Cpts 515. 11/18/2020

Two famous problems that we don't whether they are NP-complete or not:

(1). GI. (Graph Isomorphism)

(2). FACT. (large number factorization).

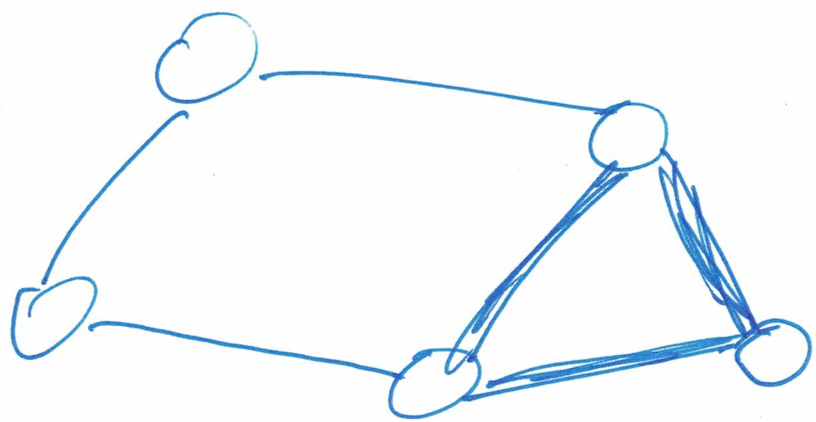
Given: n

Question: $\exists\ p, q$ s.t. $n = p \cdot q$ and p, q are primes?

For some NP-complete problems, they are tractable (i.e., in P) if some parameters are fixed (or small).

Special case of $\underbrace{\text{subgraph-isomorphism}}_{\text{is NP-complete.}}$:

Finding whether a graph contains a triangle.



// directed graph
// or undirected graph,
// it doesn't matter.

naive alg: Enumerate all nodes i, j, k and check $i \to j$, $j \to k$, $k \to i$ is an edge.

$O(|V|^3)$.

$\llcorner$ # of nodes in G.

A better alg (to check existence of a triangle).

let A be the adjacent matrix of G.

// so, $A[i,j] = 1$ if $i \to j$ is an edge

// $A[i,j] = 0$ if o.w.

G has a triangle iff $\exists\, i, j, k$ s.t.

$A[i,j] \neq 0$ and $A[i,k] \neq 0$ and

$A[k,j] \neq 0$

iff $\exists\, i, j$ s.t. $\underbrace{A[i,j] \neq 0 \ \& \ A^2[i,j] \neq 0}_{\text{for undirected graph.}}$.

which runs in $O(n^{2.38})$ for $A \cdot A$.

(Coppersmith alg).

Reasoning: $A^2[i,j] \neq 0$ iff, by def,

$$\sum_k A[i,k] \cdot A[k,j] \neq 0 \text{ iff}$$

$$\exists k,\ A[i,k] \neq 0 \ \& \ A[k,j] \neq 0.$$

Fixed Parameter Tractability:

Consider an NP-complete problem X. Assume that, in addition to the problem size n, we have a parameter k. if an alg for X can run in time

$$\underbrace{2^{O(k)}}_{\text{exp.}} \cdot n^{O(1)}$$

// exp in k, but polynomial in n.

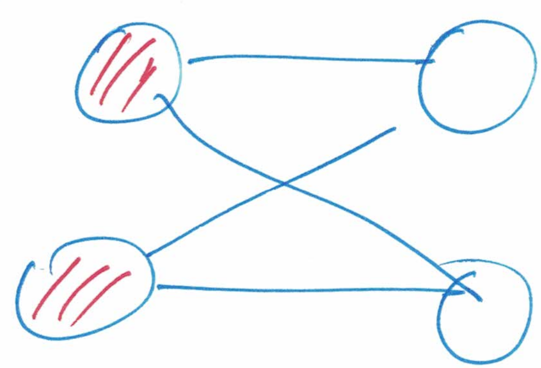
Then we say that the Π is fixed parameter tractable wrt the k.

Example. VertexCover (VC) is the following problem

Given: a graph $G = (V, E)$ and a number k

Question: Is there a vertex cover $C \subseteq V$ s.t. $|C| = k$ and each edge $(u, v) \in E$, at least one of u and v is in C?

Example graph:



$k = 2$

Thm. VC is fixed parameter tractable wrt the k.

Bad idea. Usually, such a proof starts with brute force on the part concerning the k.

