Complexity Theory
VU 181.142, SS 2017

5. NP-Completeness

Reinhard Pichler
Institut für Informationssysteme
Arbeitsbereich DBAI
Technische Universität Wien

25 April, 2017

Outline

5. NP-Completeness
5.1 Some Variants of Satisfiability
5.2 CIRCUIT SAT
5.3 NOT-ALL-EQUAL-SAT
5.4 1-IN-3-SAT
5.5 Some Graph Problems
5.6 3-COLORABILITY
5.7 HAMILTON-PATH, etc.
5.8 Summary

Narrowing NP-complete languages

An NP-complete language can sometimes be narrowed down by transformations which eliminate certain features of the language but still preserve NP-completeness.

Restricting SAT to formulae in CNF and a further restriction to 3-SAT are typical examples. Generally, k-SAT (i.e., formulae are restricted to CNF with exactly k literals in each clause) is NP-complete for any $k \geq 3$.

Here is another example of narrowing an NP-complete language:

Proposition

3-SAT remains NP-complete even if the Boolean expressions $\varphi$ in 3-CNF are restricted such that

- each variable appears at most three times in $\varphi$ and
- each literal appears at most twice in $\varphi$. 

We have already encountered several versions of satisfiability problems:

- intractable: SAT, 3-SAT
- tractable: 2-SAT, HORNSAT

We shall encounter further intractable versions of satisfiability problems:

- restricted (but still intractable) versions of SAT
- CIRCUIT SAT
- Not-all-equal SAT (NAESAT)
- (MONOTONE) 1-IN-3-SAT
- strongly related problem: HITTING SET
Boolean Circuits

Semantics
Let $C$ be a Boolean circuit and let $X(C)$ denote the set of variables appearing in the circuit $C$. A truth assignment for $C$ is a function $T : X(C) \rightarrow \{\text{true, false}\}$.

The truth value $T(i)$ for each gate $i$ is defined inductively:

- If $s(i) = \text{true}$, $T(i) = \text{true}$ and if $s(i) = \text{false}$, $T(i) = \text{false}$.
- If $s(i) = x_j \in X(C)$, then $T(i) = T(x_j)$.
- If $s(i) = \neg$, then $T(i) = \text{true}$ if $T(j) = \text{false}$, else $T(i) = \text{false}$ where $(j, i)$ is the unique edge entering $i$.
- If $s(i) = \wedge$, then $T(i) = \text{true}$ if $T(j) = T(j') = \text{true}$ else $T(i) = \text{false}$ where $(j, i)$ and $(j', i)$ are the two edges entering $i$.
- If $s(i) = \vee$, then $T(i) = \text{true}$ if $T(j) = \text{true}$ or $T(j') = \text{true}$ else $T(i) = \text{false}$ where $(j, i)$ and $(j', i)$ are the two edges entering $i$.
- $T(C) = T(n)$, i.e. the value of the circuit $C$.

CIRCUIT SAT

INSTANCE: Boolean circuit $C$ with variables $X(C)$

QUESTION: Does there exist a truth assignment $T : X(C) \rightarrow \{\text{true, false}\}$ such that $T(C) = \text{true}$?

Theorem
CIRCUIT SAT is NP-complete.

Proof of NP-Membership
Consider the following NP-algorithm:

1. Guess a truth assignment $T : X(C) \rightarrow \{\text{true, false}\}$.
2. Check that $T(C) = \text{true}$ holds.
Reduction from SAT to 3-SAT

Motivation

- We have already seen how an arbitrary propositional formula $\varphi$ can be transformed efficiently into a sat-equivalent formula $\psi$ in 3-CNF.
- This transformation (first into CNF and then into 3-CNF) is intuitive and clearly works in polynomial time. However, the log-space complexity of this transformation is not immediate.
- We now give an alternative transformation by reducing CIRCUIT SAT to 3-SAT. In total, we thus have:

\[ \text{SAT} \leq_L \text{CIRCUIT SAT} \leq_L 3\text{-SAT} \]

NAESAT

Not-all-equal SAT (NAESAT)

INSTANCE: Boolean formula $\varphi$ in 3-CNF

QUESTION: Does there exist a truth assignment $T$ appropriate to $\varphi$, such that the 3 literals in each clause do not have the same truth value?

