CS21 Lecture 23

March 1, 2023

Outline

• The complexity class coNP
• The complexity class coNP ∩ NP
• The complexity class PSPACE
  – a PSPACE-complete problem
  – PSPACE and 2-player games

3

March 1, 2023

CS21 Lecture 23

MAX CUT is NP-complete

Claim: if cut has \( x_i \), \( \neg x_i \) on same side, then can move one to opposite side without decreasing # edges crossing cut

• Proof:

\[
\begin{align*}
\text{a edges} & \quad \text{b edges} \\
\text{n edges} & \quad \text{b edges}
\end{align*}
\]

\[
\begin{align*}
a + n & \leq \frac{a+b}{2} + n \\
ax + bx + ay + by & \geq a + b \\

\text{a + b} & \geq a + b
\end{align*}
\]

4

March 1, 2023

CS21 Lecture 23

coNP

• Is NP closed under complement?

Can we transform this machine:

\[
\begin{align*}
x \in L & \quad x \in \overline{L} \\
x \not\in L & \quad x \not\in \overline{L}
\end{align*}
\]

into a machine with this behavior?

5

March 1, 2023

CS21 Lecture 23

coNP

• language \( L \) is in coNP iff its complement (\( \overline{L} \)) is in NP

• it is believed that \( \text{NP} \neq \text{coNP} \)
• note: \( P = \text{NP} \) implies \( P = \text{coNP} \)
  – proving \( \text{NP} \neq \text{coNP} \) would prove \( P \neq \text{NP} \)
  – another major open problem...

6

March 1, 2023

CS21 Lecture 23

coNP

• canonical coNP-complete language:

\( \text{UNSAT} = \{ \varphi : \varphi \text{ is an unsatisfiable 3-CNF formula} \} \)

– proof?
Quantifier characterization of coNP

• recall that a language $L$ is in NP if and only if it is expressible as:
  $L = \{ (x, y) : |y| \leq |x|^p, (x, y) \in R \}$
  where $R$ is a language in P.

**Theorem:** language $L$ is in coNP if and only if it is expressible as:
  $L = \{ x \mid \forall y, |y| \leq |x|^p, (x, y) \in R \}$
  where $R$ is a language in P.

Proof interpretation of coNP

• What is a proof?
  Good formalization comes from NP:
  $L = \{ x \mid \exists y, |y| \leq |x|, (x, y) \in R \}$, and $R \subseteq P$
  "proof" = "short" proof "proof verifier"

• NP languages have short proofs of membership
• coNP 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 = \{ integer $x$ is a composite \}
  - PRIMES = \{ integer $x$ is a prime number \}
  - GRAPH-NONISOMORPHISM = \{ (G, H) : G and H are graphs that are not isomorphic \}
  - EXPANSION = \{ (G = (V, E), \alpha > 0) : every subset $S \subseteq V$ of size at most $|V|/2$ has at least $\alpha |S|$ neighbors \}

coNP

• Picture of the way we believe things are:

NP ∩ coNP

• Might guess NP ∩ coNP = P by analogy with RE (since RE ∩ coRE = DECIDABLE)

• Not believed to be true.
  • A problem in NP ∩ 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 \text{ has no small prime factor?)
  - Use this claim: PRIMES is in NP:
    PRIMES = \{ x : \forall 1 < y < x, y \text{ 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 \(\exists^*\) characterization of PRIMES
  - Step 2: this \(\Rightarrow\) 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:
  \[(\text{decision version of})\]
  \[\text{FACTORING} \quad \text{EXP} \quad \text{coNP} \quad \text{P} \quad \text{NP} \cap \text{coNP} \quad \text{NP} \]

Space complexity

- **Definition**: the space complexity of a TM M is a function
  \[ f : \mathbb{N} \to \mathbb{N} \]
  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**: SPACE(\( t(n) \)) = \{ L : \text{there exists a TM M that decides } L \text{ in space } O(t(n)) \}
  PSPACE = \( \bigcup_{k \geq 1} \) SPACE(\( n^k \))

- NP \subseteq PSPACE, coNP \subseteq PSPACE (proof?)
- PSPACE \subseteq EXP (proof?)
- containments believed to be proper