Foundations of Computing

Lecture 23

Arkady Yerukhimovich

April 18, 2023

Outline

Lecture 22 Review

- More $\mathcal{NP}$-complete problems
- The class co-$\mathcal{NP}$

Outline

Are All Problems in $\mathcal{NP}$?

Question

Do all languages have poly-size proofs?

Are All Problems in $\mathcal{NP}$?

Question

Do all languages have poly-size proofs?

Consider the following language:

UNSAT

$$\text{UNSAT} = \{\langle\phi\rangle \mid \phi \text{ is not satisfiable}\}$$

Are All Problems in $\mathcal{NP}$?

Question

Do all languages have poly-size proofs?

Consider the following language:

UNSAT

$$\text{UNSAT} = \{\langle \phi \rangle \mid \phi \text{ is not satisfiable}\}$$

- For all possible assignments $w \in \{0,1\}^{|\phi|}$, $\phi(w) = 0$

Are All Problems in $\mathcal{NP}$?

Question

Do all languages have poly-size proofs?

Consider the following language:

UNSAT

$$\text{UNSAT} = \{\langle \phi \rangle \mid \phi \text{ is not satisfiable}\}$$

- For all possible assignments $w \in \{0,1\}^{|\phi|}$, $\phi(w) = 0$
- We define co-$\mathcal{NP}$ to contain all such languages that are complements of languages in $\mathcal{NP}$

$\mathcal{P}$, $\mathcal{NP}$ and co-$\mathcal{NP}$

$\mathcal{P}$

$L \in \mathcal{P}$ if there exists poly-time DTM M s.t $M(x) = [x \in L]$

$\mathcal{P}$, $\mathcal{NP}$ and co-$\mathcal{NP}$

$\mathcal{P}$

$L \in \mathcal{P}$ if there exists poly-time DTM M s.t $M(x) = [x \in L]$

$\mathcal{NP}$

$L \in \mathcal{NP}$ if there exists poly-time DTM V s.t. for $x \in L$ there exists a witness w s.t. $V(x, w) = 1$

$\mathcal{P}$, $\mathcal{NP}$ and co-$\mathcal{NP}$

$\mathcal{P}$

$L \in \mathcal{P}$ if there exists poly-time DTM M s.t $M(x) = [x \in L]$

$\mathcal{NP}$

$L \in \mathcal{NP}$ if there exists poly-time DTM V s.t. for $x \in L$ there exists a witness w s.t. $V(x, w) = 1$

co-$\mathcal{NP}$

$L \in$ co-$\mathcal{NP}$ if there exists poly-time DTM V s.t. for $x \in L$ for all w, $V(x, w) = 0$

$\mathcal{P}$, $\mathcal{NP}$ and co-$\mathcal{NP}$

$\mathcal{P}$

$L \in \mathcal{P}$ if there exists poly-time DTM M s.t $M(x) = [x \in L]$

$\mathcal{NP}$

$L \in \mathcal{NP}$ if there exists poly-time DTM V s.t. for $x \in L$ there exists a witness w s.t. $V(x, w) = 1$

co-$\mathcal{NP}$

$L \in$ co-$\mathcal{NP}$ if there exists poly-time DTM V s.t. for $x \in L$ for all w, $V(x, w) = 0$

Question:

Can you prove that $x \in L$, when $L \in$co-$\mathcal{NP}$?

Proving that $x \in L$ for $L \in$co-$\mathcal{NP}$

The Problem

Suppose, I am given an input formula ϕ and I want to prove that ϕ is not satisfiable.

Proving that $x \in L$ for $L \in$co-$\mathcal{NP}$

The Problem

Suppose, I am given an input formula ϕ and I want to prove that ϕ is not satisfiable.

- It is widely believed that there is no poly-size, efficiently verifiable proof w that you could give for UNSAT

Proving that $x \in L$ for $L \in$co-$\mathcal{NP}$

The Problem

Suppose, I am given an input formula ϕ and I want to prove that ϕ is not satisfiable.

- It is widely believed that there is no poly-size, efficiently verifiable proof w that you could give for UNSAT
- $\mathcal{NP} \neq$ co-$\mathcal{NP}$

Outline

What is a Proof?

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

- x is a satisfiable formula

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

- x is a satisfiable formula
- The Pythagorean Theorem is true

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

- x is a satisfiable formula
- The Pythagorean Theorem is true
- $\ldots$

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

- x is a satisfiable formula
- The Pythagorean Theorem is true
- ...