Remark. Clearly NAESAT $\subset$ 3-SAT.

Theorem

NAESAT is NP-complete.
1-IN-3-SAT

1-IN-3-SAT

INSTANCE: Boolean formula $\varphi$ in 3-CNF

QUESTION: Does there exist a truth assignment $T$ appropriate to $\varphi$, such that in each clause, exactly one literal is true in $T$?

MONOTONE 1-IN-3-SAT

INSTANCE: Boolean formula $\varphi$ in 3-CNF, s.t. the clauses in $\varphi$ contain only unnegated atoms.

QUESTION: Does there exist a truth assignment $T$ appropriate to $\varphi$, such that in each clause, exactly one literal is true in $T$?

Theorem

Both 1-IN-3-SAT and MONOTONE 1-IN-3-SAT are NP-complete.

NAESAT

Proof of NP-Hardness

Recall the Boolean formula $\varphi(C)$ resulting from the reduction of CIRCUIT SAT to 3-SAT. For all one- and two-literal clauses in the resulting CNF-formula $\varphi(C)$, we add the same literal $z$ (possibly twice) to make them 3-literal clauses.

The resulting formula $\varphi_2(C)$ fulfills the following equivalence:

$$\varphi_2(C) \in \text{NAESAT} \iff C \in \text{CIRCUIT SAT}.$$  

"⇒" If a truth assignment $T$ satisfies $\varphi_2(C)$ in the sense of NAESAT, so does the complementary truth assignment $\overline{T}$.

Thus, $z$ is false in either $T$ or $\overline{T}$ which implies that $\varphi(C)$ is satisfied by either $T$ or $\overline{T}$. Thus $C$ is satisfiable.

"⇐" If a truth assignment $T$ satisfies $\varphi_2(C)$ in the sense of NAESAT, so does the complementary truth assignment $\overline{T}$.

Thus, $z$ is false in either $T$ or $\overline{T}$ which implies that $\varphi(C)$ is satisfied by either $T$ or $\overline{T}$. Thus $C$ is satisfiable.

Remarks

Clearly 1-IN-3-SAT ⊂ NAESAT ⊂ 3-SAT. The instances of these 3 problems are the same, namely 3-CNF formulae. However, the positive instances of 1-IN-3-SAT are a proper subset of NAESAT, which in turn are a proper subset of the positive instances of 3-SAT.

Note that the NP-completeness of any of these 3 problems does not immediately imply the NP-completeness of any of the other problems, since it is a priori not clear if further constraining the positive instances makes things easier or harder.

MONOTONE 1-IN-3-SAT is a special case of 1-IN-3-SAT, i.e., the instances of the former are a proper subset of the latter while the question remains the same. The NP-hardness of the special case immediately implies the NP-hardness of the general case.
Proof of the NP-hardness of **1-IN-3-SAT**

We prove the NP-hardness by a reduction from **4-SAT**:
Let \( \varphi \) be an arbitrary instance of **4-SAT**, i.e., \( \varphi \) is in 4-CNF.
We construct an instance \( \psi \) of **1-IN-3-SAT** as follows:
For every clause \( l_1 \lor l_2 \lor l_3 \) in \( \varphi \), let \( a_1, a_2, a_3, b_1, b_2, c_1, c_2, d \) be 9 fresh propositional variables. Then \( \psi \) contains the following 7 clauses:

