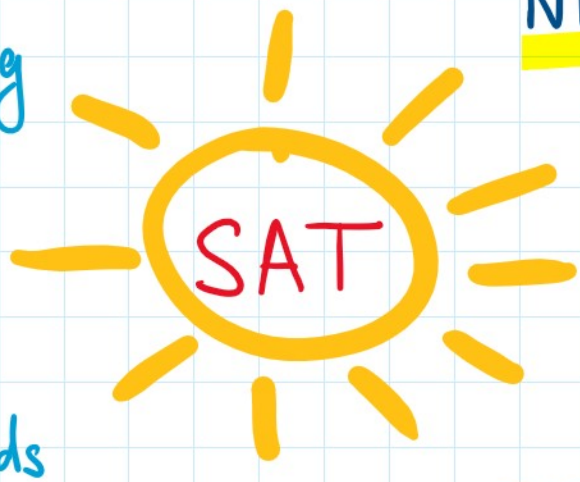
Dancing around SAT



Approx Counting

ETH / SETH

Lower Bounds

NP

PH

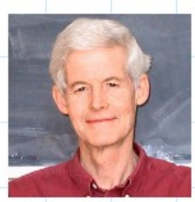
#P

IP

PCP

SAT is NP-complete

1971



S.A. Cook

L. Levin

3-SAT : Is a given 3-CNF satisfiable?

$$(x_3 \vee \bar{x}_7 \vee x_{13}) \wedge (x_1 \vee \bar{x}_3 \vee x_5) \wedge \ldots$$

$$(x_3 \vee \bar{x}_7 \vee x_{13}) \wedge (x_1 \vee \bar{x}_3 \vee x_5) \wedge \dots$$

[each clause of size 3]

Q: 3-SAT $\in P$?

[2-SAT $\in P$]

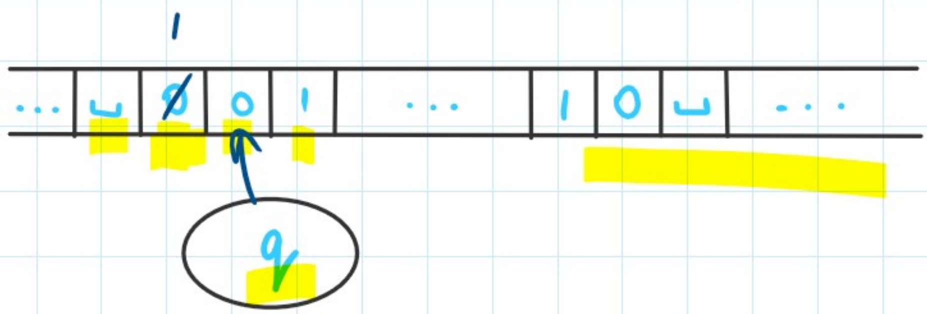
3-SAT has many faces:

3-SAT $\in P \Rightarrow NP = P$

contains lots of important problems

Definitions

Turing Machine



Local computation

if state = q & symbol = 0
then state = p & symbol = 1 & move = RIGHT

if state = q & symbol = 1
then state = r & symbol = 0 & move = LEFT

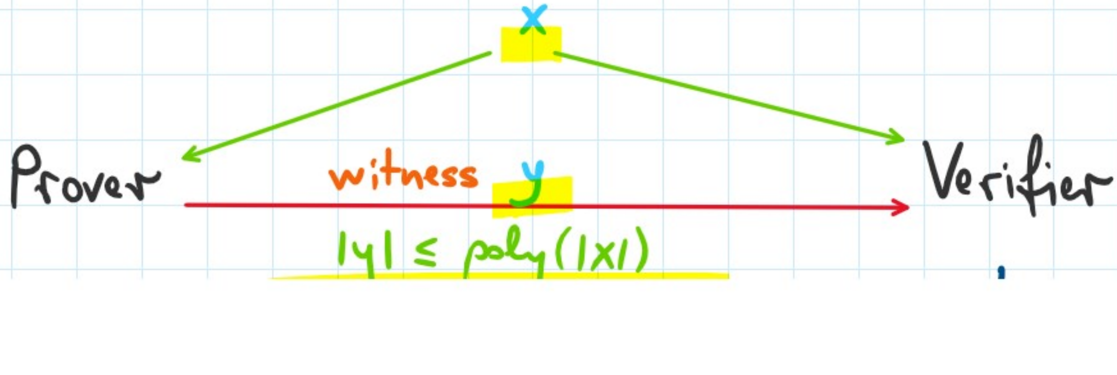
…

(q_0, q_{acc}, q_{rej})

TM time = # computation steps

P = class of problems solvable by a polytime TM

NP = class of problems solvable by a polytime NTM,
or equivalently:



