CS21 Decidability and Tractability

Lecture 23

February 28, 2024

1

coNP

- language L is in coNP iff its complement (co-L) is in NP
- it is believed that NP ≠ coNP
- note: P = NP implies NP = coNP
  - proving NP ≠ coNP would prove P ≠ NP
  - another major open problem…

2

coNP

- canonical coNP-complete language:

  UNSAT = {φ : φ is an unsatisfiable 3-CNF formula}

  – proof?

3

coNP

Disjunctive Normal Form = OR of ANDs

- another example

  3-DNF-TAUTOLOGY = {φ : φ is a 3-DNF formula and for all x, φ(x) =1}

  – proof?
- another example:

  EQUIV-CIRCUIT = $\{(C_1, C_2) : C_1$ and C_2 are Boolean circuits and for all x, $C_1(x) = C_2(x)\}$

  – proof?

4

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.

5

Proof interpretation of coNP

- What is a proof?
- Good formalization comes from NP:

  $L = \{x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \}$, 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

6

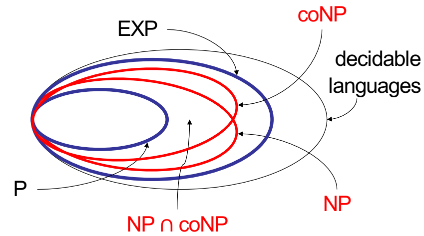
coNP

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

7

coNP

- Picture of the way we believe things are:



8

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): integer x has a prime factor p < k}

  (decision version of factoring)

9

NP ∩ coNP

- **Theorem**: This language is in NP ∩ coNP:

  L = {(x, k): integer x has a prime factor p < k}

Proof:
- In NP (why?)
- In coNP (what certificate demonstrates that x has *no* small prime factor?)
- Use this claim: PRIMES is in NP:

  PRIMES = {x : ∀ 1 < y < x, y does not divide x}

10

PRIMES in NP

Theorem: (Pratt 1975) PRIMES is in NP.

PRIMES = {x : ∀ 1 < y < x, y does not divide x}

- Proof outline:
  - Step 1: give "∃" characterization of PRIMES
  - Step 2: this $\Longrightarrow$ short certificate of primality
  - Step 3: certificate checkable in poly time

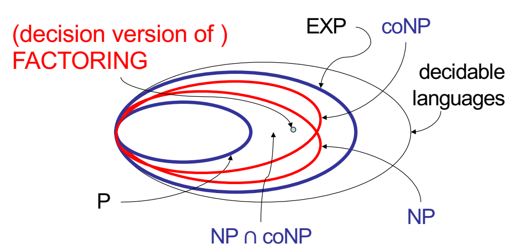
  (we will skip, because…)

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

11

Summary

- Picture of the way we believe things are:



12

Space complexity

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

$$f: \mathbf{N} \to \mathbf{N}$$

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"

13

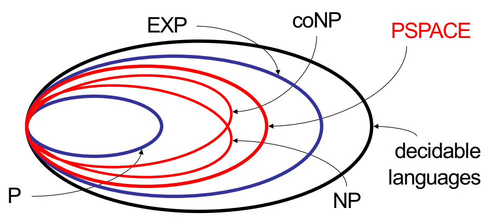
Space complexity

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

$$\text{PSPACE} = \cup_{k \geq 1} \text{SPACE}(n^k)$$

14

PSPACE



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

15

PSPACE

- A PSPACE-complete problem:
- Quantified Satisfiability:

QSAT = {φ : φ 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: $\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_2 \vee \neg x_3)$

$\exists x_1 \, \forall x_2 \, \exists x_3 \, \varphi$?

YES: x_1=T; if x_2=T, set x_3=F; if x_2=F, set x_3=T

16

PSPACE

- A PSPACE-complete problem:
- Quantified Satisfiability:

QSAT = {φ : φ 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: $\varphi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_2)$

$\exists x_1 \, \forall x_2 \, \exists x_3 \, \varphi$?

NO: x_1=T; if x_2=T…; x_1=F; if x_2=T…

17

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

18

QSAT is **PSPACE**-complete

- given TM M deciding $L \in$ **PSPACE**; input x
- 2^{n^k} possible configurations
- single START configuration
- assume single ACCEPT configuration

- define:

REACH(X, Y, i) $\Leftrightarrow$ configuration Y reachable from configuration X in at most 2^i steps.

19

QSAT is **PSPACE**-complete

REACH(X, Y, i) $\Leftrightarrow$ configuration Y reachable from configuration X in at most 2^i steps.

- Goal: produce 3-CNF $\varphi(w_1, w_2, w_3, \ldots, w_m)$ such that

$$\exists w_1 \forall w_2 \ldots \exists w_m \, \varphi(w_1, \ldots, w_m)$$
$$\Leftrightarrow \text{REACH(START, ACCEPT, } n^k)$$

20

QSAT is **PSPACE**-complete

- for i = 0, 1, … n^k produce quantified Boolean **expressions** $\psi_i(A, B, W)$ such that $\forall$ A,B:

$$\exists w_1 \forall w_2 \ldots \psi_i(A, B, W) \Leftrightarrow \text{REACH}(A, B, i)$$

- convert ψ_{n^k} to 3-CNF φ
  - add variables V
- hardwire A = START, B = ACCEPT

$$\exists w_1 \forall w_2 \ldots \exists V \, \varphi(W, V) \Leftrightarrow x \in L$$

21