(1) \( l_1 \lor a_1 \lor b_1 \)  
(2) \( l_2 \lor a_2 \lor b_1 \)  
(3) \( l_3 \lor a_3 \lor b_2 \)  
(4) \( l_1 \lor a_3 \lor b_2 \)  
(5) \( l_2 \lor a_2 \lor c_1 \)  
(6) \( l_1 \lor a_2 \lor c_2 \)  
(7) \( b_1 \lor b_2 \lor d \)

**Idea.** These seven clauses guarantee that in a legal 1-in-3 assignment of \( \psi \), the clause \( l_1 \lor \ldots \lor l_4 \) must be **true**.
By (1) – (3): If \( l_1 \) and \( l_2 \) are **false**, then \( b_1 \) must be **true**.
By (4) – (6): If \( l_3 \) and \( l_4 \) are **false**, then \( b_2 \) must be **true**.
However, by (7), it is not allowed that both \( b_1 \) and \( b_2 \) are **true**.

**HITTING SET**

**INSTANCE:** Set \( T = \{ t_1, \ldots, t_p \} \), family \( (V_i)_{1 \leq i \leq n} \) of subsets of \( T \), i.e.: for all \( i \in \{1, \ldots, n\} \), \( V_i \subseteq T \).

**QUESTION:** Does there exist a set \( W \subseteq T \), s.t. \( |W \cap V_i| = 1 \) for all \( i \in \{1, \ldots, n\} \)? (A set \( W \) with this property is called a "hitting set").

**Corollary**

**HITTING SET** is **NP-complete**.

**Proof of the NP-hardness**

By reduction from **MONOTONE 1-IN-3-SAT**: Let an instance of **MONOTONE 1-IN-3-SAT** be given by the 3-CNF formula \( \varphi \) over the variables \( X \). We define the following instance of **HITTING SET**:
\( T = X \). Moreover, suppose that \( \varphi \) contains \( n \) clauses. Then there are \( n \) sets \( (V_i)_{1 \leq i \leq n} \). If the \( i \)-th clause in \( \varphi \) is \( l_1 \lor l_2 \lor l_3 \), then \( V_i = \{ l_1, l_2, l_3 \} \).

We show how an arbitrary instance \( \varphi \) of **1-IN-3-SAT** can be transformed into an equivalent instance \( \psi \) of **MONOTONE 1-IN-3-SAT**:

Let \( X = \{ x_1, \ldots, x_n \} \) be the variables in \( \varphi \). Then the variables in \( \psi \) are \( X \cup \{ x'_i \mid 1 \leq i \leq n \} \cup \{ a, b, c \} \). In \( \varphi \), we replace every negative literal of the form \( \neg x_i \) (for some \( i \)) by the unnegated atom \( x'_i \).
Moreover, for every \( i \in \{1, \ldots, n\} \), we add the following 3 clauses:

(1) \( x_i \lor x'_i \lor a \)  
(2) \( x_i \lor x'_i \lor b \)  
(3) \( a \lor b \lor c \)

**Idea.** These three clauses guarantee that in a legal 1-in-3 assignment of \( \psi \), the variables \( x_i \) and \( x'_i \) have complementary truth values. Hence, \( x'_i \) indeed encodes \( \neg x_i \).

**Some Graph Problems**

We have already proved the NP-completeness of the following graph problems:
- **INDEPENDENT SET**
- **CLIQUE**
- **VERTEX COVER**

We shall now show the following results:
- **3-COLORABILITY** is NP-complete.
- **HAMILTON-PATH \( \leq_L \) HAMILTON-CYCLE \( \leq_L \) TSP(D)**
INDEPENDENT SET

INSTANCE: Undirected graph \( G = (V, E) \) and integer \( K \).

QUESTION: Does there exist an independent set \( I \) of size \( \geq K \)?
i.e., \( I \subseteq V \), s.t. for all \( i, j \in I \) with \( i \neq j \), \([i, j] \notin E\).

