Foundations of Computing

Lecture 22

Arkady Yerukhimovich

April 13, 2023

Outline

1. Lecture 21 Review
2. More $\mathcal{NP}$-Complete Problems
3. co-$\mathcal{NP}$

Lecture 21 Review

- $\mathcal{P}$ and $\mathcal{NP}$
- Polynomial-Time Reductions
- $\mathcal{NP}$-completeness of SAT

Outline

1 Lecture 21 Review

2 More $\mathcal{NP}$-Complete Problems

3 co-$\mathcal{NP}$

What We Already Know

1. SAT is $\mathcal{NP}$-complete

What We Already Know

1. SAT is $\mathcal{NP}$-complete
2. 3-SAT is $\mathcal{NP}$-complete

What We Already Know

1. SAT is $\mathcal{NP}$-complete
2. 3-SAT is $\mathcal{NP}$-complete
3. 3-SAT $\leq_P$ CLIQUE – So CLIQUE in $\mathcal{NP}$-complete

What We Already Know

1. SAT is $\mathcal{NP}$-complete
2. 3-SAT is $\mathcal{NP}$-complete
3. 3-SAT $\leq_P$ CLIQUE – So CLIQUE in $\mathcal{NP}$-complete
4. 3-SAT $\leq_P$ Vertex Cover

What We Already Know

1. SAT is $\mathcal{NP}$-complete
2. 3-SAT is $\mathcal{NP}$-complete
3. 3-SAT $\leq_P$ CLIQUE – So CLIQUE in $\mathcal{NP}$-complete
4. 3-SAT $\leq_P$ Vertex Cover
5. Vertex Cover $\leq_P$ Independent Set

What We Already Know

1. SAT is $\mathcal{NP}$-complete
2. 3-SAT is $\mathcal{NP}$-complete
3. 3-SAT $\leq_P$ CLIQUE – So CLIQUE in $\mathcal{NP}$-complete
4. 3-SAT $\leq_P$ Vertex Cover
5. Vertex Cover $\leq_P$ Independent Set
6. More on the HW

A Key Tool to Build Reductions

A Key Tool to Build Reductions

Gadgets

A Key Tool to Build Reductions

Gadgets

- Gadgets are structures in the target problem that can simulate structures in the source problem

A Key Tool to Build Reductions

Gadgets

- Gadgets are structures in the target problem that can simulate structures in the source problem
- For example, in proof of 3SAT $\leq_P$ CLIQUE
  - We replaced each clause with set of 3 nodes
  - Edges were used to capture common variables between clauses

A Key Tool to Build Reductions

Gadgets

- Gadgets are structures in the target problem that can simulate structures in the source problem
- For example, in proof of 3SAT $\leq_P$ CLIQUE
  - We replaced each clause with set of 3 nodes
  - Edges were used to capture common variables between clauses
- Similarly in proof of 3SAT $\leq_P$ Vertex Cover
  - We replaced each clause with a triangle and

A Key Tool to Build Reductions

Gadgets

- Gadgets are structures in the target problem that can simulate structures in the source problem
- For example, in proof of 3SAT $\leq_P$ CLIQUE
  - We replaced each clause with set of 3 nodes
  - Edges were used to capture common variables between clauses
- Similarly in proof of 3SAT $\leq_P$ Vertex Cover
  - We replaced each clause with a triangle and
  - each variable with a pair of nodes connected by an edge

NAE-3SAT

NAE-kSAT Problem

$$\text{NAE-kSAT} = \{\langle\phi\rangle \mid \phi \text{ is in } k\text{-CNF and } \phi \text{ has a satisfying assignment s.t. each clause has at least one 0 and at least one 1}\}$$

$(x_1 \vee x_2 \vee x_3 \quad) \wedge (y_1 \vee y_2 \vee y_3)$

NAE-3SAT

NAE-kSAT Problem

NAE-kSAT $= \{\langle\phi\rangle \mid \phi$ is in k-CNF and ϕ has a satisfying assignment s.t. each clause has at least one 0 and at least one 1$\}$

Definition:

- x is an NAE-assignment of ϕ if $\phi(x) = 1$ and x does not assign all the same variables to any clause

