Lesson 13: NP-complete problems

Theme: NP-complete problems as the boundary of the class of computationally feasible problems.

1 Polynomial time reduction

Recall the definition of reduction in Lesson 11: \( L_1 \leq_m L_2 \), if there is a computable function \( F \) such that for every \( w \in \Sigma^* \):

\[
w \in L_1 \text{ if and only if } F(w) \in L_2
\]

We say that a TM \( M \) computes \( F \) in time \( O(g(n)) \), if there is a constant \( c > 0 \) such that on every word \( w \), \( M \) accepts \( w \) with the accepting run:

\[
q_0 \ x \vdash \cdots \vdash q_{\text{acc}} \ F(w)
\]

and the length of the run is \( \leq c \cdot g(|w|) \). Such a function \( F \) is called polynomial time computable function, if \( g(n) = n^k \) for some \( k > 0 \).

Definition 13.1 A language \( L_1 \) is polynomial time reducible to another language \( L_2 \), denoted by \( L_1 \leq_P L_2 \), if there is a polynomial time computable function \( F \) such that for every \( w \in \Sigma^* \):

\[
w \in L_1 \text{ if and only if } F(w) \in L_2
\]

Such a function \( F \) is called polynomial time reduction, also known as Karp reduction.

2 The class of NP-complete problems

Definition 13.2 Let \( L \) be a language.

- \( L \) is NP-hard, if for every \( L' \in \text{NP} \), \( L' \leq_p L \).
- \( L \) is NP-complete, if \( L \in \text{NP} \) and \( L \) is NP-hard.

Recall that a propositional formula (Boolean expression) with variables \( x_1, \ldots, x_n \) is in Conjunctive Normal Form (CNF), if it is of the form: \( \bigwedge_i \bigvee_j \ell_{i,j} \) where each \( \ell_{i,j} \) is a literal, i.e., a variable \( x_k \) or its negation \( \neg x_k \). It is in 3-CNF, if it is of the form \( \bigwedge_i (\ell_{i,1} \lor \ell_{i,2} \lor \ell_{i,3}) \).

A formula \( \varphi \) is satisfiable, if there is an assignment of Boolean values True or False to each variables in \( \varphi \) that evaluates to True.

<table>
<thead>
<tr>
<th>SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong></td>
</tr>
<tr>
<td><strong>Task:</strong></td>
</tr>
</tbody>
</table>

Theorem 13.3 SAT is NP-complete.

<table>
<thead>
<tr>
<th>3-SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong></td>
</tr>
<tr>
<td><strong>Task:</strong></td>
</tr>
</tbody>
</table>

Theorem 13.4 3-SAT is NP-complete.
3 More NP-complete problems

We need a few terminologies. Let $G = (V, E)$ be a (undirected) graph.

- $G$ is 3-colorable, if we can color the vertices in $G$ with 3 colors (every vertex must be colored with one color) such that no two adjacent vertices have the same color.
- A set $C \subseteq V$ is a clique in $G$, if every pair of vertices in $C$ are adjacent.
- A set $W \subseteq V$ is a vertex cover, if every edge in $E$ is adjacent to at least one vertex in $W$.
- A set $I \subseteq V$ is independent, if every pair of vertices in $I$ are non-adjacent.
- A set $D \subseteq V$ is dominating, if every vertex in $V$ is adjacent to at least one vertex in $D$.

All the following problems are NP-complete.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Input: A (undirected) graph $G = (V, E)$.</th>
<th>Task: Output True, if $G$ is 3-colorable. Otherwise, output False.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clique</td>
<td>Input: A (undirected) graph $G = (V, E)$ and an integer $k \geq 0$ in binary form.</td>
<td>Task: Output True, if $G$ has a clique of size $\geq k$. Otherwise, output False.</td>
</tr>
<tr>
<td>Independent-Set</td>
<td>Input: A (undirected) graph $G = (V, E)$ and an integer $k \geq 0$ in binary form.</td>
<td>Task: Output True, if $G$ has an independent set of size $\geq k$. Otherwise, output False.</td>
</tr>
<tr>
<td>Vertex-Cover</td>
<td>Input: A (undirected) graph $G = (V, E)$ and an integer $k \geq 0$ in binary form.</td>
<td>Task: Output True, if $G$ has a vertex cover of size $\leq k$. Otherwise, output False.</td>
</tr>
<tr>
<td>Dominating-Set</td>
<td>Input: A (undirected) graph $G = (V, E)$ and an integer $k \geq 0$ in binary form.</td>
<td>Task: Output True, if $G$ has an dominating set of size $\leq k$. Otherwise, output False.</td>
</tr>
</tbody>
</table>