- This conversion of a boolean CNF to an algebraic polynomial is called <u>arithmetization</u>.

- Another way to arithmetize CNF:

<u>Proposition</u>: $\text{QuadEqn}_2 := \{S \mid S$ is a system of quadratic equations modulo 2 & $S$ has a root$\}$ is NP-complete.

<u>Proof</u>:

- Since, given a point $x \in \mathbb{F}_2^n$ it is easy to verify whether it is a root of $S$, we have $\text{QuadEqn} \in NP$.

- For any 3CNF $\phi$ we now convert each clause to a quadratic system (mod 2).

• Eg. the clause $(x_1 \vee \bar{x}_2 \vee x_3)$ becomes:

$$\begin{cases} (1-x_1)z = 0 \pmod 2 \\ z = x_2(1-x_3) \pmod 2 \end{cases}$$

• The clause is true iff the quadratic system has a root.

• Similarly, $\phi$ is satisfiable iff the corresponding quadratic system has a root.

$\Rightarrow$ 3SAT $\leq_p$ $\text{QuadEqn}_2$. $\square$

- Exercise: How about $\text{QuadEqn}_p$ ?

# co-Classes

- For a language $L$ we define the <u>co-problem</u> as $\overline{L} := \{0,1\}^* \setminus L$.

- This gives us <u>co-classes</u> as:

  $$\underline{\text{coNP}} := \{ \overline{L} \mid L \in \text{NP} \}.$$

- In other words, for a language $L$ in coNP it is "easy" to <u>verify</u> $x \notin L$, for a string $x$.

- What is the "hardest" problem in coNP?

<u>Defn:</u> $\underline{\text{Taut}} := \{ \phi \mid \phi$ is a DNF formula & $\phi$ is a tautology $\}$.

<u>Proposition:</u> Taut is coNP-complete.

<u>Proof:</u>
- Given a DNF $\phi$ we consider the CNF $\neg\phi$.
- $\neg\phi \in$ SAT iff $\phi \notin$ Taut.

$\Rightarrow$ Taut $\in$ coNP.

- Let $L \in$ coNP. Thus $\exists$ poly-time TM $M$ s.t. $x \notin L$ iff $\exists u \in \{0,1\}^{|x|^c}$, $M(x,u) = 1$.
- Use Cook-Levin's reduction on $M$ to get a boolean CNF $\phi_x(u)$ s.t.

$x \in \bar{L}$ iff $\phi_x(u)$ is satisfiable.

$\Rightarrow$ $x \in L$ iff $\phi_x(u)$ is unsatisfiable.

$\Rightarrow$ $x \in L$ iff $\neg\phi_x(u) \in$ Taut.

$\Rightarrow$ $L \leq_p$ Taut.

- Thus, Taut is coNP-complete. $\square$

- Open qn: NP $\neq$ coNP? Equivalently, Taut $\notin$ NP?

- **Proposition:** (i) $P = coP \subseteq NP \cap coNP$.

(ii) $P = NP \implies NP = coNP$.

(Thus, $NP \neq coNP \Rightarrow P \neq NP$.)

(iii) $NP \cup coNP \subseteq EXP$.

## NEXP

- It is the nondeterministic version of EXP:

$$\underline{NEXP} := \bigcup_{c \in \mathbb{N}} Ntime(2^{n^c}).$$

- Easily,

▷ $$P \subseteq NP \subseteq EXP \subseteq NEXP.$$

**Theorem:** $P = NP \implies EXP = NEXP$.

Proof: Idea: Padding a language.

- Suppose $P = NP$ & $L \in NEXP$.
- Let the poly-time verifier TM (that uses an exp. certificate) be M st.

$x \in L$ iff $\exists u \in \{0,1\}^{2^{|x|^c}}$, $M(x, u) = 1$.

- Consider the <u>padded</u> version of $L$:

$L' := \{(x, 0^{2^{|x|^c}}) \mid x \in L\}$.

- $L' \in NP$. ($\because$ any $x' \in L'$ can now be verified by a $|x'|$-sized certificate in poly-time by $M$.)

- By the hypothesis $L' \in P$.
  Say, $L' \in \mathrm{Dtime}(n^d)$.

$\Rightarrow$ For an $x$, we can decide $x \in L$ in time $O((|x| + 2^{|x|^c})^d)$.

$\Rightarrow$ $L \in EXP$

$\Rightarrow$ $NEXP = EXP$. $\square$

- Where to place NEXP ?

<u>Defn</u>: <u>EEXP</u> $:= \bigcup_{c \in \mathbb{N}} \text{Dtime}(2^{2^{n^c}})$ .

▷ $\text{EXP} \subseteq \text{NEXP} \subseteq \text{EEXP}$ .

and so on ......

<u>Gödel's computation qn.</u>

- <u>Thms</u> $:= \{(\varphi, 1^n) \mid \varphi$ is a math. statement with a proof of length $\leq n\}$.

- Since it is "easy" to verify a proof:

▷ Thms $\in$ NP.

- If P=NP then every math. statement can be "easily" resolved.

No need for mathematicians!