Lecture 21: Intractability - NP and coNP

Rafael Oliveira

University of Waterloo
Cheriton School of Computer Science

rafael.oliveira.teaching@gmail.com

November 23, 2023

Overview

- Complexity Classes & Complete Problems
  - NP
  - coNP
  - Completeness for NP
- Completing Karp Reductions/Polynomial Transformations
  - NP-completeness of 3SAT
  - Current Worldview
- Acknowledgements

NP

- Let Π be a decision problem and let L_Π be the set of all YES instances of Π. Then $L_\Pi \subseteq \{0,1\}^*$

  decision problems $\leftrightarrow$ subsets of all boolean strings

NP

- Let Π be a decision problem and let L_Π be the set of all YES instances of Π. Then $L_\Pi \subseteq \{0,1\}^*$

  decision problems $\leftrightarrow$ subsets of all boolean strings

- NP := class of decision problems Π with following property:
  - There is a poly-time algorithm V_Π and a constant $c > 0$ such that
    - For any $x \in L_\Pi$ (i.e., YES instance) of size n, there is a *proof/witness* y of size n^c such that $V_\Pi(x, y) = 1$
    - For any $x' \notin L_\Pi$ (i.e., NO instance) there is *no such proof* z of size n^c such that $V_\Pi(x', z) = 1$.

NP

- Let Π be a decision problem and let L_Π be the set of all YES instances of Π. Then $L_\Pi \subseteq \{0,1\}^*$

  decision problems $\leftrightarrow$ subsets of all boolean strings
- NP := class of decision problems Π with following property:
  - There is a poly-time algorithm V_Π and a constant $c > 0$ such that
    - For any $x \in L_\Pi$ (i.e., YES instance) of size n, there is a *proof*/*witness* y of size n^c such that $V_\Pi(x, y) = 1$
    - For any $x' \notin L_\Pi$ (i.e., NO instance) there is *no such proof* z of size n^c such that $V_\Pi(x', z) = 1$.
- In other words, NP is the class of decision problems where the YES instances have a *small proof* that can be *verified* in poly-time

Problems in NP

- Clique
- Independent Set
- SAT (and 3SAT)
- TSP
- Hamilton cycle (and Hamilton path)
- Subset Sum
- Vertex Cover
- 3COLOR (and the graph coloring problem)
- *every* problem in P

coNP

- The class coNP is essentially the opposite of NP.
- for a decision problem Π, let $\overline{\Pi}$ be the *opposite* problem to Π, that is,

$$x \in L_{\Pi} \Leftrightarrow x \notin L_{\overline{\Pi}}$$

  equivalently, $L_{\overline{\Pi}} = \overline{L_{\Pi}}$.
- In simpler terms, every YES instance of Π is a NO instance of $\overline{\Pi}$ (and vice-versa)

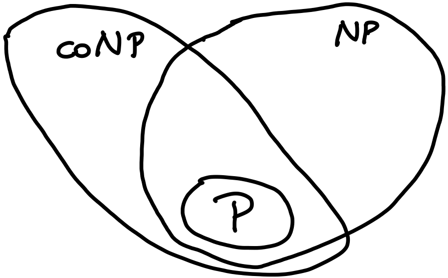
coNP

- The class coNP is essentially the opposite of NP.
- for a decision problem Π, let $\overline{\Pi}$ be the *opposite* problem to Π, that is,

$$x \in L_\Pi \Leftrightarrow x \notin L_{\overline{\Pi}}$$

  equivalently, $L_{\overline{\Pi}} = \overline{L_\Pi}$.
- In simpler terms, every YES instance of Π is a NO instance of $\overline{\Pi}$ (and vice-versa)
- coNP := class of decision problems Π such that $\overline{\Pi} \in$ NP.

Relation between P, NP and coNP



Unknown:

1) is $P = NP \cap coNP$?
2) is $NP = coNP$?
3) is $P = NP$?

- Complexity Classes & Complete Problems
  - NP
  - coNP
  - Completeness for NP
- Completing Karp Reductions/Polynomial Transformations
  - NP-completeness of 3SAT
  - Current Worldview
- Acknowledgements

A remark about reductions

- Given a particular reduction $\leq$ (Turing, Karp), we can define a *complete* problem for a complexity class $\mathcal{C}$ as follows:
  - **Hardness:** Π is $\mathcal{C}$-hard if for *every* problem $\Gamma \in \mathcal{C}$, we have

$$\Gamma \leq \Pi$$

  - **Membership in $\mathcal{C}$:** $\Pi \in \mathcal{C}$

A remark about reductions

- Given a particular reduction $\leq$ (Turing, Karp), we can define a *complete* problem for a complexity class $\mathcal{C}$ as follows:
  - **Hardness:** Π is $\mathcal{C}$-hard if for *every* problem $\Gamma \in \mathcal{C}$, we have

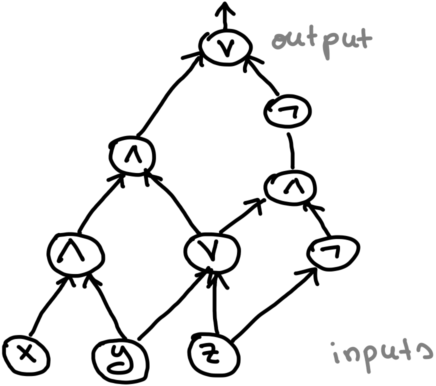
$$\Gamma \leq \Pi$$

  - **Membership in $\mathcal{C}$:** $\Pi \in \mathcal{C}$
- Complexity theorists prefer to define NP-completeness under *Karp reductions* (or polynomial transformations) because, as we will see, NP is closed under such reductions
  - Note that we *do not know* whether NP is closed under Turing reductions
  - The above would imply NP $=$ coNP, which is considered unlikely

$$\text{Under Turing reductions, UNSAT} \equiv \text{SAT}$$

CIRCUIT-SAT

- A *boolean circuit* is a DAG with:
  - input gates
  - AND/OR/NOT gates,
  - and a special gate (the output gate)



CIRCUIT-SAT

- A *boolean circuit* is a DAG with:
  - input gates
  - AND/OR/NOT gates,
  - and a special gate (the output gate)
- **CIRCUIT-SAT** problem:
  - **Input:** a boolean circuit Φ
  - **Output:** YES, if there is a truth assignment α such that $\Phi(\alpha) = 1$, NO otherwise.

Cook-Levin Theorem: CIRCUIT-SAT is NP-complete

Theorem (Cook-Levin)

CIRCUIT-SAT is NP-complete under polynomial transformations.

Cook-Levin Theorem: CIRCUIT-SAT is NP-complete

Theorem (Cook-Levin)

CIRCUIT-SAT is NP-complete under polynomial transformations.

- Want to prove that for any $\Pi \in \mathrm{NP}$, we have $\Pi \leq_m$ CIRCUIT-SAT
- **Proof sketch:** *computation is local*
  - $\Pi \in \mathrm{NP} \Rightarrow \exists$ poly-time verification algorithm V_Π and $c > 0$ such that for any instance $x \in \{0,1\}^n$,

$$x \in L_\Pi \Leftrightarrow \exists y \in \{0,1\}^{n^c} \text{ s.t. } V_\Pi(x,y) = 1$$

Cook-Levin Theorem: CIRCUIT-SAT is NP-complete

Theorem (Cook-Levin)

CIRCUIT-SAT is NP-complete under polynomial transformations.

- Want to prove that for any $\Pi \in \mathrm{NP}$, we have $\Pi \leq_m$ CIRCUIT-SAT
- **Proof sketch:** *computation is local*
  - $\Pi \in \mathrm{NP} \Rightarrow \exists$ poly-time verification algorithm V_Π and $c > 0$ such that for any instance $x \in \{0,1\}^n$,

$$x \in L_\Pi \Leftrightarrow \exists y \in \{0,1\}^{n^c} \text{ s.t. } V_\Pi(x,y) = 1$$

  - If $V_\Pi(x,y)$ runs in time $O(n^\gamma)$ (since it is polynomial in terms of the input size), there is circuit of size $O(n^\gamma)$ simulating computation of V_Π

    Can construct this circuit (from description of V_Π) in $\mathrm{poly}(n)$-time!

