- the takeaway for today is for you to know that there is a theory on how to show that a problem does not have an FPT alg
- as with classical complexity, the tool to do that is

1. TAKE A HARD PROBLEM

2. REDUCE IT TO YOUR PROBLEM

- the keywords you need to know:
  - parameterized reductions
  - canonical hard problems (CLIQUE)
  - what the analogue of NP-h to P is
    - W[1]-hard problems do not admit FPT algs (under standard complexity assumptions)
  - using Exponential Time Hypothesis, we can give runtime lower bounds on W[1]-h problems
    - e.g. ETH $\Rightarrow$ no $f(k)\, n^{o(k)}$ alg for k-CLIQUE
  - humans know how to show that a problem w/ an FPT alg does not have a poly kernel

PARAMETERIZED REDUCTION

Def: algo $\mathcal{A}$ is a parameterized reduction from problem (Π, p) to (S, q) if given an instance (I, k) of (Π, p) $\mathcal{A}$ produces an instance (J, ℓ) of (S, q) s.t.

1. I is a yes-instance iff J is a yes-instance
2. $\ell \leq g(k)$ for some computable fct g
3. $\mathcal{A}$ runs in time $f(k)\,\mathrm{poly}(|I|)$ for some computable fct f

---

standard stuff apply

- if A is FPT and we can param reduce A to $B \Rightarrow B$ is FPT
- if $A \to B$ & $B \to C \Rightarrow A \to C$

- in general poly reductions & FPT reductions are incomparable
- but 99% of reductions in the wild in FPT field are poly reductions

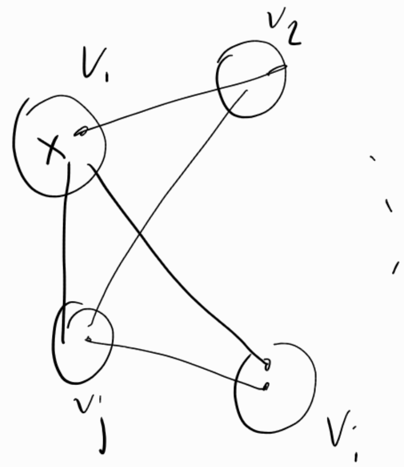
Typical hard problems

. CLIQUE / IS

MULTICOLORED CLIQUE

<u>In</u>: gr G w/ vertex partition $V_1, \dots, V_k$ s.t. each V_i is an IS

<u>Out</u>: Does G have a k-clique?



. if a clique exists, it has exactly 1 vertex in each V_i

<u>Reduction</u>: from CLIQUE

. for each $v \to v^1, \dots, v^k$ ($v^1 \in V_1$, $v^k \in V_k$)

. if $uv \in E(G)$, make $u_i v_j$ adj for every $1 \le i, j \le k$, $i \ne j$)

<u>=></u>:

. suppose $u_1, \dots, u_k \in V(G)$ is a clique

. is $u_1^1, u_2^2, \dots, u_k^k$ a clique?

. ... yes

<= . let $u_{i_1}^1, u_{i_2}^2, \dots u_{i_k}^k$ be a clique

. <u>claim</u>: $u_{i_1}, u_{i_2}, \dots, u_{i_k}$ is a clique in G

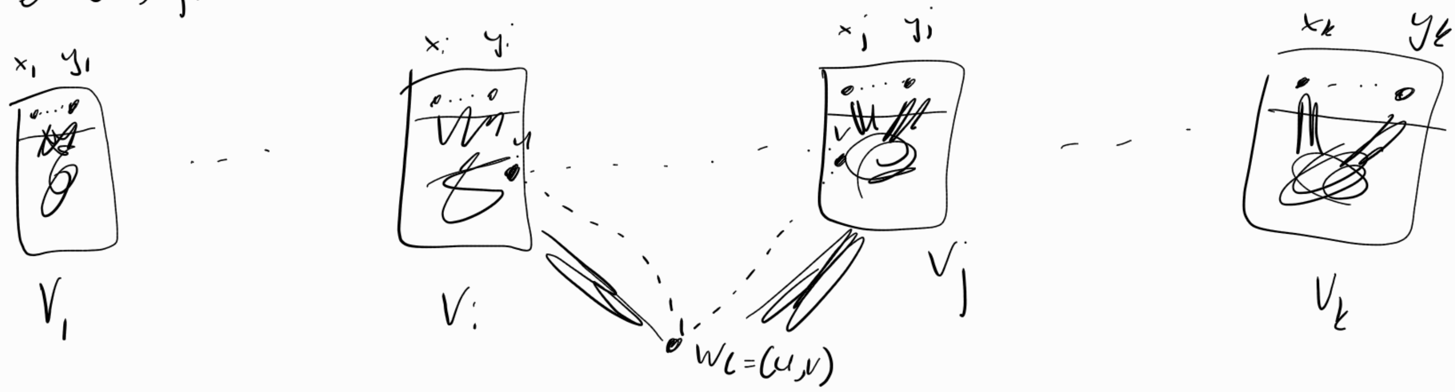
MULTICOLORED CLIQUE is good for starting param reductions because there

