairing of int. keys

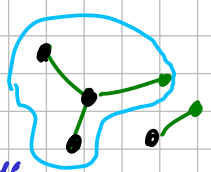
↳ ET-tree handles: Cut, Re-root, Link, FindRoot in $O(\log n)$ time
↳ Connectivity for forest

Augmenting ET-trees to store extra information attached to vertices

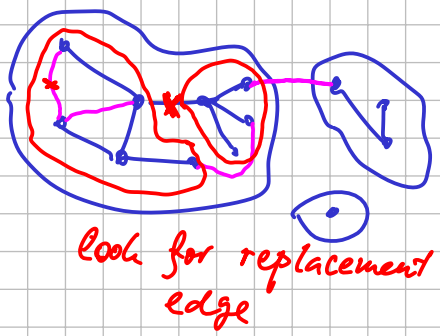
Example: coloring vertices black / white
store color in active occurrence (ext. node)
in internal nodes: keep # of black ext. nodes in the subtree

In $O(\log n)$ time: we can: change color of vertex, count black vertices in a tree, enumerate black vertices

Beware:
- original tree
- ET-tree (nodes)



General graphs: maintain a spanning forest
↳ Holm, de Lichtenberg, Thorup 2001


look for replacement edge

We maintain: F ... a spanning forest
$\ell : E \to \{0 \dots L\}$ levels of edges
$L := \lfloor \log n \rfloor$
F_i := subforest of F with edges of level $\geq i$

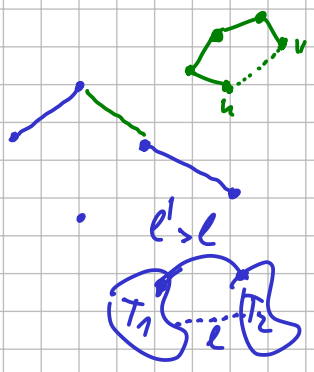
$$F = F_0 \supseteq F_1 \supseteq F_2 \supseteq \dots \supseteq F_L$$

Non-tree edges: stored in ext. nodes of the ET-tree
↳ Insert/Delete nontree
enumerate nontree
in $O(\log n)$

(I1) F is a max. spanning tree wrt. levels
or: if uv is a non-tree edge of level ℓ, then u, v are connected in F_ℓ.

(I2) $\forall i$ components of F_i have at most $\lfloor n/2^i \rfloor$ vertices.

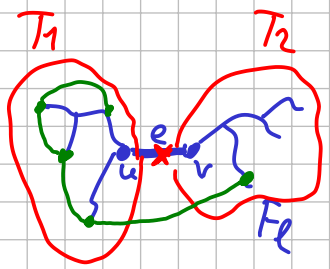
At the beginning: no edges $\Rightarrow$ (I1) & (I2) are satisfied
New edges get level 0.



Deleting a tree edge of level ℓ: $e = uv$

- delete from $F_0 \dots F_\ell$ (higher F_i's not affected)
- due to (I1), replacement edge (if exists) has level $\leq \ell$
- looking for replacement at level ℓ:
  - let T_1, T_2 be trees of F_ℓ containing u, v
  - wlog $|T_1| \leq |T_2|$
  - move level-ℓ tree edges in T_1 to level $\ell+1$
  - enumerate level-ℓ non-tree edges incident with T_1
    - goes to T_1: increase level to $\ell+1$ (I1) & (I2) ok
    - goes to T_2: Success → insert this edge as tree edge (remove as non-tree) in $F_\ell, F_{\ell-1}, \dots, F_0$
  - if no replacement found at level ℓ: move to level $\ell-1, \ell-2, \dots$
  - if it failed even at level 0: we deleted a bridge (no repl. exists)

$O(\log n)$ per level
$O(\log^2 n)$ for all level
$O(\log n)$ per increase of level of edge
↳ $O(\log n)$ increases per edge
$O(\log n)$ per F_i
$O(\log^2 n)$ together
$O(\log^2 n)$ per edge



Augment ET-trees to: ① count ext. nodes
② finding level ℓ tree edges: black/white keys in int. nodes
③ finding level ℓ non-tree edges: remember just level ℓ n.t.-edges in ET-tree for F_ℓ

$O(\log n)$ per op. or edge enumerated

Amortized analysis: Insert, Delete take $O(\log^2 n)$ time.
Find (just ask F_0) $O(\log n)$ time. ← w.c.

trick to speed up Find: represent F_0 using (a,b)-tree with $a, b \sim \log n$

Ins, Del $O(\log^2 n)$ amort

Find $O\left(\frac{\log n}{\log \log n}\right)$

→ height of ET-tree: $\frac{\log n}{\log \log n}$

→ updates of ET-tree $O(\log^2 n)$

→ queries $O\left(\frac{\log n}{\log \log n}\right)$

Lower bound: $\Omega(\log n)$