Lemma: If x is NAE-assignment

$\overline{x}$ is also NAE-assignment

NAE-3SAT

NAE-kSAT Problem

$$\text{NAE-kSAT} = \{\langle\phi\rangle \;\mid\; \phi \text{ is in } k\text{-CNF and } \phi \text{ has a satisfying assignment s.t. each clause has at least one 0 and at least one 1}\}$$

Definition:

- x is an NAE-assignment of ϕ if $\phi(x) = 1$ and x does not assign all the same variables to any clause

Lemma: If x is NAE-assignment of ϕ then $\overline{x}$ is NAE-assignment of ϕ

Proof:

- x must assign at least one 1 and at least one 0 to every clause
- $\overline{x}$ must also have at least one 1 and one 0 in every clause
- This means every clause is satisfied, and ϕ is satisfied since it's CNF

Goal

Prove that NAE-3SAT is $\mathcal{NP}$-complete: 3SAT $\leq_P$ NAE-3SAT

3SAT $\leq_P$ NAE-4SAT

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable

$A \Rightarrow B$

$\neg B \Rightarrow \neg A$

f is s.t.

if $x \in 3SAT \Rightarrow f(x) \in NAE\text{-}4SAT$

if $x \notin 3SAT \Rightarrow f(x) \notin NAE\text{-}4SAT$

$\Updownarrow$

$\rightarrow$ if $f(x) \in NAE\text{-}4SAT \Rightarrow x \in 3SAT$

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow$$

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow (x_1 \vee x_2 \vee x_3 \vee S)$$

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \to (x_1 \vee x_2 \vee x_3 \vee S)$$

- Why this works:

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow (x_1 \vee x_2 \vee x_3 \vee S)$$

- Why this works:
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3) = 1$

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow (x_1 \vee x_2 \vee x_3 \vee S)$$

- Why this works:
  - ($\Rightarrow$) If $(x_1 \vee x_2 \vee x_3) = 1$ at least one $x_i = 1$, so $(x_1 \vee x_2 \vee x_3 \vee S) = 1$. Set $S = 0$ to make it NAE-assignment

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow (x_1 \vee x_2 \vee x_3 \vee S)$$

- Why this works:
  - ($\Rightarrow$) If $(x_1 \vee x_2 \vee x_3) = 1$ at least one $x_i = 1$, so $(x_1 \vee x_2 \vee x_3 \vee S) = 1$. Set $S = 0$ to make it NAE-assignment
  - ($\Leftarrow$) If $(x_1 \vee x_2 \vee x_3 \vee S) = 1$

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow (x_1 \vee x_2 \vee x_3 \vee S)$$

- Why this works:
  - ($\Rightarrow$) If $(x_1 \vee x_2 \vee x_3) = 1$ at least one $x_i = 1$, so $(x_1 \vee x_2 \vee x_3 \vee S) = 1$. Set $S = 0$ to make it NAE-assignment
  - ($\Leftarrow$) If $(x_1 \vee x_2 \vee x_3 \vee S) = 1$
    - If $S = 0$, then at least one $x_i = 1$, so $(x_1 \vee x_2 \vee x_3) = 1$

3SAT $\leq_P$ NAE-4SAT

- We need a reduction f that takes 3SAT instance ϕ and converts it into NAE-4SAT instance ϕ'
  - If ϕ is satisfiable, ϕ' is NAE-satisfiable
  - If ϕ' is NAE-satisfiable, ϕ is satisfiable
  - Note that this must hold for every clause of ϕ, ϕ'
- Idea: Can we build a "gadget" for each clause of ϕ to enforce this condition?

$$(x_1 \vee x_2 \vee x_3) \rightarrow (x_1 \vee x_2 \vee x_3 \vee S)$$

- Why this works:
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3) = 1$ at least one $x_i = 1$, so $(x_1 \vee x_2 \vee x_3 \vee S) = 1$. Set $S = 0$ to make it NAE-assignment
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee S) = 1$
    - If $S = 0$, then at least one $x_i = 1$, so $(x_1 \vee x_2 \vee x_3) = 1$
    - If $S = 1$, then $(\overline{x_1} \vee \overline{x_2} \vee \overline{x_3} \vee 0)$ is also NAE-assignment. So, $(\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) = 1$

NAE-4SAT $\leq_P$ NAE-3SAT

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
    - We know that not all x_i have the same value
    - At least one of x_i is a 1 and one is a 0

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(\overset{0}{x_1} \vee \overset{0}{x_2} \vee \overset{1}{x_3} \vee \overset{1}{x_4})$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee z) \wedge (\overset{1}{x_3} \vee \overset{1}{x_4} \vee \bar{z})$$

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee x_3 \vee x_4) \to (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:
    $$(x_1 \vee x_2 \vee x_3 \vee x_4) \to (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$
- Why this works:

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow \underline{(x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})}$$

- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$

- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE, then $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:
    $$(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$
- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE, then $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
    - If $x_1 \neq x_2$:

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$

- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE, then $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
    - If $x_1 \neq x_2$:Set $z_i = x_3$

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:
$$(x_1 \vee x_2 \vee x_3 \vee x_4) \to (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$
- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE, then $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
    - If $x_1 \neq x_2$:Set $z_i = x_3$
    - If $x_1 \neq x_3$:

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$

- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE, then $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
    - If $x_1 \neq x_2$:Set $z_i = x_3$
    - If $x_1 \neq x_3$:Set $z_i = x_3$

NAE-4SAT $\leq_P$ NAE-3SAT

- Need a gadget to convert 4-CNF clause to CNF clauses that preserves NAE
- Observations: Starting with $(x_1 \vee x_2 \vee x_3 \vee x_4)$
  - We know that not all x_i have the same value
  - At least one of x_i is a 1 and one is a 0
  - Idea: Let's split the variables into two clauses:

$$(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow (x_1 \vee x_2 \vee z_i) \wedge (x_3 \vee x_4 \vee \overline{z_i})$$

- Why this works:
  - $(\Leftarrow)$ If $(x_1 \vee x_2 \vee z_i)$ and $(x_3 \vee x_4 \vee \overline{z_i})$ are both NAE, then $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
  - $(\Rightarrow)$ If $(x_1 \vee x_2 \vee x_3 \vee x_4)$ is NAE
    - If $x_1 \neq x_2$:Set $z_i = x_3$
    - If $x_1 \neq x_3$:Set $z_i = x_3$

Theorem

$$3SAT \leq_P \text{NAE-4SAT} \leq_P \text{NAE-3SAT}$$

3-Coloring

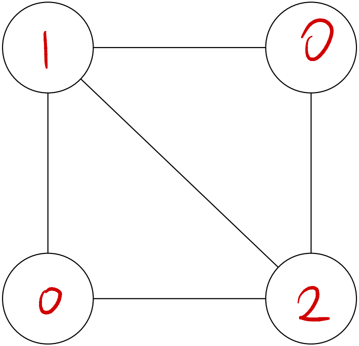
Definition

An undirected graph G is 3-colorable, if can assign colors $\{0, 1, 2\}$ to all nodes, such that no edges have the same color on both ends.

3-Coloring

Definition

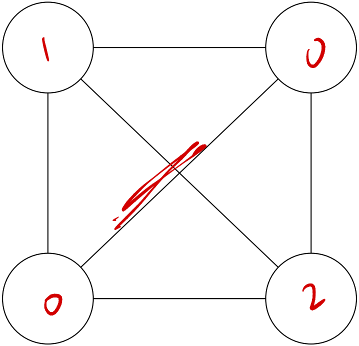
An undirected graph G is 3-colorable, if can assign colors $\{0, 1, 2\}$ to all nodes, such that no edges have the same color on both ends.



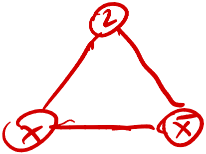
3-Coloring

Definition

An undirected graph G is 3-colorable, if can assign colors $\{0, 1, 2\}$ to all nodes, such that no edges have the same color on both ends.

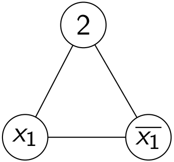


NAE-3SAT $\leq_P$ 3-Coloring



NAE-3SAT $\leq_P$ 3-Coloring

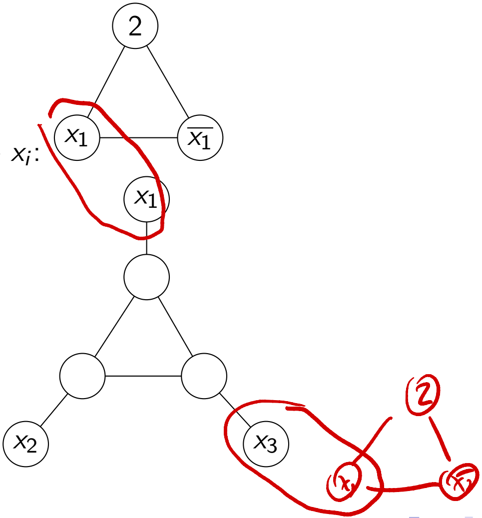
Gadgets:

- For each variable x_i: 

NAE-3SAT $\leq_P$ 3-Coloring

Gadgets:

- For each variable x_i:
- For each clause:



Conclusions

- Many useful problems are $\mathcal{NP}$-complete

Conclusions

- Many useful problems are $\mathcal{NP}$-complete
- But, as long as $\mathcal{P} \neq \mathcal{NP}$, these are hard

Conclusions

- Many useful problems are $\mathcal{NP}$-complete
- But, as long as $\mathcal{P} \neq \mathcal{NP}$, these are hard
- Given a problem L, you should:

Conclusions

- Many useful problems are $\mathcal{NP}$-complete
- But, as long as $\mathcal{P} \neq \mathcal{NP}$, these are hard
- Given a problem L, you should:
    1. Try to solve it ($L \in \mathcal{P}$)

Conclusions

- Many useful problems are $\mathcal{NP}$-complete
- But, as long as $\mathcal{P} \neq \mathcal{NP}$, these are hard
- Given a problem L, you should:
  1. Try to solve it ($L \in \mathcal{P}$)
  2. Try to prove $\mathcal{NP}$-complete

Conclusions

- Many useful problems are $\mathcal{NP}$-complete
- But, as long as $\mathcal{P} \neq \mathcal{NP}$, these are hard
- Given a problem L, you should:
  1. Try to solve it ($L \in \mathcal{P}$)
  2. Try to prove $\mathcal{NP}$-complete
- But, you must be careful

Conclusions

- Many useful problems are $\mathcal{NP}$-complete
- But, as long as $\mathcal{P} \neq \mathcal{NP}$, these are hard
- Given a problem L, you should:
    1. Try to solve it ($L \in \mathcal{P}$)
    2. Try to prove $\mathcal{NP}$-complete
- But, you must be careful

$$\text{3-Coloring is } \mathcal{NP}\text{-complete, but 2-Coloring} \in \mathcal{P}$$

Outline

1. Lecture 21 Review
2. More $\mathcal{NP}$-Complete Problems
3. co-$\mathcal{NP}$

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

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

Problems like UNSAT are in co-$\mathcal{NP}$

$\mathcal{P}$, $\mathcal{NP}$ 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:

$$\text{Is } \mathcal{NP} = \text{co-}\mathcal{NP}?$$

The Complexity Zoo

- There are many other complexity classes

The Complexity Zoo

- There are many other complexity classes
- We know some relationships between classes

The Complexity Zoo

- There are many other complexity classes
- We know some relationships between classes
- But, most big questions (e.g., $\mathcal{P} = \mathcal{NP}$, $\mathcal{NP} =$ co-$\mathcal{NP}$, does PH collapse) are still not known!!!

The Complexity Zoo

- There are many other complexity classes
- We know some relationships between classes
- But, most big questions (e.g., $\mathcal{P} = \mathcal{NP}$, $\mathcal{NP} =$ co-$\mathcal{NP}$, does PH collapse) are still not known!!!

Complexity Zoo

The complexity zoo (`https://complexityzoo.net/Complexity_Zoo`) now has 546 complexity classes.