IAS/PCMI Summer Session 2000
Clay Mathematics Undergraduate Program
Basic Course on Computational Complexity

# Lecture 7: NP-Complete Problems

David Mix Barrington and Alexis Maciel
July 25, 2000

## 1. Circuit Satisfiability

The *circuit satisfiability* problem (CIRCUIT-SAT) is the circuit analogue of SAT. Given a Boolean circuit $C$, is there an assignment to the variables that causes the circuit to output 1?

**Theorem 1** *CIRCUIT-SAT is* NP*-complete.*

**Proof** It is clear that CIRCUIT-SAT is in NP since a nondeterministic machine can guess an assignment and then evaluate the circuit in polynomial time.

Now suppose that $A$ is a language in NP. Recall from Lecture 3 that $A$ has a polynomial-time verifier, an algorithm $V$ with the property that $x \in A$ if and only if $V$ accepts $\langle x, y\rangle$ for some $y$. In addition, from Lecture 5, we know that there is a polynomial-size circuit $C$ equivalent to $V$. The input of $C$ is the entire input of $V$, i.e., both $x$ and $y$. And $C$ can be constructed in polynomial time given the length of $x$ and $y$.

The reduction from $A$ to CIRCUIT-SAT operates as follows: given an input $x$, output a description of the circuit $C(x, y)$ with the $x$ variables set to the given values and the $y$ variables left as variables. The resulting circuit is satisfiable if and only $x \in A$. And the reduction can be computed in polynomial time because of the uniformity of $C$. □

**Theorem 2** *SAT is* NP*-complete.*

**Proof** It is clear that SAT is in NP: guess an assignment an evaluate the formula as if it was a circuit. To show that SAT is NP-complete we reduce CIRCUIT-SAT to SAT.

Let $C$ be a circuit. We construct a formula whose variables are the variables of the circuit plus one new variable for each gate of the circuit. These new variables are interpreted as the value of the corresponding gates. The formula will express the fact that the circuit works properly. For example, if a NOT gate $h$ has gate $g$ input, that NOT gate works correctly if $(g \wedge \overline{h}) \vee (\overline{g} \wedge h)$. If an AND gate $h$ has gates $g_1$ and $g_2$ as inputs, that AND gate works correctly if

$$(g_1 \wedge g_2 \wedge h) \vee (\overline{g_1} \wedge g_2 \wedge \overline{h}) \vee (g_1 \wedge \overline{g_2} \wedge \overline{h}) \vee (\overline{g_1} \wedge \overline{g_2} \wedge \overline{h}).$$

The formula corresponding to the entire circuit is the conjunction of all these formulas, for every gate, and the formula $g$ where $g$ is the output gate. It should be clear that $C$ is satisfiable if and only if this formula is satisfiable. It is also easy to see that the formula can be constructed in polynomial time. □

The NP-completeness of 3SAT can be established by a simple modification of the previous proof.

**Corollary 3** *3SAT is* NP-*complete.*

The NP-completeness of CLIQUE then follows from an earlier reduction.

**Corollary 4** *CLIQUE is* NP-*complete.*

## 2. Vertex Cover

A *vertex cover* of an undirected graph is a subset of the nodes such that every edge in the graph is adjacent to one of these nodes. The set of all nodes of a graph always constitutes a vertex cover and some graphs have vertex covers of size 1. The VERTEX-COVER problem is to determine, given a graph $G$ and a number $k$, whether $G$ has a vertex cover of size $k$.

**Theorem 5** *VERTEX-COVER is* NP-*complete.*

**Proof** VERTEX-COVER is clearly in NP. We now reduce 3SAT to VERTEX-COVER.

Consider a 3CNF formula with $n$ variables and $c$ clauses. For each variable $x$, we will have two nodes labeled $x$ and $\overline{x}$ and connected by an edge. Each such group of two nodes plus an edge is called a variable "gadget". For each clause $(x \vee y \vee z)$, we will have a clause gadget that consists of three nodes labeled $x$, $y$ and $z$ and connected to each other by three edges. The graph corresponding to the 3CNF formula consists of all these gadgets plus an edge between any two nodes that have the same label.

