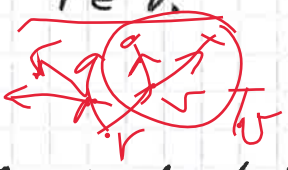
8 Treedecomposition, tree-width and NP-hard problems

8.1 Motivation, an example

Consider the maximum (weight) independent set problem on trees (MWIS)

MWIS
Input : $G=(V,E)$, $w: V \to \mathbb{R}$ (weighted version otherwise $w \equiv 1$)
Output : Independent set $S \subseteq V$, i.e. $E(G[S]) = \emptyset$, which maximizes $w(S) := \sum_{v \in S} w(v)$.

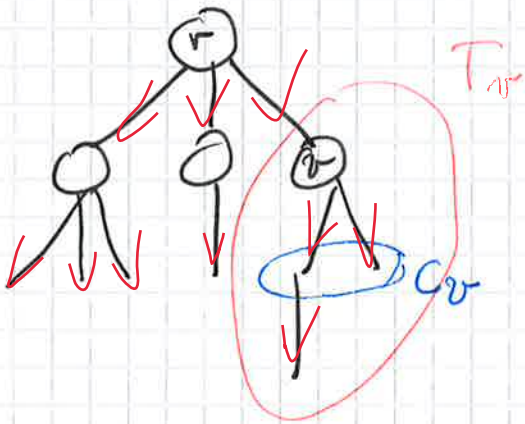
It is an NP-hard Problem in general, but solvable by dynamic programming if G is a tree.
Consider a graph $G=(V,E)$ which is a tree.
Root the tree at an arbitrary vertex $r \in V$, and orient all edges away from r



The subtree rooted at some vertex v, is denoted by T_v, includes v and all vertices reachable from v under the orientation of edges as above.

Then $T_r = G$.

The children of v, denoted by C_v, are all vertices $u \in V$ with $\{v,u\} \in E$ oriented from v to u, thus with a directed edge (or arc) (v,u).
The leaves of the tree have no children.

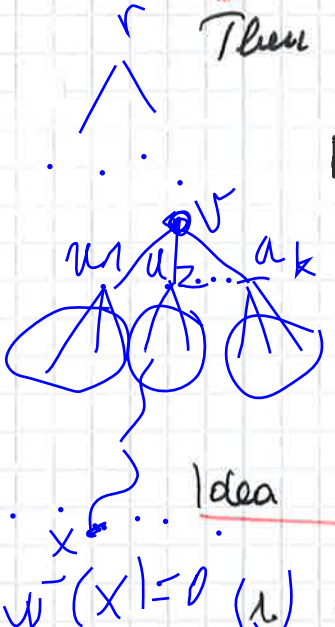


Let $W[v]$ denote the max weight of an independent set of T_v. Our goal is to compute $W[r]$. (At $S[v]$ be the set with weight $W[v]$ I don't know? $v \in S[v], v \notin S[v]$

Let $W^+[v]$ denote the max weight of an independent set in T_v which contains v;

Let $W^-[v]$ denote the max weight of an independent set in T_v which does not contain v.

$W[v] = \max\{W^+(v), W^-(v)\}$



Then we have

$$W^+[v] = w(v) + \sum_{u \in C_v} W^-(u) \quad \text{and}$$

$$W^-[v] = \sum_{u \in C_v} \max\{W^-[u], W^+[u]\}$$

$\sum_{v \in V(G)} O(\deg(v)) = O(|E(G)|) = O(|V(G)|)$

$W^-(x) = 0$ (1.) $W^+(x) = w(x)$

Idea of a dynamic programming algorithm (DPA):

1. select a vertex v such that for all of its children $u \in C_v$, $W^-[u]$ and $W^+[u]$ have already been computed.

2. Compute $W^-[v]$ and $W^+[v]$ using the recursive equations above.

Since every vertex is visited once and applying the recursive relations takes $O(1)$ time (on average) the overall runtime is $O(|V|)$.

So although MWIS is NP hard in general graphs, it is solvable in linear time ($O(N)$) if the input graph is a tree.

8.2: Well-quasi-orderings

Def: A reflexive and transitive relation is called a quasi-ordering. A quasi-ordering $\leq$ on X is called a well-quasi-ordering (WQO), and the elements of X are well-quasi-ordered by $\leq$, if for every infinite sequence $x_0, x_1, \ldots$ in X there are indices i, j with $i < j$ such that $x_i \leq x_j$. Then (x_i, x_j) is a good pair of this sequence. A sequence containing a good pair is a good sequence. Thus a quasi-ordering on X is WQO iff every infinite sequence is good. An infinite sequence is bad if it is not good.

Proposition 8.1 A quasi-ordering $\leq$ on X is WQO iff X contains neither an infinite antichain nor an infinite strictly decreasing sequence.

$x_1 \geq x_2 \geq \quad \geq x_n \geq$ (with $\neq$ between consecutive terms)

Corollary 8.2 If X is well-quasi-ordered, then every infinite sequence has an infinite increasing subsequence $x_1\ x_2 \ldots$ ($x_i \leq x_j \leq \ldots$)

Example $X = \mathbb{Z}$, $\leq$ is the usual "smaller than or equal to" relation ($1 \leq 10$)

$\leq$ is a quasi-order (because it is a total order)

but not a WQO because the sequence

$-1, -2, -3, \ldots, -n, -(n+1), \ldots$

is a bad sequence