Enumerate all $C \subseteq V$ with $|C| = k$,

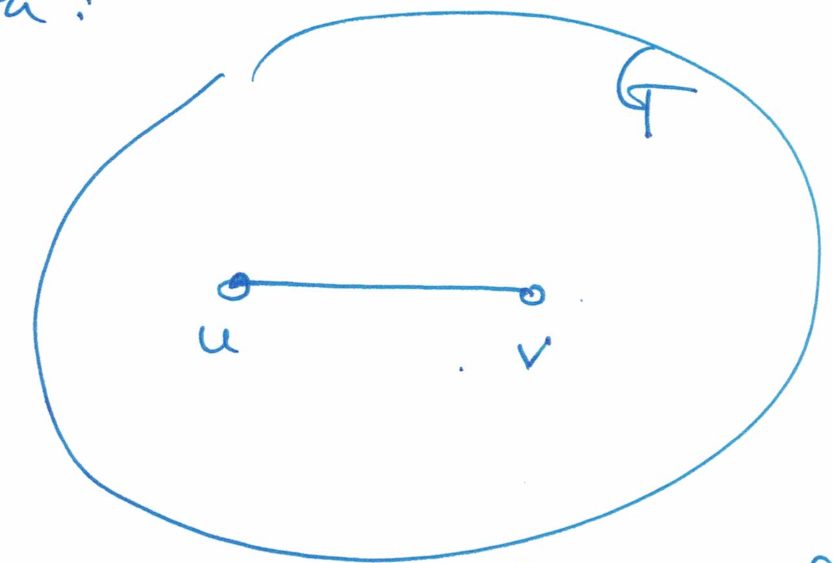
and check whether C is a VC or not.

Running time is $\binom{n}{k}$ where n is the # of nodes in G, which is roughly $O(n^k)$. Doesn't work!

But I want

$$\text{exp on } k \cdot n^{O(1)}.$$

Good idea:



Let C be a VC of size k. Then, for an edge $u \to v$, at least one of u and v must be in C.

Assume that $v \in C$. Then, we can show easily:

$$C - \{v\} \text{ is a VC of } G - \{v\}.$$

And the contrary is also true. That is, Given $v \in C$, if $C - \{v\}$ is a VC of $G - \{v\}$. Then, C is a VC of G.

Alg: Boolean VC(G, k){ // check whether
// G has VC of size k

if G has no edges, return true

take an arbitrary edge (u,v) in G; // This is <u>NOT</u>
// enumeration.

Try VC(G−u, k−1) and
VC(G−v, k−1)

if one of the two returns true,
then ret true;

else return false.

}

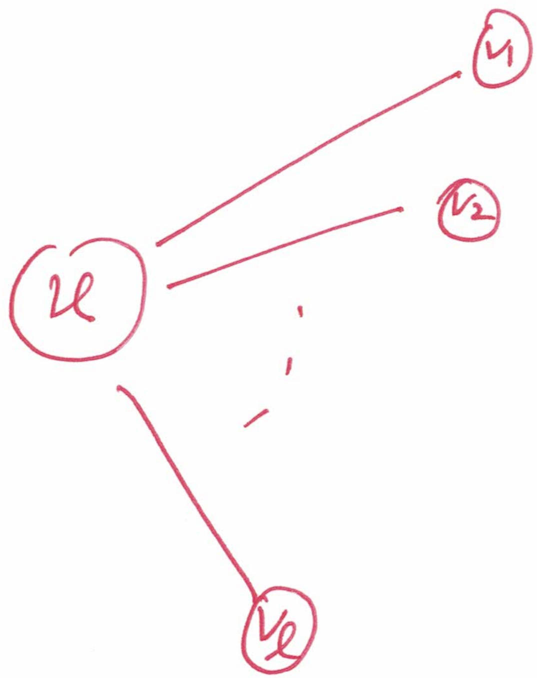
Time complexity: each G−u construction takes time $O(n)$, since we need drop edges; recursive tree size $= O(2^k)$. So, time is

$O(2^k \cdot n)$.

exp in k. polynomial in n

A better idea.

