CSE 396 Lecture Thu 5/9 Spr 2019

TopHat 5965

Examples of Problems In NP:

SATisfiability: INST: A Boolean formula $\phi(x_1, \dots, x_n)$

e.g. $(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3)$ Bar $\bar{x}_1 \equiv \neg x_1$.

QUESTION: Is there a truth assignment $\vec{a} \in \{0,1\}^n$ such that $\phi(\vec{a}) =$ true? I.e., $\vec{a}$ satisfies ϕ.

In the example, any assignment $a_1 a_2 a_3$ with $a_3 = 1$ works. So do $a = 110$ and 000. But not 010 or 100.

3SAT is the special case where $\phi = C_1 \wedge \dots \wedge C_m$ and each clause C_j is a disjunction of up to three literals x_i or $\bar{x}_i$.

SAT and 3SAT belong to NP. Design a verifier V that takes both $\langle\phi\rangle$ and a as inputs.

$$V(\phi, a) = \begin{cases} 1 & \text{if } \phi(a) = 1 \\ \text{reject} & \text{otherwise.} \end{cases}$$

* We can evaluate a Boolean formula ϕ on one given assignment a. Quickly—in $\tilde{O}(n+m)$ time in the case of 3SAT.

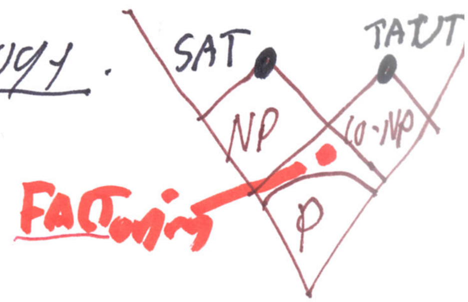
~ means ignore log n or log m factors

An NTM N, given only ϕ, can guess an assignment a that works and then run $V(\phi, a)$ to verify. Then N, too, runs in $O(n+m)$ time, which is linear, hence polynomial, time.

The complement $\overline{SAT}$ is (essentially) $\{\langle\phi\rangle : \phi$ is <u>not</u> satisfiable$\}$

ϕ is not satisfiable $\Leftrightarrow$ $\neg\phi$ is a <u>tautology</u>.

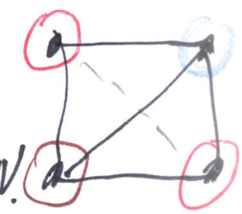
TAUT $= \{\phi' : \phi'$ is a tautology$\}$.



2. <u>Graph 3-coloring</u>: INST $G=(V,E)$

see Ex 7-38

QUES: Can you assign colors R, G, B to each node so that G has no monochromatic edges?



3. FACTORING: INST: A number N and another number $K<N$.
QUES: Does N have a prime factor p st $p<K$?

<u>YES case</u>: Verify by guessing the unique prime factorization of N and seeing if p is part of it and $p<K$.

<u>No</u> case: Ditto! verify <u>no</u> prime in the factorization is $<K$.

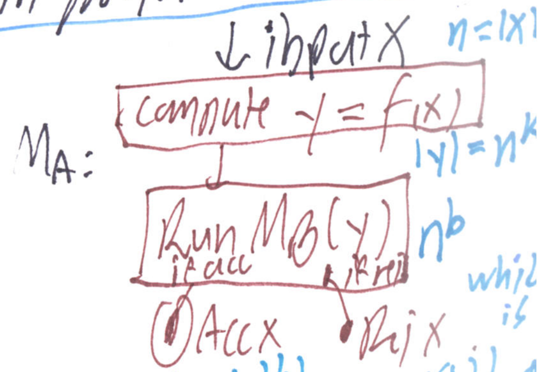
FACT: That $p_1^{b_1}\cdots p_\ell^{b_\ell}=N$ can be verified quickly and that each p_i is prime.

∴ FACTORING is in NP and in coNP.

And if you could solve this language, you could <u>find</u> factors in polynomial time by Binary Search

<u>Defn</u>: A language A <u>mapping-reduces</u> to a language B <u>in polynomial time</u>, written $A \le_m^P B$, if there is a function $f:\Sigma^*\to\Sigma^*$ that is computable <u>in polynomial time</u> s.t.

$\forall x:\ x\in A \Leftrightarrow f(x)\in B$.



<u>Theorem</u>: If $A \le_m^P B$ and $B\in P$, then $A\in P$.

