DECOMPOSITIONS AND EFFICIENT ALGORITHMS

* BRANCH-WIDTH of f over E

def. The branch-width of a graph G is the branch-width of the connectivity f λ_G over $E(G)$. $\lambda_G(X)$ = # verts incident both with X and $E \setminus X$

Def. The rank-width of a graph G is the branch-width of the cut-rank f. $\rho_G(X)$ over $V(G)$

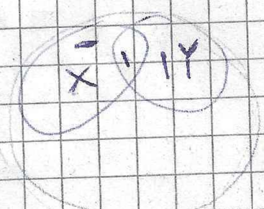
$$\rho_G(G) = \text{rank}\left(X \begin{array}{c} V\setminus X \\ \boxed{1\ 0\ 0\ 1\ \overset{0}{1}} \end{array}\right) \text{ over } GF(2)$$

LEMMA: λ_G is submodular, $\lambda_G(X \cap Y) + \lambda_G(X \cup Y) \leq \lambda_G(X) + \lambda_G(Y)$

PROOF: $A = X \cap Y$, $B = E \setminus (X \cup Y)$, $C = Y \setminus X$, $D = X \setminus Y$.

	Y	$\bar{Y}$
X	A	D
$\bar{X}$	C	B

// E(G)



Which vertices count on the left?

- twice: those having edge in A and another in B
  → count twice also on the right
- once: say, a vertex with an edge in A and other in $C \cup D$ → counts on the right, too

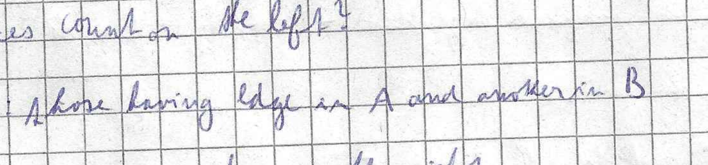
LEMMA: If G has tree-width t and branch-width b, then

$$b \leq t + 1 \leq \frac{3}{2} b$$

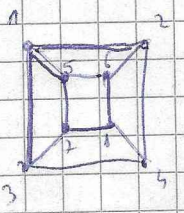
trees (not stars), a cube

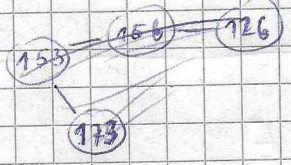
K_n

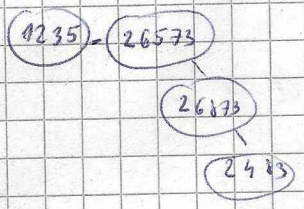
Proof: turning one decomposition into the other

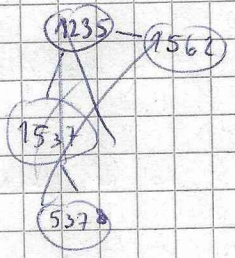


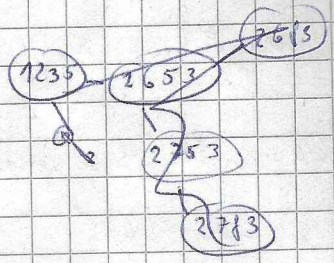
CLAIM: Branch-width of G = 1 iff G ≅ forest of stars











META-DEF: PARSE TREE of a branch decomp of f over E:

— subdivide some edge of the decomp. tree T, make this the root r,

→ get a rooted ordered tree (T, r)

— assign information to the nodes of (T, r):

→ the leaves hold the (edges/vertices) element of E,

→ the internal nodes keep information on how to compose the left and right subtrees together — usually finite info. for bounded branch width

— "boundaried" substructure of E" - the latter which is built up in the tree nodes

DEF. Parse tree of a branch-dec of a graph G is as follows:

* k-boundaried graph $\bar{H} = \cancel{H, \beta}$ (H, β) where $\beta : \{1, \dots, k\} \to V(H)$ ($\in k$) (↑ injective)

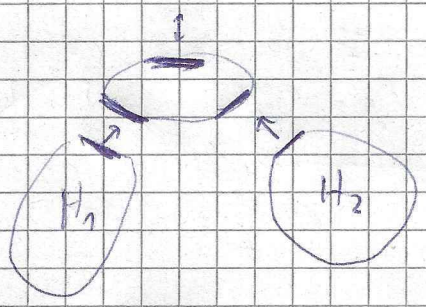
* the leaves keep the edges of G (with lab. 1, 2)

* the internal nodes keep the boundary composition operators, formally a triple $(\beta_1, \beta_2, \beta_3)$ of $\beta_i : \{1, 2, \dots, k_i\} \to N$, $k_i \leq k$.

* recursive parsing of G goes follows:

$\bar{H}_1, \bar{H}_2$ the boundaried subgraphs (of G) parsed by left, right subtrees,

are composed to match with the comp. operation at the node:



FORMAL-LANGUAGE VIEW

A parse tree of G $\leftrightarrow$ a tree word over the alphabet Σ

Σ = composition operators $\cup$ elements of E

$\Rightarrow$ can be handled by a finite tree ~~aut~~ automaton (if tw is bounded)

META-DEF: Consider a particular kind of parse trees, sorts of graphs $\bar{G}$, and the unique associated join operator $\otimes$ over bounded graphs, i.e. $\bar{G}_1 \otimes \bar{G}_2$, say merging the boundaries of $\bar{G}_1, \bar{G}_2$ together and produce a normal graph. Let P be a decision problem, i.e. asking "$G \models P$"?

The canonical equivalence of P on $\bar{\mathcal{U}}_k$ = the universe of all $\leq k$-bounderied graphs is $\bar{G}_1 \equiv_{P,k} \bar{G}_2$ $(\in \bar{\mathcal{U}}_k)$ iff

$$\forall \bar{H} \in \bar{\mathcal{U}}_k : \bar{G}_1 \otimes \bar{H} \models P \Leftrightarrow \bar{G}_2 \otimes \bar{H} \models P$$