Observation:



If u has $\ell \geq k+1$ neighbors, then u must be in the VC. Recall the VC size $= k$. Why? Otherwise, all $v_1, v_2, \ldots, v_\ell$ must be in the VC. Then the VC size $\geq \ell \geq k+1$. $\times$

Hence, we use $\mathcal{K}$ to denote the set of all nodes u s.t. u has $\geq k+1$ neighbors. Then, $G - \mathcal{K}$ is called "kernel". We can show,

G has VC of size k iff

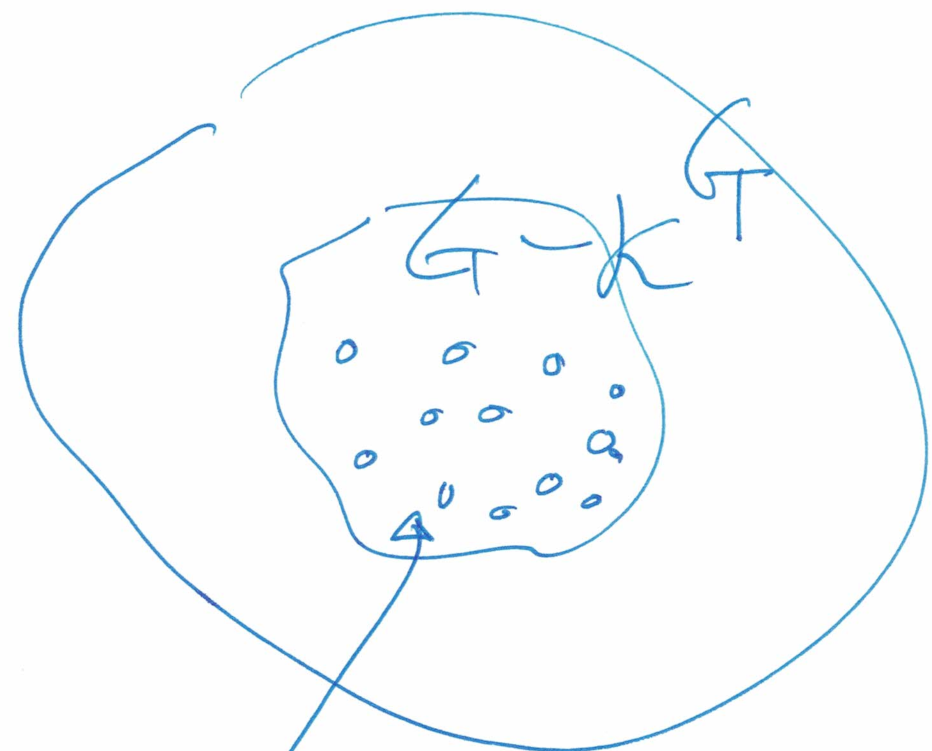
$G - \mathcal{K}$ has VC of size $k - |\mathcal{K}|$.

Key observation: What's the "size" of $G - \mathcal{K}$?

$G - \mathcal{K}$ has at most k^2 edges!!!

Why? each node in $G - \mathcal{K}$ has at most k neighbors.

if it has more than k^2 edges, then it can't have a VC with size up to k since any node only "covers" at most k edges.



Each such node has at most k neighbors. There are at most k nodes that can be chosen so that all the edges in $G-K$ are covered.

for each node, you have $\leq k$ edges.

Alg: Construct $G - \mathcal{K}$.

brute-force on $G - \mathcal{K}$ to find a VC of size $k - |\mathcal{K}|$.

this graph $G - \mathcal{K}$ has at most k^2 edges.

So: The running time <u>only</u> dep. on k.

In real world, many problems are "functions".

input instance

$\downarrow$

output a number.

Can you define a notion on these functions that is similar to NP-completeness? Yes.

#P-problems.

Set-up. Let $f : \Sigma^* \longrightarrow \mathbb{N}$ be a function.
f is in #P if there is a poly-time nondet. Turing machine M such that, $\forall x \in \Sigma^*$,

$f(x) =$ the # of accepting runs of M on input x.

"Better" definition that is equivalent:

Consider a Predicate $Q(x, y)$:

$$\Sigma^* \times \Sigma^* \longrightarrow \{\text{true}, \text{false}\},$$

Such that there is a polynomial $p(\cdot)$ s.t.:

$\forall x, y,$

$$Q(x, y) \Rightarrow |y| \leq p(|x|).$$

// for input x, the "output" y can't be
// too long.
//

Such Q is called an NP-predicate when Q is in P.

f is in $\#\underline{P}$ iff there is an NP-predicate Q such that $\forall x$,

$$f(x) = |\{y : Q(x,y)\}|.$$

Proof. easy.