It is clear that this graph can be constructed in polynomial time. We claim that the formula is satisfiable if and only if the graph has a vertex cover of size $n + 2c$. (Details can be found in Sipser's textbook.) $\square$

## 3. Subset Sum

Given a list of numbers and a number $k$, is there a subset of the numbers that adds to exactly $k$? For example, the answer is yes for $\langle(3, 4, 12, 7, 4), 20\rangle$ and no for $\langle(3, 4, 12, 7, 4), 6\rangle$. This is called the SUBSET-SUM problem. In a exercise, we already showed that the problem is in P if all the numbers are given in unary. Here we show that the problem is NP-complete if the numbers are given in the usual binary notation.

**Theorem 6** *SUBSET-SUM is* NP*-complete.*

**Proof** Once again, it clear that the problem is in NP. We reduce 3SAT to SUBSET-SUM. Consider a 3CNF formula with variables $x_1, \ldots, x_n$ and clauses $c_1, \ldots, c_r$. For each variable $x_i$, we will have two numbers $y_i$ and $z_i$ in the list. For each clause $c_j$, we will also have two numbers $s_j$ and $t_j$. We define all of these numbers by specifying their base 10 representations.

The construction is best explained by an example and a picture. If the formula is

$$(x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee \overline{x_3}),$$

then the base 10 representations of the numbers will look like this:

|  | $x_1$ | $x_2$ | $x_3$ | $c_1$ | $c_2$ |
|---|---|---|---|---|---|
| $y_1$ | 1 | 0 | 0 | 1 | 0 |
| $z_1$ | 1 | 0 | 0 | 0 | 1 |
| $y_2$ | 0 | 1 | 0 | 1 | 1 |
| $z_2$ | 0 | 1 | 0 | 0 | 0 |
| $y_3$ | 0 | 0 | 1 | 0 | 0 |
| $z_3$ | 0 | 0 | 1 | 1 | 1 |
| $s_1$ | 0 | 0 | 0 | 1 | 0 |
| $t_1$ | 0 | 0 | 0 | 1 | 0 |
| $s_2$ | 0 | 0 | 0 | 0 | 1 |
| $t_2$ | 0 | 0 | 0 | 0 | 1 |
| $k$ | 1 | 1 | 1 | 3 | 3 |

The number $y_i$ corresponds to the positive occurrences of $x_i$ in the formula while the number $z_i$ corresponds to its negative occurrences.

It should be clear how to generalize this construction to an arbitrary 3CNF formula. And the list of numbers can clearly be constructed in polynomial time. We claim that a subset of these numbers adds to exactly $k$ if and only if the formula is satisfiable. A key point is that the sum of the numbers can be done column by column, independently, because carries will never occur. (Details can be found in Sipser's textbook.) $\square$

## 4. Exercises

1. Give a direct reduction of SAT to CIRCUIT-SAT.

2. Give direct reductions of CLIQUE to VERTEX-COVER, and vice-versa. (Hint: There is a reduction that works in both directions.)

3. Consider the following BIN-PACKING problem: given a collection of objects each with a positive integer size and a collection bins each with a positive integer capacity, is there a way to put all the objects into the bins so that the capacity of no bin is exceeded? Show that SUBSET-SUM reduces to BIN-PACKING.

4. Show that 3-colorability is NP-complete. (Hint: See the hint for Exercise 7.34 in Sipser's textbook.)

5. Show that if P = NP, then there is a polynomial-time algorithm that given a Boolean formula determines if the formula is satisfiable and finds a satisfying assignment, if one exists.

6. 2SAT is defined exactly like 3SAT but for clauses with two literals. Show that 2SAT is in P.