New Definition

A proof is any process at the end of which one party (the prover) can convince the other party (the verifier) of the truth of some statement x

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

- x is a satisfiable formula
- The Pythagorean Theorem is true
- ...

New Definition

A proof is any process at the end of which one party (the prover) can convince the other party (the verifier) of the truth of some statement x

- A proof doesn't have to be a string

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

- x is a satisfiable formula
- The Pythagorean Theorem is true
- ...

New Definition

A proof is any process at the end of which one party (the prover) can convince the other party (the verifier) of the truth of some statement x

- A proof doesn't have to be a string
- Can be an interactive procedure

What is a Proof?

Traditional Definition

A proof is a string that convinces us of the truth of some mathematical statement x

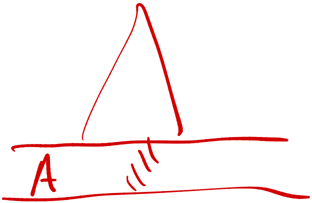
- x is a satisfiable formula
- The Pythagorean Theorem is true
- $\ldots$

New Definition

A proof is any process at the end of which one party (the prover) can convince the other party (the verifier) of the truth of some statement x

- A proof doesn't have to be a string
- Can be an interactive procedure
- The verifier (and prover) can use randomness to decide whether to accept

An Example – Aladdin's Cave



S: flip a coin
if heads, come out on L
if tails, come out on R

Outline

The Class $\mathcal{IP}$

Definition

$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

The Class $\mathcal{IP}$

unbounded poly-time

Definition

$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

1. (Completeness) If $x \in L$, then $\Pr[\underline{\langle P, V\rangle(x)} = 1] = 1$

The Class $\mathcal{IP}$

Definition

$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

1. (Completeness) If $x \in L$, then $\Pr[\langle P, V\rangle(x) = 1] = 1$
2. (Soundness) If $x \notin L$, then for any (possibly unbounded) P^*, we have $\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/2$

The Class $\mathcal{IP}$

Definition

$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

1. (Completeness) If $x \in L$, then $\Pr[\langle P, V\rangle(x) = 1] = 1$
2. (Soundness) If $x \notin L$, then for any (possibly unbounded) P^*, we have $\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/2$

Examples:

The Class $\mathcal{IP}$

Definition

$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

1. (Completeness) If $x \in L$, then $\Pr[\langle P, V\rangle(x) = 1] = 1$
2. (Soundness) If $x \notin L$, then for any (possibly unbounded) P^*, we have $\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/2$

Examples:

- Aladdin's cave example from earlier

The Class $\mathcal{IP}$

Definition

$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

1. (Completeness) If $x \in L$, then $\Pr[\langle P, V\rangle(x) = 1] = 1$
2. (Soundness) If $x \notin L$, then for any (possibly unbounded) P^*, we have $\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/2$

Examples:

- Aladdin's cave example from earlier
- $\mathcal{P} \subseteq \mathcal{IP}$

The Class $\mathcal{IP}$

Definition

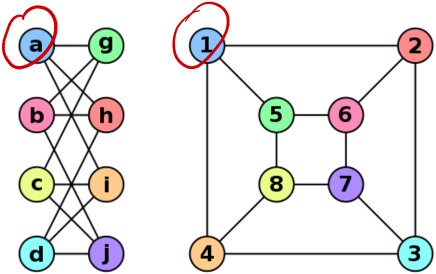
$L \in \mathcal{IP}$ if there exist a pair of interactive algorithms (P, V) with V being poly-time (in $|x|$) s.t.

1. (Completeness) If $x \in L$, then $\Pr[\langle P, V\rangle(x) = 1] = 1$
2. (Soundness) If $x \notin L$, then for any (possibly unbounded) P^*, we have $\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/2$

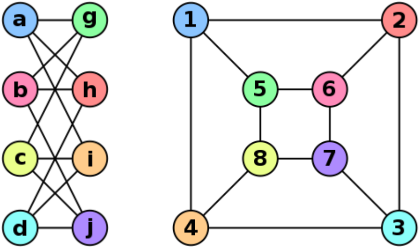
Examples:

- Aladdin's cave example from earlier
- $\mathcal{P} \subseteq \mathcal{IP}$
- $\mathcal{NP} \subseteq \mathcal{IP}$