. MULTICOLORED IS exists too (V_i's are cliques)

MIS $\rightarrow$ DS

$G = ((V_1, \dots, V_k), E) \longrightarrow G' = (V', E'), k'$

- for each $v \in G$ add v to G'
- make V_i in G' a clique
- for each V_i, add x_i, y_i to $V(G')$ and make them complete to $V_i \subseteq V'$ each
- $e = (u_i, v_j) \in E \rightarrow$ add w_e to V', make it adj to each $(V_i \cup V_j) \setminus \{u, v\}$



$\Rightarrow$

- suppose $u_1, \dots, u_k$ is an IS where $u_i \in V_i$
- each $V_i \cup \{x_i, y_i\}$ is dominated by u_i
- are all w_e's dominated?
  - suppose not, then both endpoints $(u, v) = e$ have to be in the IS
  - but they form an edge, so they can't both be there

$\rightarrow$ all w's are dominated too

$\Leftarrow$

- due to x_i's & y_i's existing, each V_i has to contain a vertex of the solution
- and it's exactly one due to k being k
- let D be the solution
- is it an IS in G?
- suppose not $\rightarrow$ $\exists u_i u_j \in D$ s.t. $u_i u_j \in E(G)$
- but then $w_{u_i u_j}$ is not adj to D since $N(w_{uv}) = V_i \cup V_j \setminus \{u_i, u_j\}$

$\rightarrow D$ is an IS in G

Conceptual difference between NP-h proofs and W[1]-hard proofs

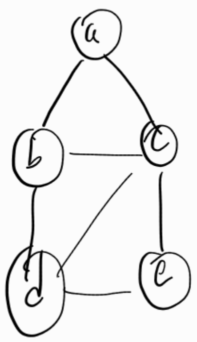
- in NP-h reductions, e.g., from IS to VC, we try to capture the structure of the input by produced output
- in W[1]-h reductions, in addition to the structure, we try to capture the structure of the solution as well
  - i.e. there has to be gadgets for "picking a vertex into the solution"

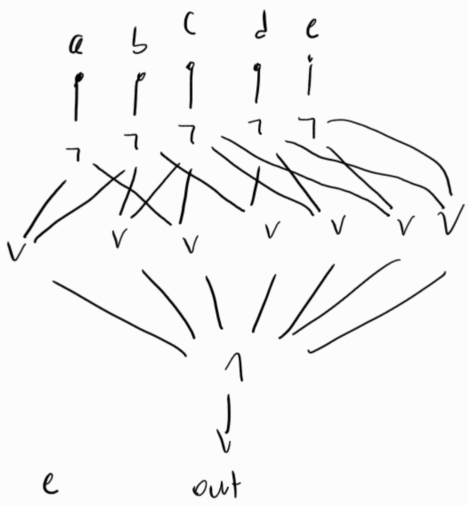
W-hierarchy

WEIGHTED CKT SAT

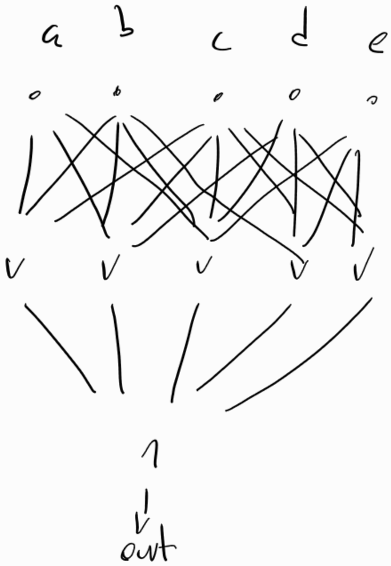
- given a boolean ckt C (unbounded fan-in last), $k \in \mathbb{N}$, w/ one output gate, find an assignment w/ $\leq k$ true's that sats C.

