TopHat 9816

Theorem: A language A belongs to NP iff there is a Verifier DTM V that runs in polynomial time and a polynomial p s.t.

crucial

$$\text{for all } x: x \in A \iff (\exists y : |y| \leq p(|x|))[V \text{ accepts } \langle x, y \rangle].$$

$n = |x|$

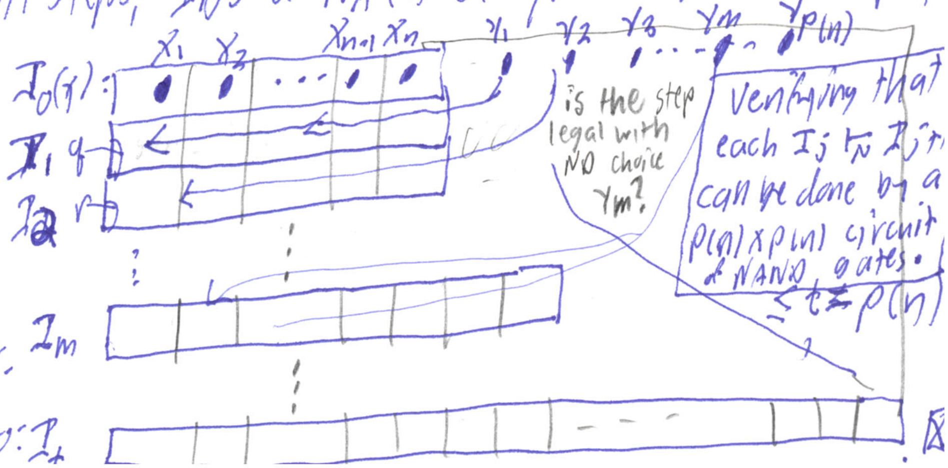
Moreover, the body of V can be either:

- The predicate $T(\langle N_A \rangle, x, \vec{c})$ — applied to a poly-time NTM N_A for A, where whole computations $\vec{c}$ are the "y"s.
- An easy-to-build (given x and $n = |x|$) sequence $[C_n]$ of poly-size circuits of NAND gates and $\overbrace{n}^{x} + \overbrace{p(n)}^{y}$ inputs.

Wlog: $\Sigma = \{0,1\}$, we can demand $|y| = p(n)$, all nondet^c steps by NTMs are binary, and we use single-tape NTMs and DTMs. Building C_n from x and n takes det^c poly(n) time by itself.

Proof: ($\Leftarrow$) Given V in any form above, we can take $p(n)$ to be its polynomial runtime or circuit size. Define an NTM N that on any input x uses (up to) $p(n)$ nondeterministic steps to "guess" y and then deterministically runs $V(x,y)$, accepting x on that run if and only if V accepts $\langle x, y \rangle$. Then N is a poly-time NTM s.t. $L(N) = A$.

($\Rightarrow$) Given $A \in NP$, we can take an NTM N_A that runs in some polynomial time $p(n)$ such that $L(N_A) = A$. Within $p(n)$ steps, IDs of $N_A(x)$ can grow to size at most $p(n)$. Hence accepting computations $\vec{c}$ can be coded by strings $y \in \{0,1\}^*$ of length $q(n) = O(p(n) * p(n))$. So the $T(N_A, x, \vec{c})$ predicate from the last lecture is a poly-time verifier. Moreover we can stack IDs of N like so: I_0



Focal Example of a Problem/Language in NP. (2)

SATisfiability: INST: A Boolean formula $\phi(x_1, \ldots, x_n)$ in variables $x_1, \ldots, x_n$ with logical gates $\wedge, \vee, \neg$.

QUES: Is there an assignment $a_1 \cdots a_n \in \{0,1\}^n$ that satisfies ϕ, ie. $\phi(\vec{a}) = \text{TRUE}$?

$N = |\langle\phi\rangle|$ then $n = o(N)$.

Evaluating a Bool Fmla ϕ is quick

$$SAT = \{\langle\phi\rangle : (\exists \vec{a} \in \{0,1\}^n) : \underbrace{\phi(\vec{a}) = \text{true}}\}$$

$n < |\langle\phi\rangle|$ $p(n) = O(n)$.

$\therefore$ SAT $\in$ NP.

Again $n << |\langle G\rangle|$ but we think of n as "the" size

Example 2: INST: An undir. graph $G = (V, E)$ and an integer $k \leq n = |V|$.

"IND SET"

$\therefore$ INDSET is in NP.

