CPSC 421/501 Nov 27

Today: give an easy reduction.

Then: Ch 8, Ch 9 ← How not to solve P vs NP

Ch 9 Section 3 } How to solve P vs NP

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

Last week: given 3SAT is NP-complete showed that SUBSET-SUM is NP-complete

Real work:

$$3SAT \leq_p SUBSET\text{-}SUM$$

Another example, on Homework

$$3SAT \leq_p 3COLOR$$

All these reductions are tricky.

Not so tricky:

Show that 4SAT is NP-complete.

$$4SAT = \left\{ \langle f \rangle \;\middle|\; \begin{array}{l} f \text{ is Boolean formula } (\\ \text{in 4CNF form that} \\ \text{is satisfiable} \end{array} \right\}$$

4CNF: $c_1 \wedge c_2 \wedge \ldots \wedge C_m$,

where

each $C_i = l_{i1} \vee l_{i2} \vee l_{i3} \vee l_{i4}$

l_{i1} or l_{i2} or l_{i3} or l_{i4}

l_{ij}'s are literals, meaning variables or their negations:

$x_1, \ldots, x_n, \neg x_1, \neg x_2, \ldots, \neg x_n$.

---

1 4SAT is in NP; given

$f = f(x_1, \ldots, x_n) \in 4CNF$,

"guess" the values of $x_1, \ldots, x_n = T, F$

check if $f(x_1, \ldots, x_n) = T$ or F.

Really

$|\langle f \rangle|$ ← length of description of f

$\leq_p$ ← poly time reduction !

you must take $\langle f \rangle$, in

time polynomial $(|\langle f \rangle|)$ you

must have a non-det machine

that always decides in time

$\leq$ poly$(|\langle f \rangle|)$

$f = x_1 \text{ OR } x_{1000000}$

$$\left| \langle x_1 \vee x_{1000000} \rangle \right|$$

$$= \left| x1 \vee x1000000 \right|$$

$$= 11$$

② $3SAT \leq_p 4SAT$

$$\underbrace{3SAT}_{\text{any NP-complete problem}}$$

(SAT, 3COLOR,

SUBSET-SUM)

Given $f \in 3CNF$

$(x_1 \vee x_2 \vee \neg x_3)$

AND

$(x_2 \vee x_3 \vee \neg x_4)$

$\vdots$

similarly

$(x_1 \vee x_2 \vee \neg x_3 \vee x_1)$

Similarly

$(l_{i1} \vee l_{i2} \vee l_{i3}) \leftrightarrow (l_{i1} \vee l_{i2} \vee l_{i3} \vee l_{i1})$

Make sure! $3SAT \le_p \underbrace{4SAT}$

(not the other way...).

"as difficult as any problem in NP"

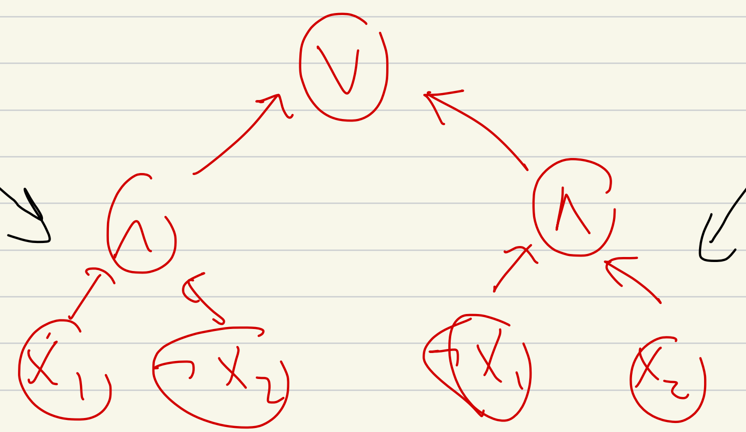
Chapter 9: How to solve P vs NP ...

Idea: We look at formulas and circuits in Boolean algebra...

Boolean formula

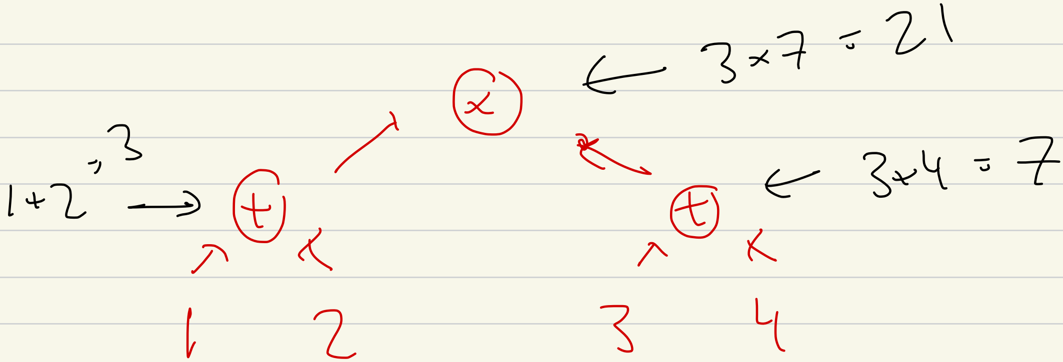
$$(x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2)$$

literals: $x_1, x_2, \neg x_1, \neg x_2$



Similarly

$$(1+2) \times (3+4)$$



Formula $\longleftrightarrow$ Tree

Circuit :

Say

$$(1+5)\times(1+5)\times(1+5)\times(1+5)$$

$$\times (2+3) \times (2+3) \;=\; ?$$

As a program:

$$y_1 = 1+5$$

$$y_2 = y_1 \times y_1 = (1+5)^2$$

$$y_3 = y_2 \times y_2 = (1+5)^4$$

$$y_4 = 2+3$$

$$y_5 = y_4 \times y_4 = (2+3)^2$$

$$y_6 = \text{Answer} = y_3 \times y_5$$

So $y_1, \dots, y_6$ represents six operations $+, \times$.