- a gate is large if it has indegree >2
- the weft of a ckt is the maximum # large gates on any input $\to$ output path

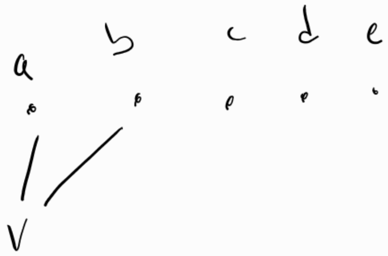




IS



DS



- a problem belongs to W[k] if it can be solved using a ckt family of weft $\geq k$
- IS $\in$ W[1], DS $\in$ W[2]
- OPEN DS $\in$ W[1] ? (leads to collapse of W-hierarchy)

ETH lower bounds

ETH: 3 SAT cannot be solved in $2^{o(n)}$

- alternatively there exists a constant c s.t. no deterministic alg solving 3-SAT runs in time 2^{cn}

⇝ leads to runtime lower bounds?

- Sparsification lemma: There exists constants c_1, c_2 s.t. any 3-SAT instance φ can be converted into $\bigvee_{i=1}^{2^{c_1 n}} \varphi_i$ where φ_i is a 3-SAT fla with n vars and $c_2 n$ clauses

→ we can assume that the starting fla has $O(n)$ clauses ← $\binom{n}{3}$ clauses

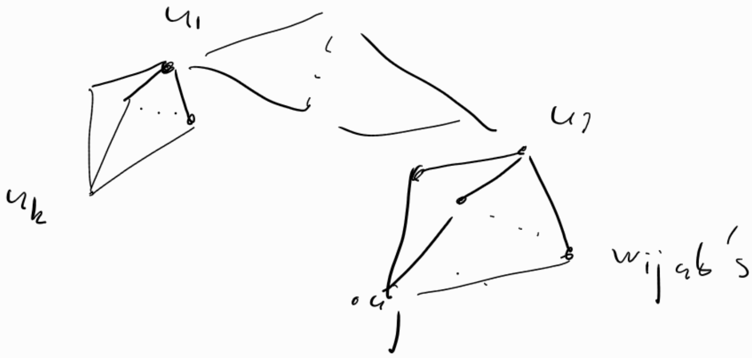
- o/w lower bounds could be in the form $2^{\Omega(n^{\frac{1}{3}})}$

→ it can be proven that there's no $f(k) n^{o(k)}$ alg for CLIQUE

LIST COLORING

- is W[1]-hard on low tw grs
- reduce from MIS
- for each V_i add a vertex u_i, $L(u_i) = V_i$
- for each edge $ab \in V_i \times V_j$ $(1 \leq i < j \leq k)$, add vertex w_{ijab} w/

$L(w_{ijab}) = \{a,b\}$ & add edges $u_i w_{ijab}$ $u_j w_{ijab}$



- u_i's are a VC

$\to$ low tw

$\Rightarrow$

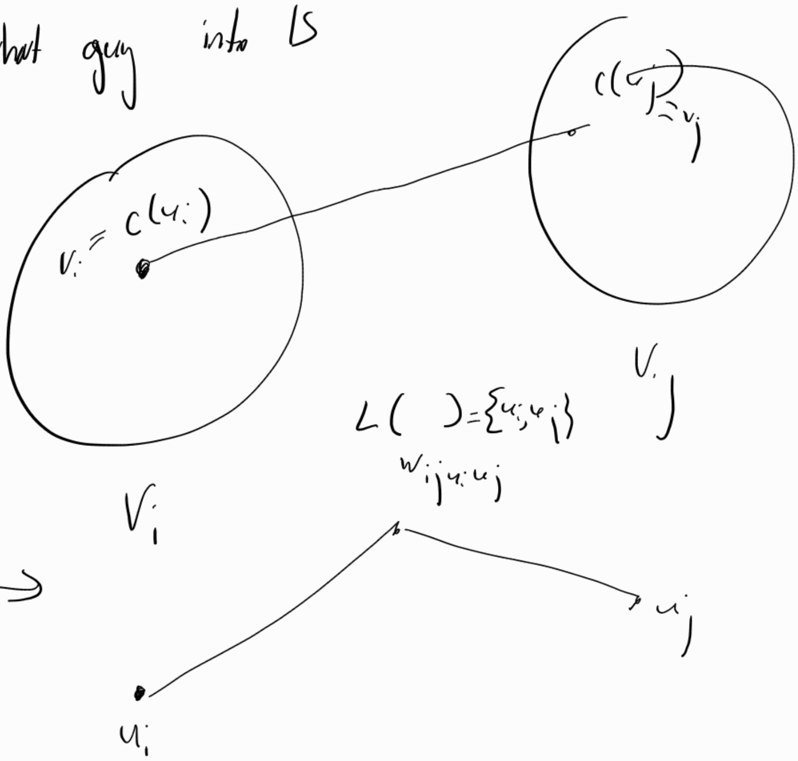
- assume $v_1, \dots, v_k$ is an IS
- set color of u_i to v_i
- look at any edge $ab \in V_i \times V_j$
- both endpts can't be in IS $\Rightarrow$ at least one of a or b is not used by u_i, u_j