QUES: Does there exist a set $I \subseteq V$, $|I| = k$ ($k \leq n$) st. no two nodes in I have an edge between them?

verify by checking up to $\sim n^2$ edges, in poly(n) time.

universal quantifier — Still poly-time deterministically verifiable

Note: $\phi \in \widetilde{SAT} \Leftrightarrow (\forall \vec{a} \in \{0,1\}^n)\ \phi(\vec{a}) \neq \text{TRUE}$.

$\Leftrightarrow \neg\phi$ is a tautology. Essentially,

TAUT is complementary to SAT, so it is in co-NP $= \{L : \tilde{L} \in NP\}$.

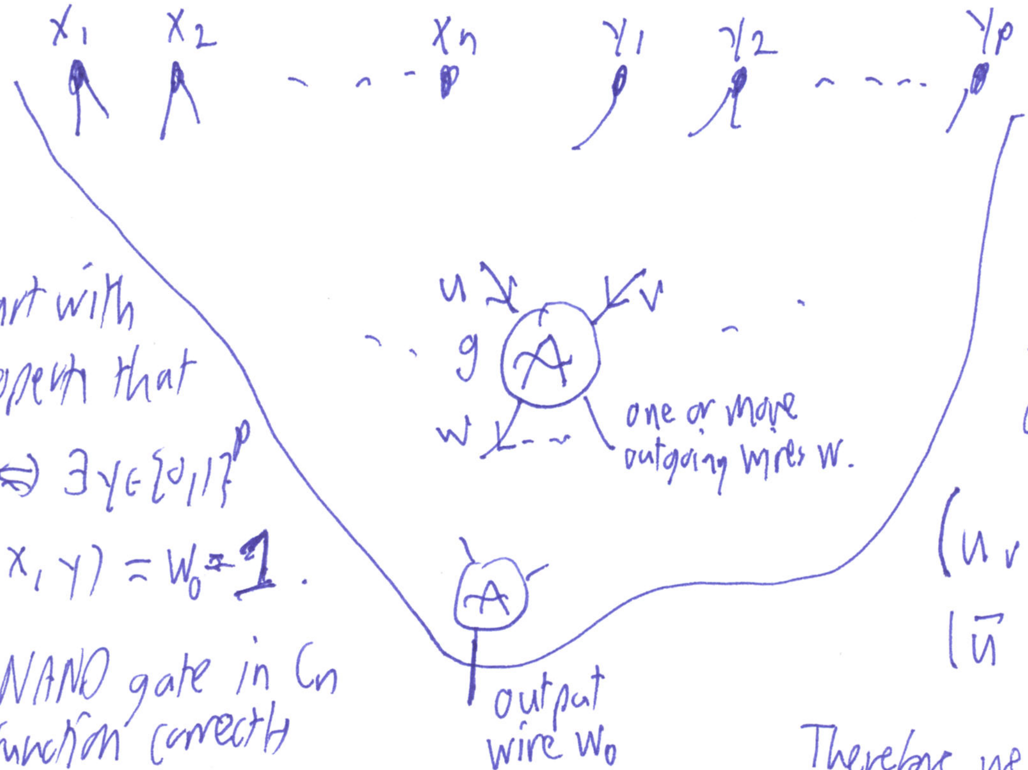
Cook-Levin Theorem: $SAT \in NP$ ✓ and for all $A \in NP$, $A \leq^p_m SAT$

Stephen 1970-1 Leonid 1970-73

SAT is NP-complete.

Proof: Let any $A \in NP$ be given. Take a poly-time NTM N_A s.t. $L(N_A) = A$. Given any x, take $n = |x|$, and compute the circuit C_n of NAND gates for the verifying (x, y), $p = p(n)$.

C_n



We start with the property that $x \in A \iff \exists y \in \{0,1\}^p$ s.t. $C_n(x, y) = w_0 = 1$.

Every NAND gate in C_n must function correctly

A generic NAND gate g with a given output w is correct iff

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

Therefore we can compute a formula ϕ by

- AND-ing together all triples of clauses ϕ_g over all gates g in C_n
- Conjoin the singleton clause (w_0) mandating $w_0 = 1$
- Finally given a particular $x \in \{0,1\}^n$, use n singleton clauses (x_i) or $(\neg x_i)$ to set each bit.