Another Example – Graph Isomorphism



Another Example – Graph Isomorphism



Claim

Graph Isomorphism $\in \mathcal{IP}$

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0,1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0,1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0,1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0,1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
   - Then G^* can only be isomorphic to one of the two graphs

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
    - Then G^* can only be isomorphic to one of the two graphs
    - P can perfectly determine which one this is

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
    - Then G^* can only be isomorphic to one of the two graphs
    - P can perfectly determine which one this is
    - So $\Pr[b' = b] = 1$

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
   - Then G^* can only be isomorphic to one of the two graphs
   - P can perfectly determine which one this is
   - So $\Pr[b' = b] = 1$
2. (Soundness) Suppose that G_0 and G_1 are isomorphic

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
   - Then G^* can only be isomorphic to one of the two graphs
   - P can perfectly determine which one this is
   - So $\Pr[b' = b] = 1$
2. (Soundness) Suppose that G_0 and G_1 are isomorphic
   - Then G^* is isomorphic to both G_0 and G_1

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
   - Then G^* can only be isomorphic to one of the two graphs
   - P can perfectly determine which one this is
   - So $\Pr[b' = b] = 1$
2. (Soundness) Suppose that G_0 and G_1 are isomorphic
   - Then G^* is isomorphic to both G_0 and G_1
   - P has no way to tell which one V started from

Graph Non-Isomorphism

Question

How can we prove that two graphs G_0 and G_1 are NOT isomorphic?

The Protocol:

1. V chooses $b \leftarrow \{0, 1\}$, and applies a random permutation π to the vertices of G_b and sends this graph G^* to P
2. P determines if G^* is isomorphic to G_0 and sends $b' = 0$ if so, or $b' = 1$ otherwise back to V
3. V accepts if $b' = b$

Why This Works:

1. (Completeness) Suppose that G_0 and G_1 are not isomorphic.
   - Then G^* can only be isomorphic to one of the two graphs
   - P can perfectly determine which one this is
   - So $\Pr[b' = b] = 1$
2. (Soundness) Suppose that G_0 and G_1 are isomorphic
   - Then G^* is isomorphic to both G_0 and G_1
   - P has no way to tell which one V started from
   - Thus, $\Pr[b' = b] = 1/2$

Important Takeaways

Important Takeaways

- GNI $\in$ co-$\mathcal{NP}$

Important Takeaways

- GNI $\in$ co-$\mathcal{NP}$
- It is not believed that there is a short witness w s.t. $V((G_0, G_1), w) = 1$ if G_0 and G_1 are not isomorphic. I.e., GNI $\notin \mathcal{NP}$

Important Takeaways

- GNI $\in$ co-$\mathcal{NP}$
- It is not believed that there is a short witness w s.t. $V((G_0, G_1), w) = 1$ if G_0 and G_1 are not isomorphic.
  I.e., GNI $\notin \mathcal{NP}$
- The power of interaction and randomness has allowed us to do what we couldn't do before

Boosting Soundness

So far, we defined soundness as:

$$\Pr[\langle P^*, v\rangle(x) = 1] \leq 1/2$$

Boosting Soundness

So far, we defined soundness as:

$$\Pr[\langle P^*, v\rangle(x) = 1] \leq 1/2$$

What if we don't want malicious prover to win so often?

Boosting Soundness

So far, we defined soundness as:

$$\Pr[\langle P^*, v\rangle(x) = 1] \leq 1/2$$

What if we don't want malicious prover to win so often?

Soundness Amplification

Boosting Soundness

So far, we defined soundness as:

$$\Pr[\langle P^*, v\rangle(x) = 1] \leq 1/2$$

What if we don't want malicious prover to win so often?

Soundness Amplification

1. Run the proof n times sequentially on same input x, but different randomness

Boosting Soundness

So far, we defined soundness as:

$$\Pr[\langle P^*, v\rangle(x) = 1] \leq 1/2$$

What if we don't want malicious prover to win so often?

Soundness Amplification

1. Run the proof n times sequentially on same input x, but different randomness
2. Accept if ALL proofs accept

Boosting Soundness

So far, we defined soundness as:

$$\Pr[\langle P^*, v\rangle(x) = 1] \leq 1/2$$

What if we don't want malicious prover to win so often?

Soundness Amplification

1. Run the proof n times sequentially on same input x, but different randomness
2. Accept if ALL proofs accept
3. P^* wins with probability $\leq 1/2$ in each run, so

$$\Pr[\langle P^*, V\rangle(x) = 1] \leq 1/2^n$$