Prover $\xrightarrow[\;|y| \leq poly(|x|)\;]{\text{witness } y}$ Verifier

accept $\iff V(x,y) = 1$

polytime predicate

3-SAT $\in$ NP : On input 3-CNF $\varphi(x_1, \ldots, x_n)$

φ

Prover $\xrightarrow{\text{witness } y \in \{0,1\}^n}$ Verifier

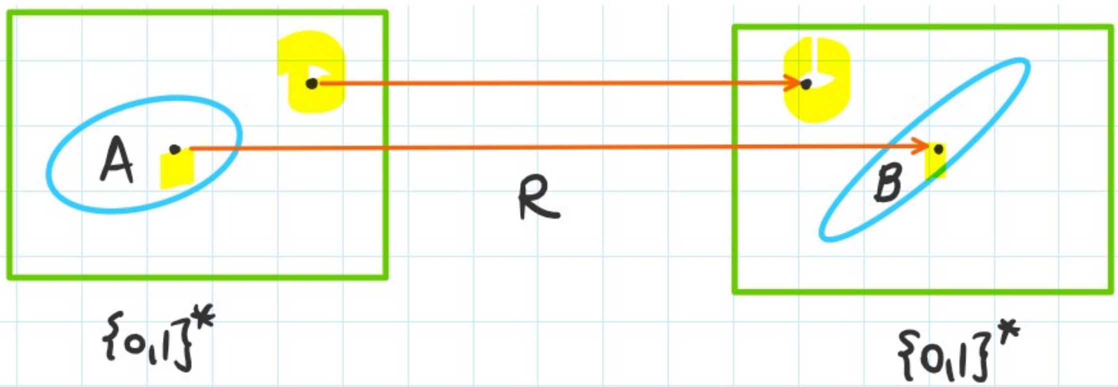
accept $\iff \varphi(y) = 1$

p-Reduction : $A, B \subseteq \{0,1\}^*$

$A \leq_p B$ if $\exists$ polytime $R : \{0,1\}^* \to \{0,1\}^*$

$\forall x \in \{0,1\}^*$

$x \in A \iff R(x) \in B$



Fact: $A \leq_p B$ & $B \in P$ $\Rightarrow$ $A \in P$.

NP-completeness : B is NP-complete if

(1) $B \in NP$, and

(2) $\forall A \in NP$, $A \leq_p B$

[B is the hardest problem in NP]

Cook-Levin's Theorem: 3-SAT is NP-complete.

Need to show:

$\forall A \in NP$, $x \xmapsto{R} \varphi_x(\vec{y})$ s.t.

$x \in A \iff \varphi_x \in$ 3-SAT

$x \in A \iff \varphi_x \in$ 3-SAT

Will show:

$\forall\ A \in NTime(t(n))$, $\quad x \xmapsto{R} \varphi_x \quad$ s.t.

$$x \in A \iff \varphi_x \in \text{3-SAT}$$

$\&\ \ |\varphi_x| \leq O(t \cdot \log t)$

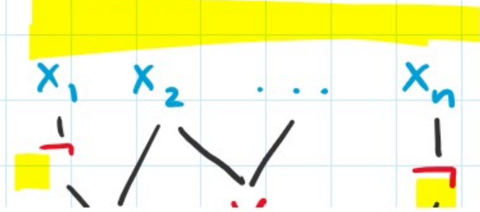
["usual" proof : $|\varphi_x| \leq O(t^2)$]

Actually, we'll prove a stronger result:

$$\text{Time}(t) \subseteq \text{Size}(O(t \cdot \log t))$$

Efficient simulation of algorithms with circuits.

input $x = x_1 x_2 \dots x_n \in \{0,1\}^n$

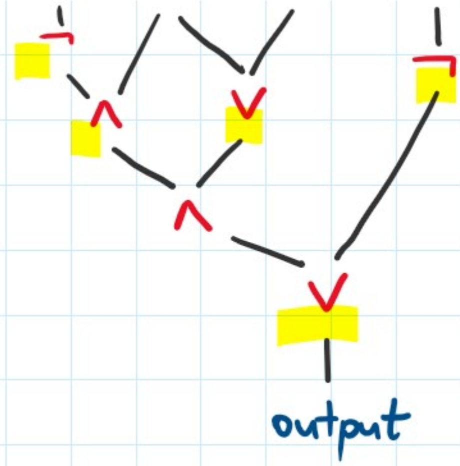


if state = q & symbol = 0

if state = q & symbol = 0
then state = p & symbol = 1 & move = RIGHT

if state = q & symbol = 1
then state = r & symbol = 0 & move = LEFT

…



circuit size = # gates (nodes)

Let V be a time $t(n)$ algorithm.

Want: $n \longrightarrow$ circuit $C(x_1, \ldots, x_n)$ s.t.