$\Leftarrow$

- suppose we have a list coloring c
- look at $c(u_i)$ and pick that guy into IS
- suppose it is not an IS
- if we have a list coloring,

then $\longrightarrow$

can't happen

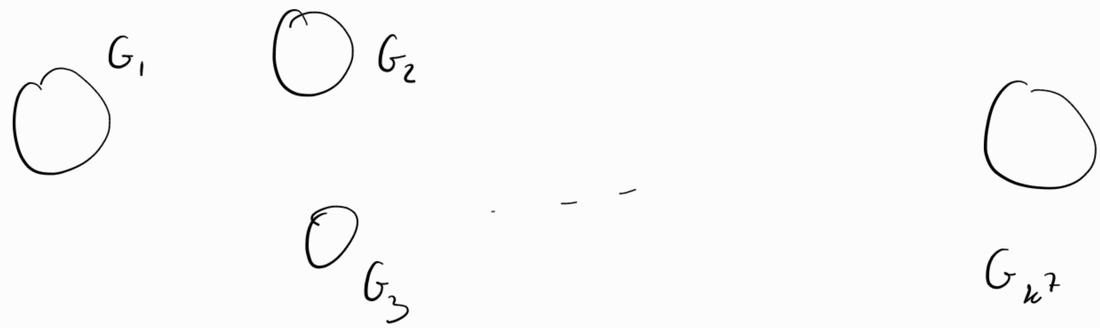


KERNELIZATION LOWER BOUNDS

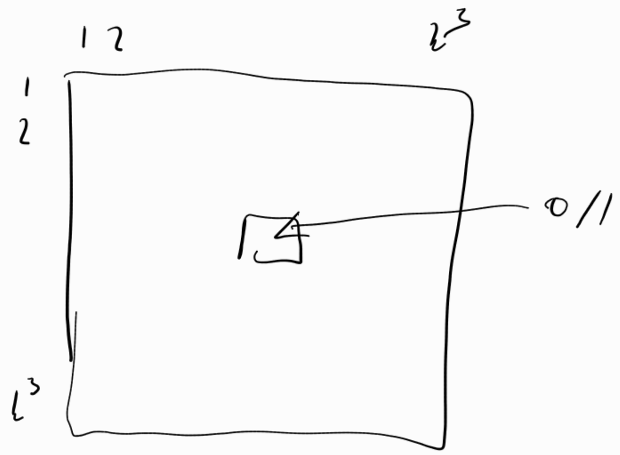
- you know from the first lecture that a param problem Π has an FPT alg $\iff$ Π has a kernel
- but this kernel might be exponentially (or worse) sized
- the holy grail are poly kernels
- how can we prove that a problem which admits an FPT alg does not admit poly sized kernels?

Informal example: LONGEST PATH

- suppose there exists a k^3 kernel for LONGEST PATH
  - meaning that given (G, k) where $|V(G)|, |E(G)|$ can be arbitrary, there is a poly alg which returns a LONGEST PATH instance w/ $\leq k^3$ vertices
- synthesize an instance $I = (G, k)$ of LONGEST PATH which consists of k^7 different graphs that are pairwise disconn

$G =$ 

- i.e. G is k^7 different graphs
- observe : G has a path of len $k \iff \exists i \in [k^7] : G_i$ has a path of len k (*)
- apply k^3 kernel on $G \longrightarrow$ receive a graph G' w/ k^3 vertices
- how much information can G' give me?
- G' can be represented by an adj matrix



- and we need to somehow name vertices : $\lg k^3 \rightarrow O(\lg k)$ labels
- in total we can get $k^6 \lg k$ bits of information
- that is strange, because the input instance G consists of k^7 distinct instances
- and by (*), what we want to know is which G_i has the long path

- but we get $k^6 \lg k \ll k^7$ bits of info $\Rightarrow$ we can now talk about $\leq k^6 \lg k$ out of k^7 instances

$\rightarrow$ somehow the poly alg for kernelization was able to determine $\frac{k}{\lg k}$ grs that do not contain a path of len k, which is an NP-h problem

$\rightarrow$ this is evidence that LONGEST PATH does not have a poly kernel

- and the complexity assumption here is that LONGEST PATH does not have a poly kernel, unless $coNP \subseteq NP/poly$