NP-Hardness Proof of 3-COLORABILITY

By reduction from \textsc{NAESAT}: Let an arbitrary instance of \textsc{NAESAT} be given by a Boolean formula \( \varphi = c_1 \land \ldots \land c_m \) in 3-CNF with variables \( x_1, \ldots, x_n \). We construct the following graph \( G(\varphi) \):

Let \( V = \{a\} \cup \{x_i, \neg x_i \mid 1 \leq i \leq n\} \cup \{l_1, l_2, l_3 \mid 1 \leq i \leq m\} \),
i.e. \( |V| = 1 + 2n + 3m \).

For each variable \( x_i \) in \( \varphi \), we introduce a triangle \([a, x_i, \neg x_i]\),
i.e. all these triangles share the node \( a \).

For each clause \( c_i \) in \( \varphi \), we introduce a triangle \([l_{i1}, l_{i2}, l_{i3}]\). Moreover, each of these vertices \( l_{ij} \) is further connected to the node corresponding to this literal, i.e.: if the \( j \)-th literal in \( c_i \) is of the form \( x_a \) (resp. \( \neg x_a \)) then we introduce an edge between \( l_{ij} \) and \( x_a \) (resp. \( \neg x_a \)).

NP-Completeness of \( \textsc{3-COLORABILITY} \)

The \( k \)-\textsc{COLORABILITY}-problem is \( \text{NP}\)-complete for any fixed \( k \geq 3 \).
The \( 2 \)-\textsc{COLORABILITY}-problem is in \( \text{P} \).

Theorem

\( \text{NP-Membership of } k\text{-COLORABILITY} \):
1. Guess an assignment \( f : V \rightarrow \{1, \ldots, k\} \)
2. Check for every edge \([i, j] \in E\) that \( f(i) \neq f(j) \).

\( \text{P-Membership of } 2\text{-COLORABILITY} \): (w.l.o.g., \( G \) is connected)
1. Start by assigning an arbitrary color to an arbitrary vertex \( v \in V \).
2. Suppose that the vertices in \( S \subseteq V \) have already been assigned a color. Choose \( x \in S \) and assign to all vertices adjacent to \( x \) the opposite color. \( G \) is 2-colorable iff step 2 never leads to a contradiction.
Example

The 3-CNF formula \( \varphi = (x_1 \lor \neg x_2 \lor x_3) \land (x_2 \lor x_3 \lor \neg x_4) \) is reduced to the following graph:

\[
\begin{array}{c}
\text{x}_1 \\
\text{x}_2 \\
\text{x}_3 \\
\text{x}_4
\end{array}
\begin{array}{c}
\neg\text{x}_1 \\
\neg\text{x}_2 \\
\neg\text{x}_3 \\
\neg\text{x}_4
\end{array}
\begin{array}{c}
h_1 \\
h_2 \\
h_3
\end{array}
\begin{array}{c}
h_1 \\
h_2 \\
h_3
\end{array}
\begin{array}{c}
l_1 \\
l_2 \\
l_3
\end{array}
\begin{array}{c}
l_1 \\
l_2 \\
l_3
\end{array}
\begin{array}{c}
l_1 \\
l_2 \\
l_3
\end{array}
\begin{array}{c}
l_1 \\
l_2 \\
l_3
\end{array}
\]

Correctness of the Problem Reduction

Proof (continued)

\[ \text{"\Rightarrow" Suppose that } G \text{ has a 3-coloring with colors } \{0, 1, 2\}. \text{ W.l.o.g., the node } a \text{ has the color 2. This induces a truth assignment } T \text{ via the colors of the nodes } x_i; \text{ if the color is 1, then } T(x_i) = \text{true}, \text{ else } T(x_i) = \text{false}. \text{ We claim that } T \text{ is a legal NAESAT-assignment. Indeed, if in some clause, all literals had the value false (resp. true), then we could not use the color 0 (resp. 1) for coloring the triangle } [l_1, l_2, l_3], \text{ a contradiction.}

\[ \text{"\Leftrightarrow" Suppose that there exists an NAESAT-assignment } T \text{ of } \varphi. \text{ Then we can extract a 3-coloring for } G \text{ from } T \text{ as follows:}
\]

(i) Node \( a \) is colored with color 2.
(ii) If \( T(x_i) = \text{true} \), then color \( x_i \) with 1 and \( \neg x_i \) with 0 else vice versa.
(iii) From each \([l_1, l_2, l_3]\), color two literals having opposite truth values with 0 (true) and 1 (false). Color the third with 2.