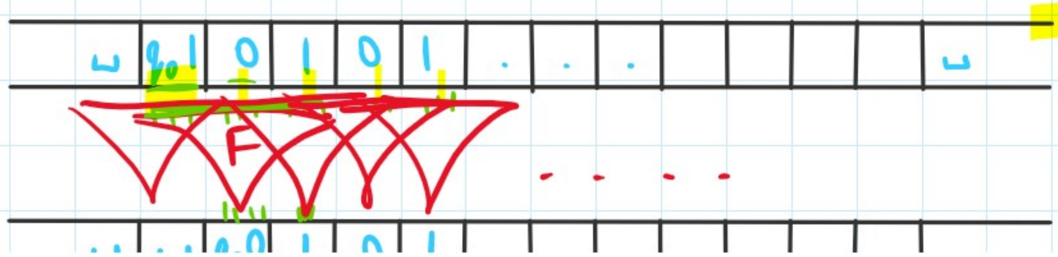
$\forall\, z \in \{0,1\}^n$, $V(z)$ accepts $\Leftrightarrow C(z) = 1$

Transcript of the computation of $V(z)$

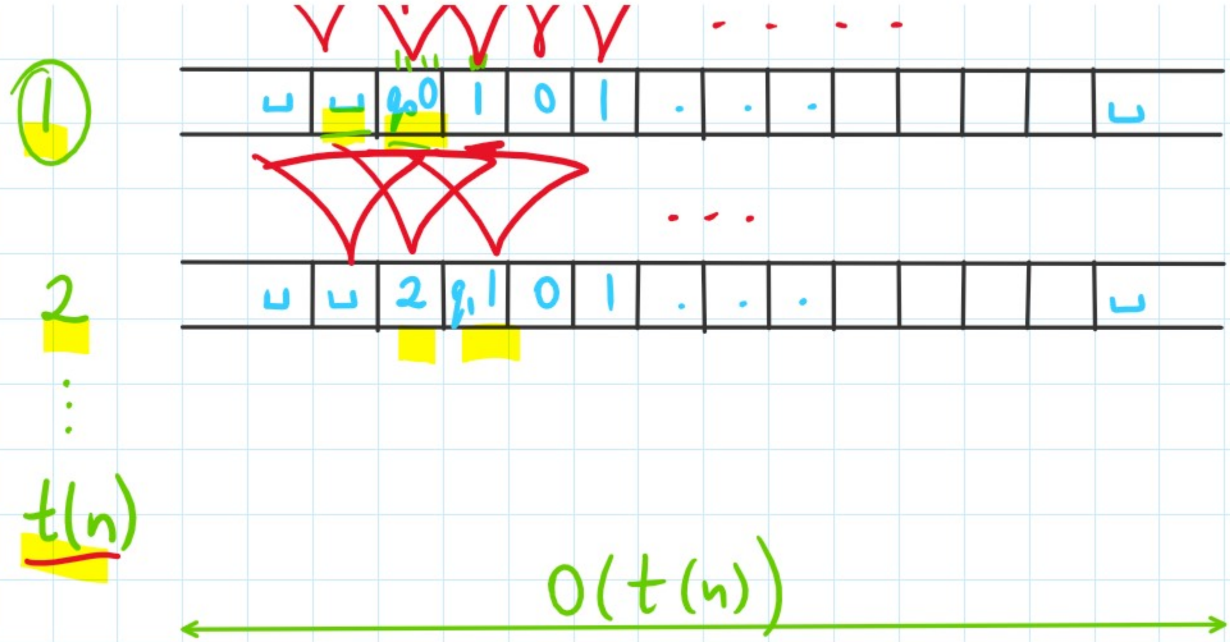
$z = \underbrace{10101\ldots}_{n}$

Time

0



$|F| \le$ const



if state = q_0 & symbol = 1
then state = q_0 & symbol = _ & move = RIGHT

if state = q_0 & symbol = 0
then state = q_1 & symbol = 2 & move = RIGHT

...

TM V

Size of the whole circuit: $O(t^2(n))$

To do better, we need special TMs.

Oblivious TM: $\forall z \in \{0,1\}^n$

tape head position at each time step i
depends only on n (but not on a specific z)

Transcript of the computation of
oblivious TM $V(z)$ $\qquad z = \underbrace{10101\ldots}_{n}$
(one-tape only, for simplicity)

⋮

$t(n)$

$$\text{Size} \leq O(t(n))$$

Theorem [Pippenger, Fisher ; Hennie, Stearns]

Every k-tape time $t(n)$ TM has an equivalent 2-tape time $O(t \cdot \log t)$ oblivious TM.

Corollary: $\text{Time}(t) \subseteq \text{Size}(O(t \cdot \log t))$.

Back to proving the NP-completeness of 3-SAT.

$L \in \text{NTime}(t(n))$

verifier $V(x, y)$, $|x| = n$, $|y| = t(n)$

verifier $V(x, y)$, $|x| = n$, $|y| = t(n)$

with runtime $O(t(n))$.

$\forall n$, can efficiently construct a circuit C s.t.

$V(x,y)$ accepts $\iff$ $C(x\,y) = 1$

& $|C| \leq O(t(n) \cdot \log t(n))$.

