1 Knapsack Problem

Definition 1 (Knapsack Problem) Given a set \( W = \{w_1, \ldots, w_n\} \) of positive integer numbers (weights of objects) and a positive number \( C \) (knapsack capacity) determine if there exists a subset \( S \) of \( W \) with sum of its elements equal to \( C \):

\[
\sum_{w \in S} w = C.
\]

Definition 2 (Knapsack Language) The Knapsack Language is the set of pairs \( (W, C) \), for which there exists a solution to the Knapsack Problem.

Remark: Can you encode a set of binary strings in one binary string?

Problem 1 Design an algorithm solving the Knapsack Problem in time polynomial in \( C \cdot |W| \).

Hint: Use dynamic programming: For each \( 1 \leq k \leq n \) consider the following set:

\[
C_k = \left\{ \sum_{w \in S} w : S \subseteq \{w_1, \ldots, w_k\} \right\}.
\]

How can we construct \( C_{k+1} \) given \( C_k \)?

Definition 3 We say that a set of binary strings \( A \subseteq \{0, 1\}^* \) is Karp-reducible to a set \( B \subseteq \{0, 1\}^* \) (and denote this by \( A \leq_K B \)) if there exists a polynomial time algorithm \( M : \{0, 1\}^* \rightarrow \{0, 1\}^* \) such that for all \( x \),

\[
x \in A \text{ if and only if } M(x) \in B.
\]

Problem 2 Compare Karp-reduction with \( m \)-reduction. What is the main difference? Can you give two sets between which you have an \( m \) reduction, but don’t expect a Karp-reduction to exist? Why?

Problem 3 Prove that the Knapsack Language is in \( \mathcal{NP} \). Show that the Knapsack Language is \( \mathcal{NP} \)-complete by reducing the Circuit–SAT to it.

Remark: A similar problem will be discussed in class.

Hint: Assign a boolean variable to each input bit and each gate. For each variable construct a number and add it to the set \( W \). Then every subset \( S \) of \( W \) corresponds to an assignment of boolean values to the variables: a number is in \( S \) if and only if the corresponding boolean variable is true.
2 Circuit SAT and Three Coloring

Definition 4 (3-COL Language) The Three Coloring Language (3-COL) is the set of graphs that are three colorable.

Recall that a graph \( G = (V, E) \) is three colorable if there exists a coloring of the vertices of the graph in three colors such that the colors of adjacent vertices are distinct.

Definition 5 (Circuit–SAT Language) The Circuit–SAT Language is the set of satisfiable circuits (i.e., those circuits \( C \) for which there exists an input \( x \) such that \( C(x) = 1 \)).

3 Oracles and Self Reducibility

In this section we will see that for many sets \( L \), solving the decision problem (answering whether \( x \in L \)) implies an efficient solution for the search problem, of finding an NP-witness for \( x \).

Assume that there exists a powerful oracle that answers the question whether a string \( x \) belongs to \( L \). We can send requests to the oracle using a special query “Is \( x \) in \( L \)?”. If \( x \in L \) the oracle returns 1 (or true), otherwise 0 (or false).

Problem 4 1. Given an oracle for Circuit–SAT design a polynomial time algorithm that finds a witness (namely a satisfying assignment if one exists) for Circuit–SAT problem.

   Hint: Try to determine the bits of a satisfying assignment one at a time, using the given oracle on the appropriate restricted circuits.

2. Given an oracle for 3-COL design a polynomial time algorithm that finds a three coloring of a graph.

   Hint: One possible way is to determine the colors of vertices one at a time. To impose a partial coloring condition one can add a triangle to the graph, and connect subsets of its vertices to specific vertices to impose their color.