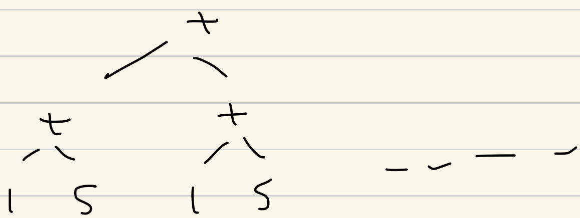
As a formula:

$$(1+5)\times(1+5)\times(1+5)\times(1+5)$$
$$\times(2+3)\times(2+3)$$

has $+$ occurring 6 times

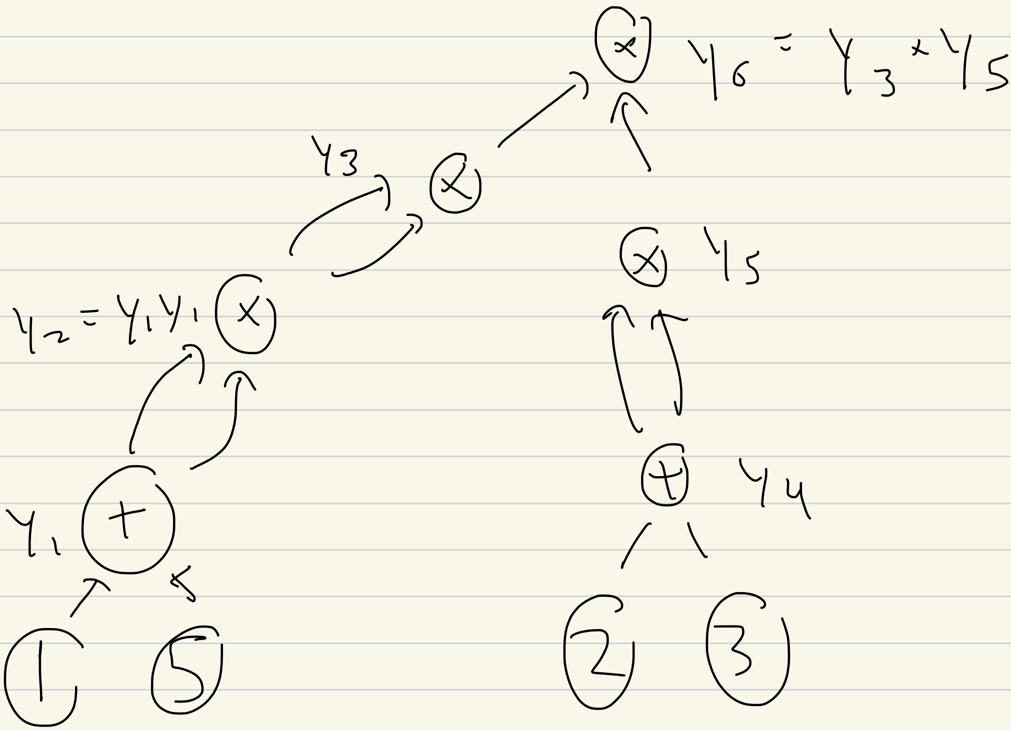
$\times$ " 5 times

11 operations



or 1, 2, 3, 5 occurs 12 times

By contrast:



How "wide" is the circuit?

How "deep"?

Many practical considerations ~

Chapter 9: If $L \in NP$,

$L \subset \Sigma^*$, $\Sigma = \{T, F\}$.

Then for each $n \in \mathbb{N}$, there

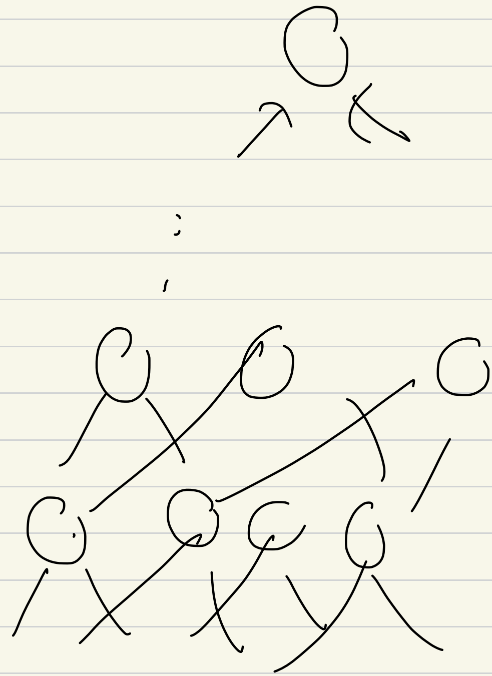
is a circuit, C_n, of size poly(n)

that computes, given $w \in \Sigma^n$,

($|w| = n$) such that

$w \in L \iff C_n$ on input w gives T

size is $\text{poly}(n)$



$\leftarrow \vee, \wedge, \neg$

$w = \sigma_1 \; --- \; \sigma_n$

(§ 9.3 of [Sip])