So, $\forall x$, $x \in L \iff \exists y\ V(x,y)$ accepts

$\iff \exists y\ C(x\,y) = 1$

$\iff C_x(y) := C(x\,y)$ is satisfiable

$\left(|C_x| \leq O(t \cdot \log t) \right)$

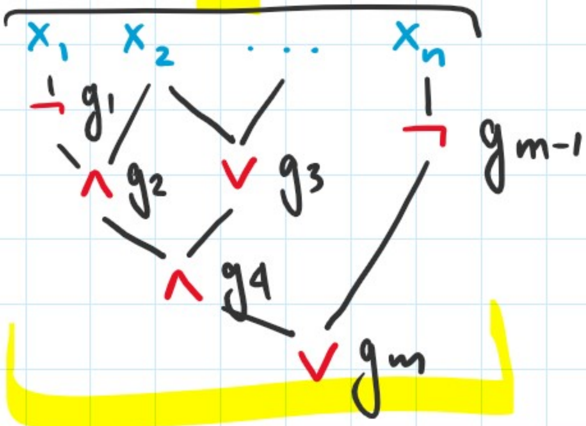
Reduction $x \xmapsto{R} C_x$ shows that

Circuit-SAT is NP-complete.

Next:

$$\text{Circuit-SAT} \leq_p \text{3-SAT}$$

$$C \longmapsto \varphi_C$$



$$\varphi_C = (g_1 \equiv \neg x_1) \wedge (g_2 \equiv g_1 \wedge x_2) \wedge \ldots \wedge (g_m \equiv 1)$$

Locality of computation again!

Note: $|\varphi_C| \leq O(|C|)$

So, $\text{NTime}(t(n))$

$$x,\ |x| = n \longmapsto C_x,\ |C_x| \leq O(t \cdot \log t)$$

$$\longmapsto \varphi_{C_x},\ |\varphi_{C_x}| \leq O(t \cdot \log t)$$

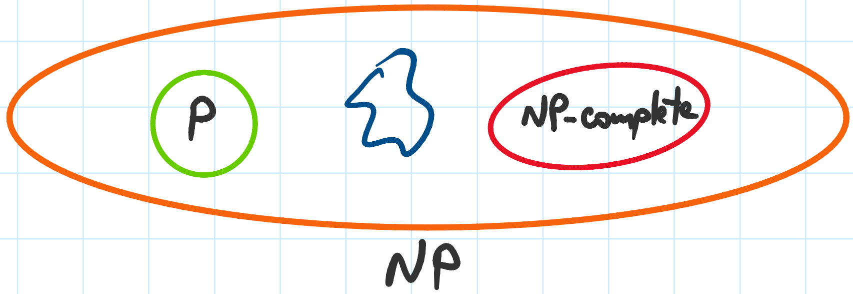
$$(\leq 3)\text{-SAT} \leq_p 3\text{-SAT}$$

$$(x \vee \bar{y}) \longmapsto (x \vee \bar{y} \vee z) \wedge (x \vee \bar{y} \vee \bar{z})$$

Corollary: $P \subseteq \text{Size}(\text{poly})$

Hence, $\underbrace{3\text{-SAT} \notin \text{Size}(\text{poly})}_{\text{circuit complexity}} \Rightarrow 3\text{-SAT} \notin P$

SAT Dichotomy



Ladner's Theorem ('75): If $P \neq NP$, then there are NP-intermediate problems in NP: neither in P, nor NP-complete.

But, SAT is special!

Dichotomy for SAT variants

1. 2-SAT $\in$ P

3-SAT is NP-complete

2. XOR-SAT $\in$ P $[\, x_1 \oplus x_3 \oplus x_7 = 1, \ldots]$

3. Horn-SAT $\in$ P $[\, \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_5 \vee x_7, \ldots]$

4. dual Horn-SAT $\in$ P $[\, x_1 \vee x_2 \vee x_5 \vee \bar{x}_7, \ldots]$

5. [satisfiable by an all-0 truth assignment
satisfiable by an all-1 truth assignment]

Schaefer's Dichotomy Theorem ('78):

Every SAT-variant whose clauses are of the same type i, $1 \le i \le 5$, is in P.

Otherwise (if mixing clause types), the resulting SAT-variant is NP-complete.

Proof idea: Say mix dual Horn clauses and width-2 clauses.

width-2 clauses.

Then can express NP-complete problem 3-COL:

$G = (V, E)$ undirected graph

$\exists ?$ $C : V \to \{R, G, B\}$ s.t.

$\forall (u,v) \in E$ $C(u) \neq C(v)$.

Constraints:

$\forall v \in V,$ $R_v \vee G_v \vee B_v$

$\forall (u,v) \in E,$ $(\overline{R}_u \vee \overline{R}_v) \wedge (\overline{G}_u \vee \overline{G}_v) \wedge (\overline{B}_u \vee \overline{B}_v)$

Dichotomy Theorem also holds for non-Boolean SAT-variants (a.k.a. CSPs) [Bulatov; Zhuk 2017]

What can you do with a SAT-algorithm?

