CS 341: Algorithms
Module 8: Intractability and Undecidability

Armin Jamshidpey, Eugene Zima

Based on lecture notes by many previous CS 341 instructors

David R. Cheriton School of Computer Science, University of Waterloo

Spring 2019

The Complexity Class NPC

The complexity class NPC denotes the set of all decision problems Π that satisfy the following two properties:

- $\Pi \in NP$
- For all $\Pi' \in NP, \Pi' \leq_P \Pi$.

NPC is an abbreviation for NP-complete.
Note that the definition does not imply that NP-complete problems exist!

The Complexity Class NPC (cont.)

Theorem 1

If $P \cap NPC \neq \emptyset$, then $P = NP$.

Proof.

We know that $P \subseteq NP$, so it suffices to show that $NP \subseteq P$.
Suppose
$\Pi \in P \cap NPC$ and let $\Pi' \in NP$. We will show that $\Pi' \in P$.

1. Since $\Pi \in NP$ and $\Pi \in NPC$, it follows that $\Pi' \leq_P \Pi$ (definition of NP-completeness).
2. Since $\Pi' \leq_P \Pi$ and $\Pi \in P$, it follows that $\Pi' \in P$

Satisfiability and the Cook-Levin Theorem

CNF-Satisfiability Problem

Instance: A boolean formula F in n boolean variables $x_1, \ldots, x_n$, such that F is the conjunction (logical "and") of m clauses, where each clause is the disjunction (logical "or") of literals. (A literal is a boolean variable or its negation.)
Question: Is there a truth assignment such that F evaluates to true?

Cook-Levin Theorem

CNF-Satisfiability $\in$ NPC.

Proving Problems NP-complete

Now, given any NP-complete problem, say Π_1, other problems in NP can be proven to be NP-complete via polynomial transformations from Π_1, as stated in the following theorem:

Theorem 2

Suppose that the following conditions are satisfied:

- $\Pi_1 \in NPC$,
- $\Pi_1 \leq_P \Pi_2$, and
- $\Pi_2 \in NP$.

Then $\Pi_2 \in NPC$.

More Satisfiability Problems

3-CNF-Satisfiability Problem

Instance: A boolean formula F in n boolean variables, such that F is the conjunction of m clauses, where each clause is the disjunction of exactly three literals.
Question: Is there a truth assignment such that F evaluates to true?

2-CNF-Satisfiability Problem

Instance: A boolean formula F in n boolean variables, such that F is the conjunction of m clauses, where each clause is the disjunction of exactly two literals.
Question: Is there a truth assignment such that F evaluates to true?

3-CNF-Satisfiability $\in$ NPC, while 2-CNF-Satisfiability $\in P$

CNF-Satisfiability $\leq_P$ 3-CNF-Satisfiability

Suppose that (X, C) is an instance of CNF-SAT, where $X = \{x_1, \ldots, x_n\}$ and $C = \{C_1, \ldots, C_m\}$. For each C_j , do the following:

- case 1: If $|C_j| = 1$, say $C_j = \{z\}$, construct four clauses $\{z, a, b\}, \{z, a, \bar{b}\}, \{z, \bar{a}, b\}, \{z, \bar{a}, \bar{b}\}$.
- case 2: If $|C_j| = 2$, say $C_j = \{z_1, z_2\}$, construct two clauses $\{z_1, z_2, c\}, \{z_1, z_2, \bar{c}\}$.
- case 3: If $|C_j| = 3$, then leave C_j unchanged.
- case 4: If $|C_j| \geq 4$, say $C_j = \{z_1, z_2, \ldots, z_k\}$, then construct $k - 2$ new clauses

$$\{z_1, z_2, d_1\}, \{\overline{d_1}, z_3, d_2\}, \{\overline{d_2}, z_4, d_3\}, \ldots, \\ \{\overline{d_{k-4}}, z_{k-2}, d_{k-3}\}, \{\overline{d_{k-3}}, z_{k-1}, z_k\}. \quad (1)$$

Correctness of the Transformation

Suppose I is a yes-instance of CNF-SAT. We show that $f(I)$ is a yes-instance of 3-CNF-SAT. Fix a truth assignment for X in which every clause contains a true literal. We consider each clause C_j of the instance I.