Cook-Levin Theorem: CIRCUIT-SAT is NP-complete

Theorem (Cook-Levin)

CIRCUIT-SAT is NP-complete under polynomial transformations.

- Want to prove that for any $\Pi \in \mathsf{NP}$, we have $\Pi \leq_m$ CIRCUIT-SAT
- **Proof sketch:** *computation is local*
  - $\Pi \in \mathsf{NP} \Rightarrow \exists$ poly-time verification algorithm V_Π and $c > 0$ such that for any instance $x \in \{0,1\}^n$,

$$x \in L_\Pi \Leftrightarrow \exists y \in \{0,1\}^{n^c} \text{ s.t. } V_\Pi(x,y) = 1$$

  - If $V_\Pi(x,y)$ runs in time $O(n^\gamma)$ (since it is polynomial in terms of the input size), there is circuit of size $O(n^\gamma)$ simulating computation of V_Π
  - So, we get a $\mathrm{poly}(n)$-sized circuit $\Phi_x(y)$ which is satisfiable iff $x \in L_\Pi$!

Cook-Levin Theorem: CIRCUIT-SAT is NP-complete

Theorem (Cook-Levin)

CIRCUIT-SAT is NP-complete under polynomial transformations.

- Want to prove that for any $\Pi \in \mathrm{NP}$, we have $\Pi \leq_m$ CIRCUIT-SAT
- **Proof sketch:** *computation is local*
  - $\Pi \in \mathrm{NP} \Rightarrow \exists$ poly-time verification algorithm V_Π and $c > 0$ such that for any instance $x \in \{0,1\}^n$,

$$x \in L_\Pi \Leftrightarrow \exists y \in \{0,1\}^{n^c} \text{ s.t. } V_\Pi(x,y) = 1$$

  - If $V_\Pi(x,y)$ runs in time $O(n^\gamma)$ (since it is polynomial in terms of the input size), there is circuit of size $O(n^\gamma)$ simulating computation of V_Π
  - So, we get a $\mathrm{poly}(n)$-sized circuit $\Phi_x(y)$ which is satisfiable iff $x \in L_\Pi$!
  - Thus, we have a transformation

$$x \mapsto \Phi_x$$

  such that $x \in L_\Pi \Leftrightarrow \Phi_x \in$ CIRCUIT-SAT.

- Complexity Classes & Complete Problems
    - NP
    - coNP
    - Completeness for NP
- Completing Karp Reductions/Polynomial Transformations
    - NP-completeness of 3SAT
    - Current Worldview
- Acknowledgements

3SAT is NP-complete

- To prove this, by Cook-Levin theorem, need to show that

$$\text{CIRCUIT-SAT} \le_m \text{3SAT}$$

3SAT is NP-complete

- To prove this, by Cook-Levin theorem, need to show that
$$\text{CIRCUIT-SAT} \leq_m \text{3SAT}$$
- By transitivity of polynomial transformations, enough to show
$$\text{CIRCUIT-SAT} \leq_m \text{SAT}$$

3SAT is NP-complete

- To prove this, by Cook-Levin theorem, need to show that

$$\text{CIRCUIT-SAT} \le_m \text{3SAT}$$

- By transitivity of polynomial transformations, enough to show

$$\text{CIRCUIT-SAT} \le_m \text{SAT}$$

- Let $\Phi \in$ CIRCUIT-SAT of size n (i.e., n gates and wires). We will construct CNF Ψ with $O(n)$ clauses such that

$$\Phi \text{ is satisfiable} \Leftrightarrow \Psi \text{ is satisfiable.}$$

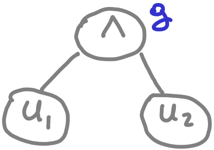
3SAT is NP-complete

- To prove this, by Cook-Levin theorem, need to show that
$$\text{CIRCUIT-SAT} \le_m \text{3SAT}$$
- By transitivity of polynomial transformations, enough to show
$$\text{CIRCUIT-SAT} \le_m \text{SAT}$$
- Let $\Phi \in$ CIRCUIT-SAT of size n (i.e., n gates and wires). We will construct CNF Ψ with $O(n)$ clauses such that
$$\Phi \text{ is satisfiable} \Leftrightarrow \Psi \text{ is satisfiable.}$$
- Can do the above simulating gate-by-gate (wire-by-wire):
  - each gate has a new variable, which will tell us the value of the gate
  - Simulate each gate operation (AND/OR/NOT) as a CNF
  - ensure that output gate variable should be true