"Search to Decision" reduction for SAT

SAT-Search

input: $\varphi(x_1, x_2, \ldots, x_n)$

output: $a_1 a_2 \ldots a_n \in \{0,1\}^n$

s.t. $\varphi(a_1, a_2, \ldots, a_n) = 1$,

or "NO" if $\varphi \notin SAT$

SAT

input: $\varphi(x_1, x_2, \ldots, x_n)$

output: $\begin{cases} 1 & \text{if } \varphi \in SAT \\ 0 & \text{otherwise} \end{cases}$

SAT

Theorem: SAT-Search $\in P^{SAT}$

Proof:

SAT-Search ($\phi(x_1, \dots, x_n)$)

if $\phi \notin$ SAT **then** output "NO"
else
  for $i = 1$ to n
    if $\phi(a_1, \dots, a_{i-1}, 0, x_{i+1}, \dots, x_n) \in$ SAT
    then $a_i = 0$ **else** $a_i = 1$ **endif**
  endfor
  output $a_1 a_2 \dots a_n$
endif

Alternating Quantifiers

$\exists x_1 \exists x_2 \dots \exists x_n \quad \varphi(x_1, x_2, \dots, x_n)$ SAT

$\forall x_1 \forall x_2 \dots \forall x_n \quad \varphi(x_1, x_2, \dots, x_n)$ TAUT

$\varphi \in$ TAUT $\Leftrightarrow \neg\varphi \notin$ SAT

Q: $NP = coNP$? [proof complexity]

Note: $NP = P \Rightarrow coNP = P = NP$.

So, $NP \neq coNP \Rightarrow NP \neq P$.

Σ_2^P: on input $x \in \{0,1\}^n$, decide if

$$\exists y \, \forall z \quad W(x, y, z)$$

$|y|, |z| \leq poly(n)$

polytime predicate W

Ex: Circuit Minimization

on input $\langle C(b_1, b_2, \dots, b_n) \rangle$, decide if

$\exists \ \langle \delta(b_1, b_2, \dots, b_n) \rangle$ "$|\delta| < |C|$" &

$\forall \; z_1 z_2 \ldots z_n \in \{0,1\}^n$

"$C(z_1, z_2, \ldots, z_n) = D(z_1, z_2, \ldots, z_n)$" $\in W$

$\Sigma_k^p, \quad \forall \; k \geq 0 \qquad \left(\Pi_k^p = co\Sigma_k^p \right)$

$PH = \bigcup_{k \geq 0} \Sigma_k^p$

Polynomial-Time Hierarchy

Theorem: $SAT \in P \Rightarrow PH = P$

Proof Idea: Let A be a polytime Circuit-SAT algo.

Σ_2^p : on $x \in \{0,1\}^n$, decide if

$\exists y \; \forall z \;\; W(x, y, z)$

$|y|, |z| \leq poly(n)$

$W \in P$

By "Algorithm $\to$ Circuit" construction,

$n \overset{R_W}{\longmapsto}$ circuit $C(x,y,z)$ equivalent to $W(x,y,z)$

Σ_2^p : on $x \in \{0,1\}^n$, decide if

$$\exists y \; \boxed{\forall z \; C(x,y,z)}, \text{ for } C = R_W(n)$$

$\iff$

$$\exists y \; \boxed{\forall z \; C_{x,y}(z)}, \text{ for } C = R_W(n),$$

$$C_{x,y}(z) = C(x,y,z)$$

$\iff$

$$\exists y \; \boxed{\mathcal{A}(\neg C_{x,y}) = 0, \text{ for } C \text{ as above}}$$

$\iff$

Σ_1^p: $\exists y \; W'(x,y)$

$\text{time}_{W'}(n) \le \text{time}_{\mathcal{A}}(\tilde{O}(\text{time}_W(n)))$

$$\text{time}_{W'}(n) \leq \text{time}_{\mathcal{A}}(O(\text{time}_{W}(n)))$$

growing polytime

$$\Longleftrightarrow \boxed{\exists y \;\; C'_x(y)}, \quad C' = R_{W'}(n),$$

$$C'_x(y) = C'(x,y)$$

$$\Longleftrightarrow \mathcal{A}(C'_x) = 1, \text{ for } C' \text{ as above}$$

P :

$$\text{time}(n) \leq \text{time}_{\mathcal{A}}(\tilde{O}(\text{time}_{W'}(n)))$$

$$\tilde{\leq} \text{time}_{\mathcal{A}}(\text{time}_{\mathcal{A}}(\text{time}_{W}(n)))$$

Remarks:

(1) can only handle Σ_k^p for CONSTANT k

(2) need "white-box" access to SAT-algo $\mathcal{A}$

(2) need "white-box" access to SAT-algo $\mathcal{A}$

SAT-Search $\in P^{SAT}$

but our proof does not show that $PH \subseteq P^{SAT}$

#SAT

input: $\varphi(x_1, x_2, \ldots, x_n)$

output: # satisfying assignments of φ, i.e.,

$$\sum_{x_1 \ldots x_n \in \{0,1\}^n} \varphi(x_1, \ldots, x_n).$$

$NP \subseteq P^{\#SAT}$

Toda's Theorem ('91): $PH \subseteq P^{\#SAT}$

Corollary: $\#SAT \in \mathrm{Time}(2^{n^{o(1)}}) \Rightarrow$
$PH \subseteq \mathrm{Time}(2^{n^{o(1)}})$

our earlier proof of "$SAT \in P \Rightarrow PH = P$"
doesn't show "$SAT \in SUBEXP \Rightarrow PH \subseteq SUBEXP$"

Unique-SAT

either $\#SAT(\varphi) = 1$

Unique-SAT

input: $\varphi(x_1, \ldots, x_n)$ — either $\#SAT(\varphi) = 1$ or $\#SAT(\varphi) = 0$

decide: is $\varphi \in SAT$?

$$\text{Unique-SAT} \le_p SAT$$

$$SAT \in RP^{\text{Unique-SAT}}$$

[RP = Randomized Polytime]

Valiant-Vazirani Theorem ('86):

There is a randomized polytime algorithm that, on input $\varphi(x_1, \ldots, x_n)$ outputs a list $\psi_1(x_1, \ldots, x_n), \ldots, \psi_n(x_1, \ldots, x_n)$ such that:

- $\varphi \notin SAT \Rightarrow$ each $\psi_i \notin SAT$
- $\varphi \in SAT \Rightarrow$ with probability $\geq \frac{1}{8}$, some $\psi_i \in$ Unique-SAT

Proof Idea:

Pick random $S \subseteq \{1, \dots, n\}$ & random $b \in \{0,1\}$.

Let $h_{S,b}(x_1, \dots, x_n) =$ " $\sum_{i \in S} x_i = b \pmod 2$ "

If $\varphi(x_1, \dots, x_n)$ has M sat. assignments, then $\varphi \wedge h_{S,b}$ has $\frac{M}{2}$ sat. assignments

in expectation.

So, $\varphi \wedge$ " $\log_2 M$ random parity constraints" is expected to have 1 sat. assignment.

"Expectation $\longrightarrow$ High Probability" follows from 2-wise independence of $h_{S,b}$.

(hashing)

Don't know $\#SAT(\varphi)$!

But, enough to get integer K:

$$2^{K-2} \leq \#SAT(\varphi) \leq 2^{K-1}$$

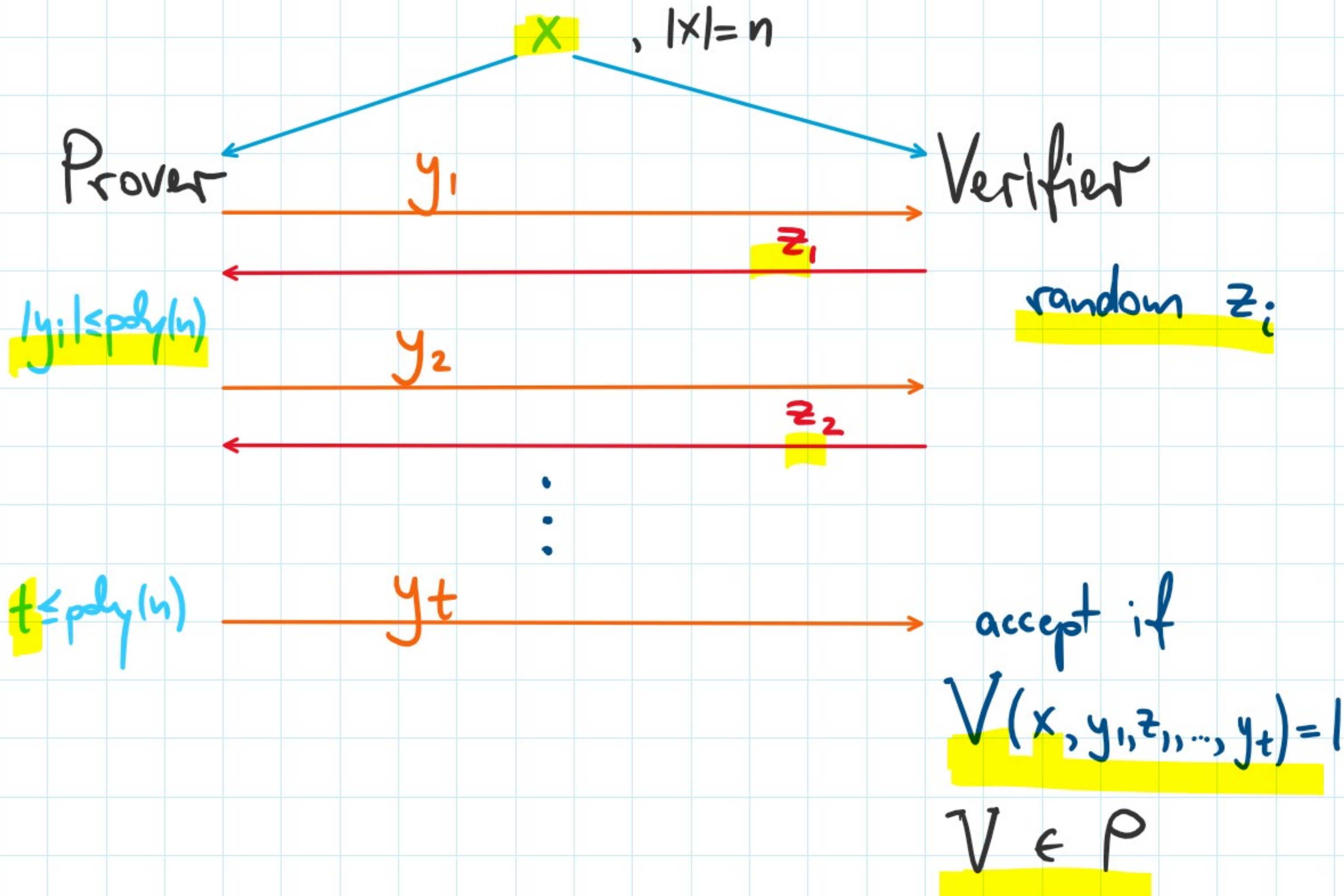
Try all $2 \leq K \leq n+1$,

$\psi_K = \varphi \wedge$ [K random parity constraints]

a list of n formulas.

#SAT $\in$ IP

Interactive Proofs



Def: $L \in IP$ if there exists a polytime V

s.t. $\forall\ x \in \{0,1\}^n$,

s.t. $\forall x \in \{0,1\}^n$,

$x \in L \Rightarrow \exists$ Prover $\Pr_{z_1,\dots,z_t}[V(x,y_1,z_1,\dots,y_t,z_t)=1]=1$ completeness

$x \notin L \Rightarrow \forall$ Prover $\Pr_{z_1,\dots,z_t}[V(\dots)=1] \le \frac{1}{3}$ soundness

define $\#SAT_D$

input: 3cnf $\varphi(x_1,\dots,x_n)$, $0 \le K \le 2^n$

decide: is $\#SAT(\varphi)=K$?

Theorem [Lund, Fortnow, Karloff, Nisan '90]:

$\#SAT_D \in IP$.

Proof:

(1) 3cnf $\varphi(x_1,\dots,x_n) \longmapsto$ arithmetic formula

$A_\varphi(x_1,\dots,x_n)$ computing a polynomial

$p_\varphi(x_1,\dots,x_n)$ of degree $\le \mathrm{poly}(|\varphi|)$

such that $\forall z \in \{0,1\}^n$, $\varphi(z) = p_\varphi(z)$

(2) IP protocol, given $A_\varphi(x_1,\dots,x_n)$, $0 \le K \le 2^n$,

verifying that

$$\sum_{x_1=0}^{1} \dots \sum_{x_n=0}^{1} p_\varphi(x_1,\dots,x_n) = K.$$

(1) arithmetization :

formula $\varphi \longmapsto$ polynomial p_φ

$$0 \longmapsto 0$$
$$1 \longmapsto 1$$
$$x \longmapsto x$$
$$\neg\psi \longmapsto 1 - p_\psi$$
$$\psi_1 \wedge \psi_2 \longmapsto p_{\psi_1} \cdot p_{\psi_2}$$
$$\psi_1 \vee \psi_2 \longmapsto 1 - (1 - p_{\psi_1})\cdot(1 - p_{\psi_2})$$

[$\psi_1 \vee \psi_2 \equiv \neg\neg(\psi_1 \vee \psi_2)$

$$\left[\psi_1 \vee \psi_2 \equiv \neg\neg(\psi_1 \vee \psi_2) \equiv \neg(\neg\psi_1 \wedge \neg\psi_2)\right]$$

Ex. $\underbrace{(x \vee \bar{y} \vee z)}_{\text{clause}} \longmapsto \underbrace{1-(1-x)\,y\,(1-z)}_{P_{\text{clause}}}$

$$\varphi(x_1,\dots,x_n) = \text{clause}_1 \wedge \dots \wedge \text{clause}_m$$

$$\longmapsto P_{\text{clause}_1} \cdot \dots \cdot P_{\text{clause}_m}$$

arithmetic formula of size $O(m)$

degree $\leq 3 \cdot m$

(2) IP protocol for $\sum_{x_1=0}^{1} \dots \sum_{x_n=0}^{1} P_\varphi(x_1,\dots,x_n) \stackrel{?}{=} K$

Randomized Σ-quantifier Elimination

$$(*) \quad K \stackrel{?}{=} \sum_{x_1=0}^{1} \sum_{x_2=0}^{1} \dots \sum_{x_n=0}^{1} p(x_1,x_2,\dots,x_n)$$

Prover Verifier

Prover $\longrightarrow$ Verifier

$\longleftarrow$

(**) $K' \stackrel{?}{=} \sum_{x_2=0}^{1} \ldots \sum_{x_n=0}^{1} p'(x_2, \ldots, x_n)$

(*) true $\Rightarrow$ (**) true with prob. 1

(*) false $\Rightarrow$ (**) false with prob. $1 - \frac{1}{poly}$

Define: $q(x_1) := \sum_{x_2=0}^{1} \ldots \sum_{x_n=0}^{1} p(x_1, x_2, \ldots, x_n)$

polynomial of degree $\leq 3m$

Observe:

$q(0) + q(1) = \sum_{x_1=0}^{1} \sum_{x_2=0}^{1} \ldots \sum_{x_n=0}^{1} p(x_1, x_2, \ldots x_n)$

K is correct $\Longleftrightarrow$ $K = q(0) + q(1)$

Prover sends (the coefficients of) polynomial $f(x_1)$, claiming $f(x_1) \equiv q(x_1)$.