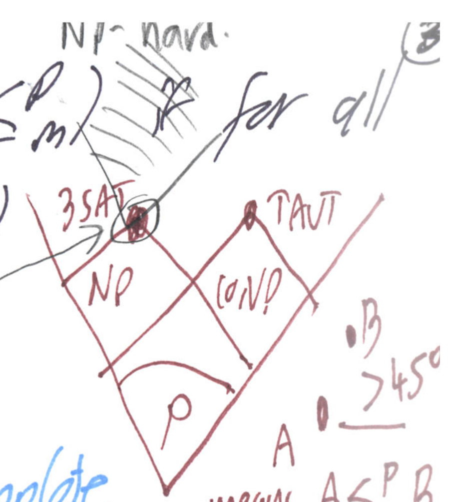
∴ Contrapositive: If $A\le_m^P B$ and $A\notin P$, then $B\notin P$

∴ If A is NP-hard and $A\le_m^P B$, then B is NP-hard!

Time is $O((n^k)^b) = O(n^{kb}))$

Def^n: A language B is NP-hard (under $\le_m^p$) if for all $A \in NP$, $A \le_m^p B$. (under $\le_m^p$ always understood)

If also $B \in NP$, then B is NP-complete!



Stephen Leonid

Cook-Levin Theorem: SAT and 3SAT are NP-complete.

Proof: We've shown (3)SAT $\in$ NP. Let any $A \in NP$ be given. Take a det^c $p(n)$-time verifier V that recognizes $\{\langle x,y\rangle : y$ is a witness for $x \in A\}$.

$x \in A$, $n = |x|$

$\iff \exists y \in \{0,1\}^{p(n)}$ st. V accepts $\langle x,y\rangle$.

$\overset{(\exists y)}{\iff}$ $C_n^{(x,y)}$ accepts xy

$\overset{(\exists y)}{\iff}$ every NAND gate g functions correctly and $w_0 = 1$

By the principle that software can be burned into hardware there is a poly-sized circuit C_n of NAND gates st. $C_n(x,y) = 1 \iff V(x,y) = 1$

And we can build C_n in $n^{O(1)}$ time. $(n+p(n))^{O(1)}$ time

$(u \vee w) \wedge (v \vee w) \wedge (\bar{u} \vee \bar{v} \vee \bar{w})$ is satisfied.

$\therefore$ Make $f(x) = \left(\bigwedge_{\text{NAND gates } g} \phi_g \right) \wedge (w_0) \wedge \underbrace{(x_1) \wedge (\bar{x}_2) \wedge (\bar{x}_3) \wedge (x_4) \wedge \cdots (\bar{x}_n)}_{\text{singleton clauses that set the } x_i \text{ inputs to the actual bits of } x.}$ if $x = (10011\cdots0)$, say

$= \phi_x$

Then $x \in A \iff \exists y\, \phi(x,y) = \text{true}$

$\iff \exists y\, \phi_x(y) = \text{true} \iff \phi_x \in$ the language SAT, indeed 3SAT. ☒

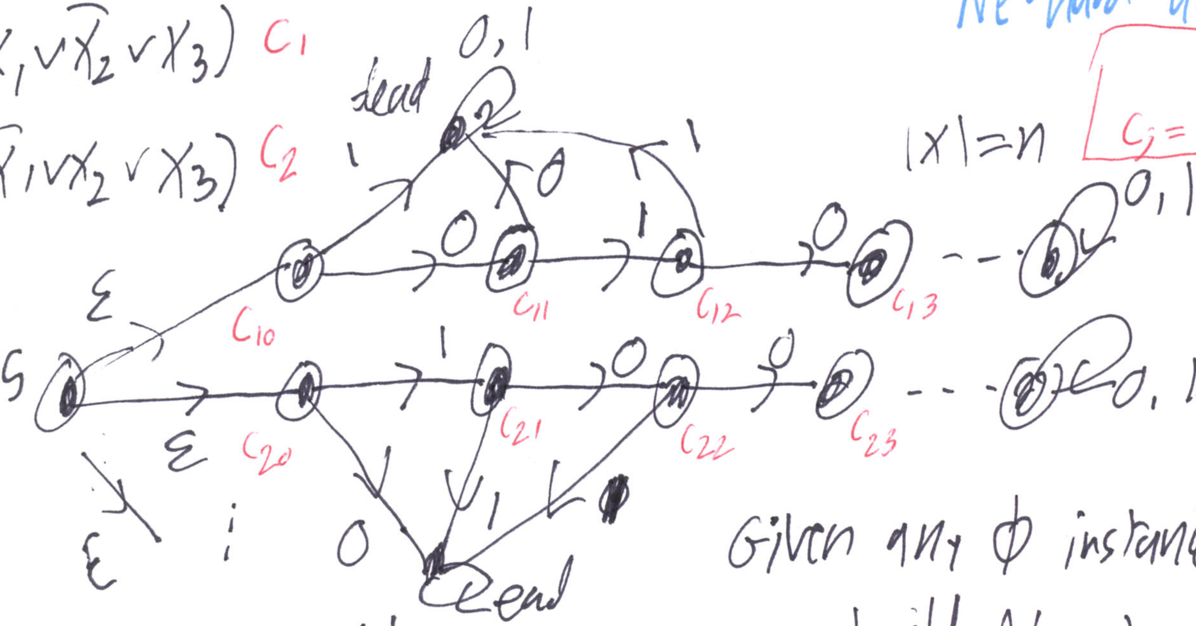
So $A \le_m^p$ (3)SAT via f, where $f(x)$ is computable in $n^{O(1)}$ time. ☒