Then ϕ has one variable for each wire or input gate of C_n but C_n has $O(p(n)^2)$ wires and is easy to build, so $f(x) = \phi$ is a polynomial time computable function. And $x \in A \iff$ there is an assignment to $y_1 \cdots y_p$ that induces an assigned value to every wire that satisfies ϕ. Thus $A \leq^p_m SAT$, indeed to 3SAT where ϕ is a conjunction $C_1 \wedge \cdots \wedge C_m$ and each clause C_i has at most 3 literals. ☒

Another NP-complete Problem: $\sim ALL^n_{NFA} \equiv NOTALL^n_{NFA}$

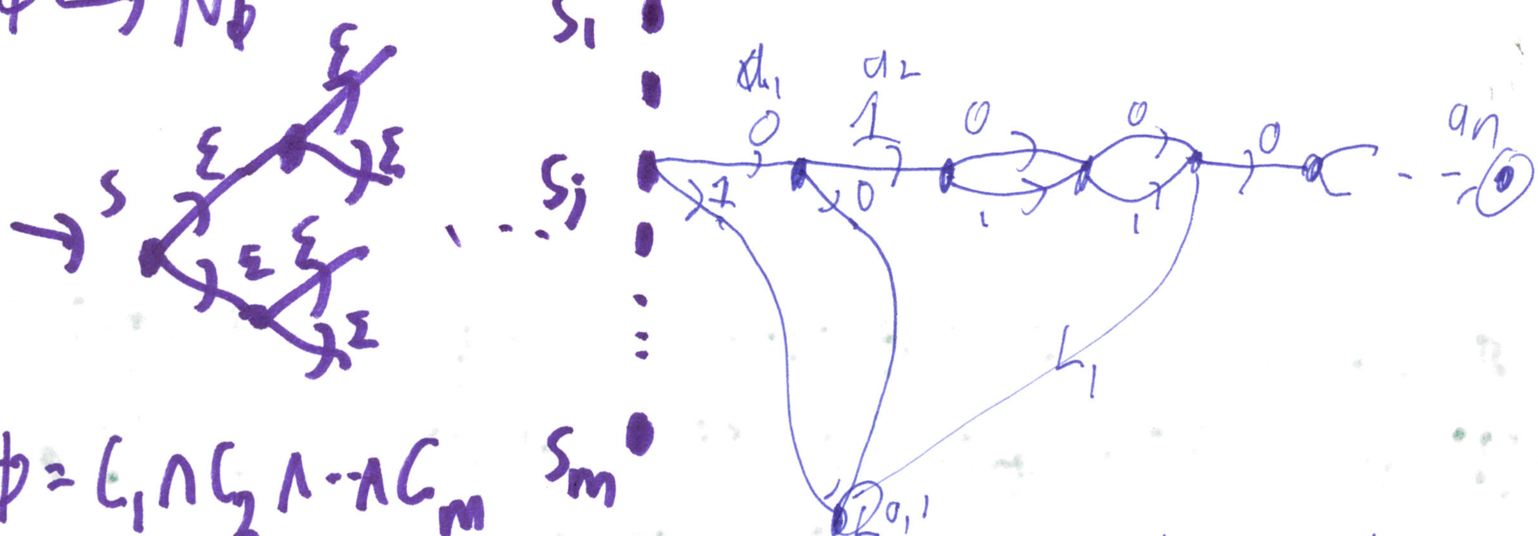
INST: An NFA $N = (Q, \Sigma, \delta, s, F)$ and a number $n < |Q|$

QUES: Is there a string $a \in \{0,1\}^n$ that N does not accept.

$NOTALL^n_{NFA}$ is in NP because we can guess a and verify that N does not accept a — not by converting N to DFA but by tracing "lights" directly.

(3) $SAT \leq^p_m NOTALL^n_{NFA}$:

$\phi \xmapsto{f} N_\phi$



$\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_m$

We will make it that some string a is not accepted iff a does not refute **any** clauses, i.e. it satisfies **all**. $C_j = (x_1 \vee \bar{x}_2 \vee x_5)$

N_ϕ has $O(nm)$ states and is built in poly time, so $NOTALL^n_{NFA}$ is NP-complete. (So is $NOTALL_{NFA}$, and finally, ditto $NOTALL^n_{Regexp}$.

The ALL_{NFA} problem is maybe harder since strings x longer than the # of states of N are involved. It is co-NP hard.

Added: By complementing, we get $TAUT \leq^p_m ALL^n_{NFA}$, so ALL^n_{NFA} is complete for co-NP. Since TAUT is complete for coNP this gives us $ALL^n_{NFA} \equiv^p_m TAUT$ just like $NOTALL^n_{NFA} \equiv^p_m SAT$.