$f(x_1)$, claiming $f(x_1) \equiv q(x_1)$.

Verifier checks if $K = f(0) + f(1)$

If not equal, Reject!

K incorrect, but $K = f(0) + f(1) \Rightarrow f \not\equiv q$

Define $K' := f(r)$, random $1 \le r \le 2^n$

$p'(x_2, \dots, x_n) := p(r, x_2, \dots, x_n)$

Observe

$$\sum_{x_2=0}^{1} \dots \sum_{x_n=0}^{1} p'(x_2, \dots, x_n) = q(r)$$

$$\Pr_r \left[K' = \sum_{x_2=0}^{1} \dots \sum_{x_n=0}^{1} p'(x_2, \dots, x_n) \right] =$$

$$\Pr_r \left[f(r) = q(r) \right] \le \frac{3m}{2^n}. \quad o(1)$$

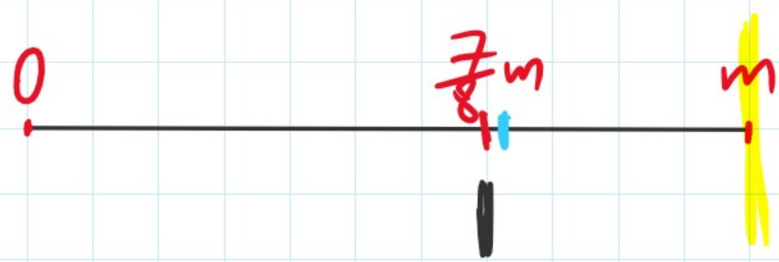
NP-hardness of Approximations

3-SAT

3-cnf $\varphi(x_1, \dots, x_n)$ on m clauses

How many clauses of φ can be satisfied?

- $\varphi \in SAT \Rightarrow m$
- $\varphi \notin SAT \Rightarrow < m$



Fact: Can always satisfy $\geq \frac{7}{8} \cdot m$ clauses.

Proof:

$$\Pr_{x_1, \dots, x_n}\left[x_i \vee \bar{x}_j \vee x_k = 1\right] = 1 - \frac{1}{8} = \frac{7}{8}$$

$$\underset{x_1,\dots,x_n}{\mathrm{Exp}}\left[\#\text{ satisfied clauses}\right] = \frac{7}{8}\cdot m \qquad \square$$

PCP Theorem [Håstad '97]: Fix any $\varepsilon > 0$.
There is a polytime reduction R from 3-SAT to 3-SAT s.t. $\forall$ 3-cnf φ

(1) $\varphi \in$ 3-SAT $\Rightarrow R(\varphi) \in$ 3-SAT

(2) $\varphi \notin$ 3-SAT $\Rightarrow$

at most $\left(\frac{7}{8} + \varepsilon\right)$ fraction of clauses in $R(\varphi)$ can be satisfied by any assignment.

Corollary: $\forall \varepsilon > 0$, it's NP-hard to distinguish

Corollary: $\forall \varepsilon > 0$, it's NP-hard to distinguish satisfiable 3-SAT formulas from $\left(\frac{7}{8}+\varepsilon\right)$-satisfiable 3-SAT formulas.

MAX-3SAT is NP-hard to C-approximate for $\boxed{C = \frac{8}{7} + \varepsilon}$, which is optimal.

C-approximation for a MAX-3SAT instance φ is a value V s.t.

$$\frac{OPT(\varphi)}{C} \le V \le OPT(\varphi),$$

where $OPT(\varphi)$ = max # satisfiable clauses.

Probabilistically Checkable Proofs

Probabilistically Checkable Proofs

3-cnf $\varphi(x_1, \dots, x_n)$

$$\text{Prover} \xrightarrow{\;y = y_1 \cdots y_t \in \{0,1\}^t\;} \text{Verifier}$$

constructs 3-cnf $\psi = R(\varphi)$

Håstad's reduction

ψ has $t \leq \text{poly}(n)$ variables

$m \leq \text{poly}(n)$ clauses

Pick a random clause C_i

& accept if $\underbrace{C_i(y)}_{\text{only reads 3 bits of } y}$ is sat.

- $\varphi \in 3SAT \Rightarrow \exists y \; \Pr_i[\text{Verifier}^y \text{ accepts}] = 1$
- $\varphi \notin 3SAT \Rightarrow \forall y \; \Pr_i[\text{Verifier}^y \text{ accepts}] \leq \frac{7}{8} + \varepsilon$