Furthermore, 3SAT $\le_m^p$ GRAPH 3-COLORING, so G3C is also NP-hard and NP-complete

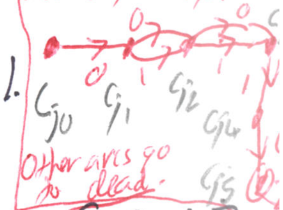
$(x_1 \vee \bar{x}_2 \vee x_3)$ C_1

$\wedge (\bar{x}_1 \vee x_2 \vee x_3)$ C_2

$|x| = n$



For clauses lik $C_j = (x_1 \vee \bar{x}_4 \vee x_5)$ w would use arcs a both 0&1 to go across for absent val

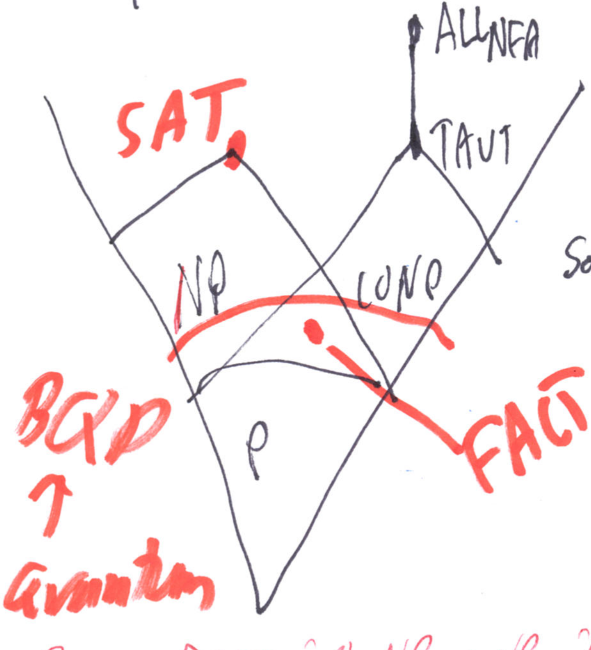


Curmudgeon NFA N_ϕ: branches to a clause and goes dead if the clause gets satisfied.

$L(N_\phi) = \Sigma^* \iff \phi \notin SAT$.

Given any ϕ instance of 3SAT, we can build N_ϕ in $n^{O(1)}$ time.

If $\phi \in SAT$ then there is some $\vec{a}$ that satisfies ϕ, which makes N_ϕ go dead on all branches, so $a \notin L(N_\phi)$ so $L(N_\phi) \notin ALL_{NFA}$. But if $\phi \notin SAT$, then every a makes at least one clause not satisfied, so $a \in L(N_\phi)$.



So TAUT $\le_m^p$ ALL_{NFA}.

We can build N_ϕ for any ϕ in Conjunctive Normal Form (CNF) by this means in linear time because the arcs directly translate the clauses one-by-one.

ADDED: In fact, SAT $\le_m^p$ ALL_{NFA} too, but that is much harder to show.

Because FACT is in NP ∩ coNP, if it were NP-complete then we would get NP = coNP. Not quite as far down as P, but close. Unlike RE ∩ coRE = DEC, "NP ∩ coNP = P" is not believed true. Quantum computers can solve FACT in Bounded-error Quantum Polynomial time (BQP).