Gate Simulations

- **AND:** CNF

$$(\overline{g} \vee u_1) \wedge (\overline{g} \vee u_2) \wedge (g \vee \overline{u_1} \vee \overline{u_2})$$



$\longmapsto$

$(\bar{g} \vee u_1) \wedge (\bar{g} \vee u_2)$

if u_1 or u_2 $= 0$ then must have $g = 0$ to satisfy above

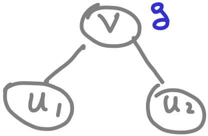
$(g \vee \bar{u}_1 \vee \bar{u}_2)$

if both $u_1 = u_2 = 1$ then $g = 1$

Gate Simulations

- **OR:** CNF

$$(g \vee \overline{u_1}) \wedge (g \vee \overline{u_2}) \wedge (\overline{g} \vee u_1 \vee u_2)$$



$(g \vee \bar{u}_1) \wedge (g \vee \bar{u}_2)$

if u_1 or $u_2 = 1$ then g must be 1 to satisfy above

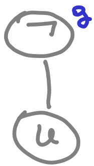
$(\bar{g} \vee u_1 \vee u_2)$

if $u_1 = u_2 = 0$ then g must be 0 to satisfy above clause.

Gate Simulations

- **NOT**: CNF

$$(\overline{g} \vee \overline{u}) \wedge (g \vee u)$$



$\bar{g} \vee \bar{u}$

if $u = 1$ then $g = 0$ to satisfy above clause

$g \vee u$

if $u = 0$ then $g = 1$ to satisfy above clause

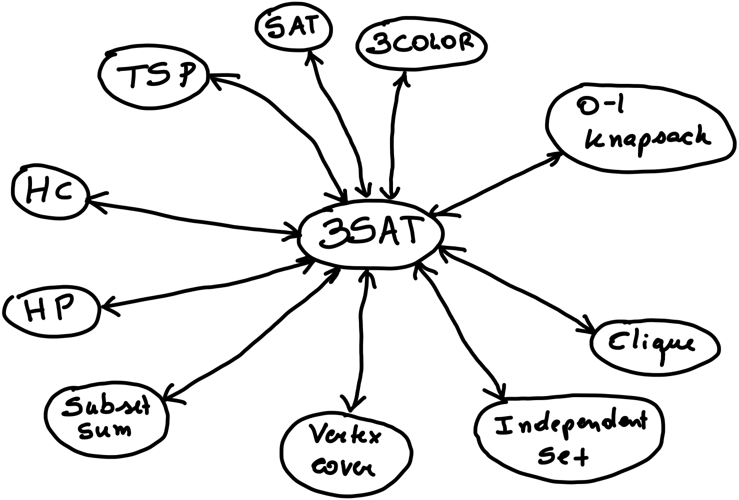
Gate Simulations

- **1**: CNF is simply literal g

Gate Simulations

- $\mathbf{0}$: CNF is simply literal $\overline{g}$

Updated Worldview



All NP-complete problems!

Where do we go next ?

- CS 360/365

Formalization of Algorithms , full proof of Cook-Levin & much more!

(Prof Blais teaching it nex term)

- Are there harder problems ?

For sure! See CS 360/365 or more advanced courses

Acknowledgement

Based on

- [Erickson 2019, Chapter 12]
- Prof. Lau's Lecture 18 notes

  https://cs.uwaterloo.ca/~lapchi/cs341/notes/L18.pdf

References I

Cormen, Thomas and Leiserson, Charles and Rivest, Ronald and Stein, Clifford (2009)
Introduction to Algorithms, third edition.
MIT Press

Dasgupta, Sanjay and Papadimitriou, Christos and Vazirani, Umesh (2006)
Algorithms

Erickson, Jeff (2019)
Algorithms
https://jeffe.cs.illinois.edu/teaching/algorithms/

Kleinberg, Jon and Tardos, Eva (2006)
Algorithm Design.
Addison Wesley