Co-NP
and
the Asymmetry of NP

Efficient Certification

B is an efficient certifier for a problem X if the following properties hold

- B is a polynomial-time algorithm that takes two input arguments s and t
- There is a polynomial function p so that for every string s, we have $s \in X$ if and only if there exists a string t such that $|t| \leq p(|s|)$ and $B(s, t) = \text{yes}$

Observation

- An input string is a ‘yes’ instance if and only if there exists t so that B(s, t) = yes
- Negating this statement, we see that an input string s is a ’no’ instance if and only if for all t, it’s the case that B(s, t) = no

Complementary Problem

For every problem x, there is a natural complementary problem $\overline{x}$:

- For all input strings s, we say $s \in \overline{x}$ if and only if $s \notin x$

If $x \in P$, then $\overline{x} \in P$

Should $\overline{x}$ $\in$ NP if x $\in$ NP?

- The problem $\overline{x}$, rather, has a different property: for all s, we have s $\in$ $\overline{x}$ if and only if for all t of length at most p(|s|), B(s, t) = no
- This is a fundamentally different definition, and it can't be worked around by simply "inverting" the output of the efficient certifier B to produce B
- The problem is that the "exists t" in the definition of NP has become a "for all t," and this is a serious change

A problem X belongs to co-NP if and only if the complementary problem $\overline{x}$ belongs to NP

If NP ≠ co-NP, then P ≠ NP

$$X \in \mathcal{NP} \Longrightarrow X \in \mathcal{P} \Longrightarrow \overline{X} \in \mathcal{P} \Longrightarrow \overline{X} \in \mathcal{NP} \Longrightarrow X \in \text{co-}\mathcal{NP}$$

and

$$X \in \text{co-}\mathcal{NP} \Longrightarrow \overline{X} \in \mathcal{NP} \Longrightarrow \overline{X} \in \mathcal{P} \Longrightarrow X \in \mathcal{P} \Longrightarrow X \in \mathcal{NP}$$

Hence it would follow that NP ⊆ co-NP and co-NP ⊆ NP, whence NP = co-NP

The Class NP ∩ co-NP

- If a problem X belongs to both NP and co-NP, then
    - When the answer is yes, there is a short proof
    - When the answer is no, there is a short proof
- The problems that belong to this intersection NP ∩ co-NP are said to have a good characterization, since there is always a nice certificate for the solution
    - Example : Determine whether a flow network contains a flow of value at least v, for some quantity v
- $P \subseteq NP \cap co\text{-}NP$

PSPACE

The set of all problems that can be solved by an algorithm with polynomial space complexity

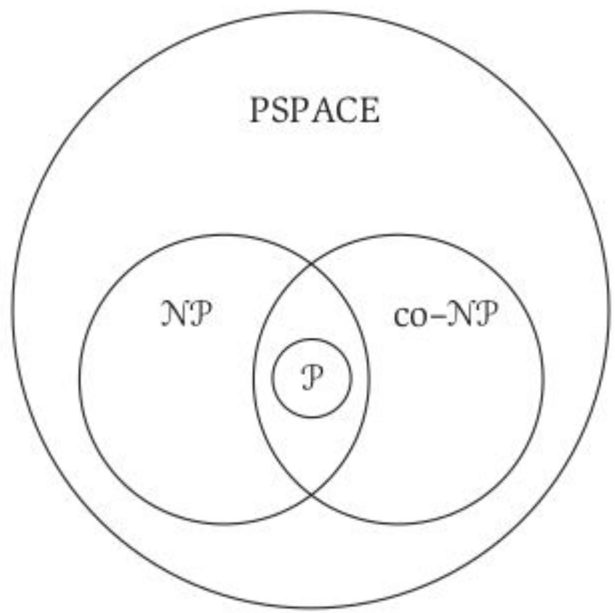
There is an algorithm that solves 3-SAT using only a polynomial amount of space

- Brute-force algorithm that tries all possible truth assignment
- Increment *n*-bit counter from 0 to 2^n-1
- Counter holding value ‘q’ then x_i take that value of i^{th} bit in q
- We spend only polynomial space in checking the truth assignment

NP $\subseteq$ ***PSPACE***

- Consider problem **Y** in ***NP***
- Since **Y≤3-SAT**,there exist an algorithm that solves **Y** in polynomial number of steps
- **Y** uses only polynomial space

Subset relationship



Quantification

Let $\boldsymbol{\Phi}(x_1, x_2, \ldots x_n)$ be a boolean formula of form

$C_1 \wedge C_2 \wedge C_3 \wedge \ldots \wedge C_k$

3-SAT

$$\exists x_1 \exists x_2 \cdots \exists x_{n-2} \exists x_{n-1} \exists x_n \Phi(x_1, \ldots, x_n)?$$

QSAT

$$\exists x_1 \forall x_2 \cdots \exists x_{n-2} \forall x_{n-1} \exists x_n \Phi(x_1, \ldots, x_n)?$$

Example

$$\Phi(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})$$

$$\exists x_1 \forall x_2 \exists x_3 \Phi(x_1, x_2, x_3)?$$

Reference

Jon Kleinberg and Éva Tardos. 2006. *Algorithm Design*

Thank you!

Q&A