HAMILTON-PATH

INSTANCE: (directed or undirected) graph \( G = (V, E) \)

QUESTION: Does \( G \) have a Hamilton path?

i.e., a path visiting all vertices of \( G \) exactly once.

HAMILTON-CYCLE

INSTANCE: (directed or undirected) graph \( G = (V, E) \)

QUESTION: Does \( G \) have a Hamilton cycle?

i.e., a cycle visiting all vertices of \( G \) exactly once.

TSP(D)

INSTANCE: \( n \) cities \( 1, \ldots, n \) and a nonnegative integer distance \( d_{ij} \) between any two cities \( i \) and \( j \) (such that \( d_{ij} = d_{ji} \)), and an integer \( B \).

QUESTION: Is there a tour through all cities of length at most \( B \)?

i.e., a permutation \( \pi \) s.t. \( \sum_{i=1}^{n} d_{\pi(i)\pi(i+1)} \leq B \) with \( \pi(n+1) = \pi(1) \).
Complexity Theory

5. NP-Completeness

5.7. HAMILTON-PATH, etc.

Complexity

**Theorem**

HAMILTON-PATH, HAMILTON-CYCLE, and TSP(D) are NP-complete.

**Proof**

We shall show the following chain of reductions:

HAMILTON-PATH ≤_L HAMILTON-CYCLE ≤_L TSP(D)

It suffices to show NP-membership for the hardest problem:

1. Guess a tour π through the n cities.
2. Check that ∑_{i=1}^{n} d(π(i), π(i+1)) ≤ B with π(n+1) = π(1).

Likewise, it suffices to prove the NP-hardness of the easiest problem.

The NP-hardness of HAMILTON-PATH (by a reduction from 3-SAT) is quite involved and is therefore omitted here (see Papadimitriou’s book).

HAMILTON-CYCLE vs. TSP(D)

HAMILTON-CYCLE ≤_L TSP(D)

Let an arbitrary instance of HAMILTON-CYCLE be given by the graph G = (V, E). We construct an equivalent instance of TSP(D) as follows:

Let V := V ∪ {z} for some new vertex z and E' := E ∪ {[v, z] | v ∈ V}. G has a Hamilton path ⇔ G' has a Hamilton cycle

“⇒” Suppose that G has a Hamilton path π starting at vertex a and ending at b. Then π ∪ {z} is clearly a Hamilton cycle in G'.

“⇐” Let C be a Hamilton cycle in G'. In particular, C goes through z. Let a and b be the two neighboring nodes of z in this cycle. Then C \ {z} is a Hamilton path (starting at vertex a and ending at b) in G.

Summary of Reductions

- SAT
  - 4-SAT
  - 1-in-3-SAT
  - CIRCUIT-SAT
- 3-SAT
- IND-SET
- HAM-P.
- NAESAT
- MON
  - 1-in-3-SAT
  - VC
  - CLQ
  - HAM-C.
  - 3-COL
- HITTING SET
- TSP(D)
Learning Objectives

- The concept of NP-completeness and its characterizations in terms of succinct certificates.
- You should now be familiar with the intuition of NP-completeness (and recognize NP-complete problems).
- Basic techniques to prove problems NP-complete.
- A basic repertoire of NP-complete problems (in particular, versions of SAT and some graph problems) to be used in further NP-completeness proofs.
- Reductions, reductions, reductions, . . .