1. If $C = \{z\}$, then z must be true. The corresponding four clauses in $f(I)$ each contain z, so they are all satisfied.
2. If $C_j = \{z_1, z_2\}$, then at least one of the z_1 or z_2 is true. The corresponding two clauses in $f(I)$ each contain z_1, z_2 , so they are both satisfied.
3. If $C_j = \{z_1, z_2, z_3\}$, then C occurs unchanged in $f(I)$.
4. Suppose $C = \{z_1, z_2, z_3, \ldots, z_k\}$ where $k > 3$ and suppose $z_t \in C_j$ is a true literal. Define $d_i = \mathit{true}$ for $1 \leq i \leq t - 2$ and define $d_i = \mathit{false}$ for $t - 1 \leq i \leq k$. It is straightforward to verify that the $k - 2$ corresponding clauses in $f(I)$ each contain a true literal.

Correctness of the Transformation (cont.)

Conversely, suppose $f(I)$ is a yes-instance of 3-CNF-SAT. We show that I is a yes-instance of CNF-SAT.

1. Four clauses in $f(I)$ having the form $\{z, a, b\}, \{z, a, \bar{b}\}, \{z, \bar{a}, b\}\{z, \bar{a}, \bar{b}\}$ are all satisfied if and only if $z = \mathit{true}$. Then the corresponding clause $\{z\}$ in I is satisfied.
2. Two clauses in $f(I)$ having the form $\{z_1, z_2, c\}, \{z_1, z_2, \bar{c}\}$ are both satisfied if and only if at least one of $z_1, z_2 = \mathit{true}$. Then the corresponding clause $\{z_1, z_2\}$ in I is satisfied.
3. If $C_j = \{z_1, z_2, z_3\}$ is a clause in $f(I)$, then C_j occurs unchanged in I.

Correctness of the Transformation (cont.)

- Finally, consider the $k-2$ clauses in $f(I)$ arising from a clause $C_j = \{z_1, z_2, z_3, \ldots, z_k\}$ in I, where $k > 3$. We show that at least one of $z_1, z_2, \ldots, z_k = \mathit{true}$ if all $k-2$ of these clauses contain a true literal.
  Assume all of $z_1, z_2, \ldots, z_k = \mathit{false}$. In order for the first clause to contain a true literal, $d_1 = \mathit{true}$. Then, in order for the second clause to contain a true literal, $d_2 = \mathit{true}$. This pattern continues, and in order for the second last clause to contain a true literal, $d_{k-3} = \mathit{true}$. But then the last clause contains no true literal, which is a contradiction. We have shown that at least one of $z_1, z_2, \ldots, z_k = \mathit{true}$, which says that the clause $\{z_1, z_2, z_3, \ldots, z_k\}$ contains a true literal, as required.

3-CNF-Satisfiability $\leq_P$ Clique

Let I be the instance of 3-CNF-SAT consisting of n variables, $x_1, \ldots, x_n$, and m clauses, $C_1, \ldots, C_m$. Let $C_i = \{z_1^i, z_2^i, z_3^i\}$, $1 \leq i \leq m$. Define $f(I) = (G, k)$, where $G = (V, E)$ according to the following rules:

- $V = \{v_j^i : 1 \leq i \leq m, 1 \leq j \leq 3\}$,
- $v_j^i v_{j'}^{i'} \in E$ if and only if $i \neq i'$ and $z_j^i \neq \overline{z_{j'}^{i'}}$, and
- $k = m$.

Non-edges of the constructed graph correspond to

1. inconsistent truth assignments of literals from two different clauses; or
2. any two literals in the same clause.

Example

$$\mathrm{I}: \begin{cases} C_1 = \{x_1, \bar{x_2}, \bar{x_3}\} \\ C_2 = \{\bar{x_1}, x_2, x_3\} \\ C_3 = \{x_1, x_2, x_3\} \end{cases}$$

$x_1 = \mathit{true}, x_2 = \mathit{true}, x_3 = \mathit{false}$

$f(I)$: