Outline

- the class co-NP
- the class co-NP ∩ NP
- the class PSPACE
  - a PSPACE-complete problem
  - PSPACE and 2-player games
Quantifier characterization of coNP

- recall that a language \( L \) is in NP if and only if it is expressible as:
  \[
  L = \{ x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \}
  \]
  where \( R \) is a language in P.

**Theorem:** language \( L \) is in coNP if and only if it is expressible as:

\[
L = \{ x \mid \forall y, |y| \leq |x|^k, (x, y) \in R \}
\]

where \( R \) is a language in P.
Proof interpretation of coNP

- What is a proof?
- Good formalization comes from NP:
  \[ L = \{ x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \}, \text{ and } R \in P \]
  “proof” “short” proof “proof verifier”

- NP languages have short proofs of membership
- co-NP languages have short proofs of non-membership
- coNP-complete languages are least likely to have short proofs of membership
coNP

• what complexity class do the following languages belong in?
  – COMPOSITES = \{x : \text{integer } x \text{ is a composite}\}
  – PRIMES = \{x : \text{integer } x \text{ is a prime number}\}
  – GRAPH-NONISOMORPHISM = \{(G, H) : G \text{ and } H \text{ are graphs that are not isomorphic}\}
  – EXPANSION = \{(G = (V,E), \alpha > 0) : \text{every subset } S \subset V \text{ of size at most } |V|/2 \text{ has at least } \alpha|S| \text{ neighbors}\}
coNP

- Picture of the way we believe things are:

\[ \text{P} \subseteq \text{NP} \cap \text{coNP} \subset \text{EXP} \]

decidable languages

\[ \text{coNP} \]

\[ \text{NP} \]

\[ \text{NP} \cap \text{coNP} \]
NP \cap \text{coNP}

- Might guess $NP \cap \text{coNP} = P$ by analogy with $RE$ (since $RE \cap \text{coRE} = \text{DECIDABLE}$)

- Not believed to be true.

- A problem in $NP \cap \text{coNP}$ not believed to be in $P$:
  \[ L = \{(x, k) : \text{integer } x \text{ has a prime factor } p < k\} \]
  (decision version of factoring)
NP \cap \text{coNP}

- **Theorem**: This language is in \( NP \cap \text{coNP} \):
  \[
  L = \{(x, k): \text{integer } x \text{ has a prime factor } p < k\}
  \]

**Proof:**
- In \( NP \) (why?)
- In \( \text{coNP} \) (what certificate demonstrates that \( x \) has no small prime factor?)
- Use this claim: PRIMES is in \( NP \):
  \[
  \text{PRIMES} = \{x : \forall 1 < y < x, \text{ y does not divide } x\}
  \]
PRIMES in NP

**Theorem**: (Pratt 1975) PRIMES is in NP.
PRIMES = \{x : \forall 1 < y < x, y \text{ does not divide } x\}

- **Proof outline**:
  - Step 1: give “∃” characterization of PRIMES
  - Step 2: this \implies short certificate of primality
  - Step 3: certificate checkable in poly time

  *(we will skip, because…)*

**Theorem**: (M. Agrawal, N. Kayal, N. Saxena 2002)
PRIMES is in P.
Summary

• Picture of the way we believe things are:

\[(\text{decision version of } \text{FACTORING})\]

P ⊆ NP ∩ coNP ⊆ EXP ⊆ coNP

decidable languages
Space complexity

**Definition**: the space complexity of a TM M is a function

\[ f: \mathbb{N} \rightarrow \mathbb{N} \]

where \( f(n) \) is the maximum number of tape cells M scans on any input of length n.

• “M uses space \( f(n) \),” “M is a \( f(n) \) space TM”
Space complexity

**Definition:** \( \text{SPACE}(t(n)) = \{L : \text{there exists a TM } M \text{ that decides } L \text{ in space } O(t(n))\} \)

\[ \text{PSPACE} = \bigcup_{k \geq 1} \text{SPACE}(n^k) \]
• NP ⊆ PSPACE, coNP ⊆ PSPACE (proof?)
• PSPACE ⊆ EXP (proof?)
• containments believed to be proper
PSPACE

• A PSPACE-complete problem:

• Quantified Satisfiability:

$$\text{QSAT} = \{ \varphi : \varphi \text{ is a 3-CNF, and } \exists x_1 \forall x_2 \exists x_3 \forall x_4 \exists x_5 \ldots \forall x_n \varphi(x_1, x_2, x_3, \ldots x_n) \}$$

• example: \(\varphi = (x_1 \lor x_2 \lor \neg x_3) \land (\neg x_2 \lor \neg x_3)\)

\[\exists x_1 \forall x_2 \exists x_3 \varphi?\]

YES: \(x_1 = T\); if \(x_2 = T\), set \(x_3 = F\); if \(x_2 = F\), set \(x_3 = T\)
PSPACE

• A PSPACE-complete problem:

• Quantified Satisfiability:

  \[ \text{QSAT} = \{ \phi : \phi \text{ is a 3-CNF, and} \]
  \[ \exists x_1 \forall x_2 \exists x_3 \forall x_4 \exists x_5 \ldots \forall x_n \phi(x_1, x_2, x_3, \ldots x_n) \} \]

• example: \( \phi = (x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_2) \)

  \[ \exists x_1 \forall x_2 \exists x_3 \phi? \]

  NO: \( x_1 = T; \) if \( x_2 = T \ldots; \quad x_1 = F; \) if \( x_2 = T \ldots \)
QSAT is **PSPACE**-complete

**Theorem:** QSAT is **PSPACE**-complete.

• **Proof:**
  - in **PSPACE:** \( \exists x_1 \forall x_2 \exists x_3 \ldots Qx_n \varphi(x_1, x_2, \ldots, x_n) \)?
  - “\( \exists x_1 \)”: for both \( x_1 = 0, x_1 = 1 \), recursively solve
    \( \forall x_2 \exists x_3 \ldots Qx_n \varphi(x_1, x_2, \ldots, x_n) \)?
    • if at least one “yes”, return “yes”; else return “no”
  - “\( \forall x_1 \)”: for both \( x_1 = 0, x_1 = 1 \), recursively solve
    \( \exists x_2 \forall x_3 \ldots Qx_n \varphi(x_1, x_2, \ldots, x_n) \)?
    • if at least one “no”, return “no”; else return “yes”
  - base case: evaluating a 3-CNF expression
  - poly(n) recursion depth
  - poly(n) bits of state at each level