CSC 373 - Algorithm Design, Analysis, and Complexity **Summer 2016**

Lalla Mouatadid

SUBSET SUM

NP-hard problems often fall into one of the following categories:

- **Packing** problems: Independent set for instance where we want to pack a subset of the vertices with a certain properties.
- **Covering** problems: Vertex cover for instance, where the goal is to cover all the edges of the graph.
- **Constraint Satisfaction** problems, such as SAT and Circuit-SAT, where the goal is to satisfy a formula.
- **Sequencing** problems: Directed Hamiltonicy for isntance, where we want to find a way to visit every vertex in the graph, thus creating a sequence of vertices with specific properties.
- **Partitioning** problems such as colouring, where the goal is to partition the vertices of the graph into colour classes.
- And finally **Numerical** problems where the problems involves the handling (some sort of arithmetic operations for example) of numbers to satisfy some property or compute a specific value.

Today, we cover an example that falls in the last category: the **SUBSET SUM** problem (SS). The problem is defined as follows:

SUBSET SUM
Input: A set $S = \{w_1, w_2, \ldots, w_n\}$ of natural numbers, and a target number W.
Problem Return 1 if and only if there is a subset $S' \subseteq S$ that adds up to precisely W.

In this lecture, we prove the following theorem:

Theorem 1. *SUBSET SUM is NP-complete.*

In order to diversify the type of reductions we have seen so far, we will introduce an NP-complete problem, called 3-DIMENSIONAL MATCHING (3DM), and use it to give a reduction from 3DM to SS. Most reductions we have seen use objects that we are familiar with (vertices, boolean variables etc), and the reduction often seemed a natural way to approach the problem. Today's reduction is slightly trickier in the sense that we have to somehow produce integers and perform some sort of arithmetic operations on them. Before we get more formal, we first define the 3DM problem.

Recall the bipartite matching problem, where given a bipartite graph $G(X \cup Y, E)$, we wanted to find a maximum subset of edges that do not share any endpoints. We say that a matching M is **perfect** if every vertex in $X \cup Y$ is covered by an edge in M. It is easy to see that if G has a perfect matching, then $|X| = |Y|$ for otherwise at least one vertex is not matched.

We can forget about the idea of a graph, and talk about sets instead. In particular, think of X and Y as two sets of equal size n, let $T \subseteq X \times Y$ be a set of pairs of the forms $(x_i, y_j), i, j \in [n]$. Asking for a perfect matching is then equivalent to asking for a subset $T' \subseteq T$ such that every element $x_i \in X, y_j \in Y$ is in exactly one pair in T'.

Approaching the Perfect Matching problem from a set perspective allows us to see a natural generalization of the problem to 3 sets X, Y, Z. The decision problem of **3-Dimensional perfect matching** is defined as

follows:

3D MATCHING
Input: Three sets X, Y, Z of equal size n, and set $T \subseteq X \times Y \times Z$ of triple (x_i, y_j, z_k) such that $x_i \in X, y_j \in Y, z_k \in Z$.
Problem Return 1 if and only if there is a subset $T' \subseteq T$ such that every element of $X \cup Y \cup Z$ is in exactly one triple.

Note: This generalization from 2 sets X, Y to 3 sets X, Y, Z can and is easily seen through graphs too. We call these graph hypergraphs. For fear to digress, I'll stop here but a quick google search will tell you all(all??) you need to know about hypergraphs.

Ok so for the remaining of the lecture, we prove Theorem 1.

Proof of Theorem 1. We first show $SS \in NP$. To this end, consider an instance of SS, a certificate would be an encoding of a subset $S' = \{w_{i_1}, w_{i_2}, \ldots, w_{i_n}\}$ that is supposed to sum up to W. In poly-time, one can compute the sum of these numbers of verify if the sum does indeed equal W.

Second, we show that SS is NP-hard by giving a reduction from 3DM: 3DM $\leq_p$ SS. This means, given an instance X, Y, Z, T of 3DM, we construct an instance of SS S, W such that X, Y, Z has a 3D perfect matching iff S has a subset that sums ups to exactly W.

The trick is to encode the manipulation of sets via the addition of integers. Consider an instance of 3DM: X, Y, Z each of size n and $T \subseteq X \times Y \times Z$ such that $|T| = m$.

A common way to represent sets is via bit vectors. Each entry in the vector corresponds to a different element in the set, and it holds a 1 iff the set contains that element. We use this approach to represent the triples of T.

In particular, for each $t \in T$, where $t = (x_i, y_j, z_k)$, we construct a corresponding number w_t with $3n$ digits where we place a 1 in positions $i, n + j, 2n + k$ and a 0 everywhere else. Basically what we are doing is expression w_t is some base $d > 1$ where $w_t = d^{i-1} + d^{n+j-1} + d^{2n+k-1}$.

Recall that for 3DM, the goal is to select a set of triple that includes every element exactly one. Suppose we select t_1, t_2, then this means $t_1 \cup t_2$ is part of the solution. Taking the union of triples is almost equivalent to adding integers, since in the setting we have: the 1s fill the places where there is an element in any of the sets. BUT addition includes carries! If we have too many 1s in the same column, they will roll over and produce a non-zero entry in the next column.

A simple trick to handle this problem is to work in a base large enough that will prevent us from having carries. Since the maximum number of 1s we can have in any column is m 1s, It suffices to set the base d to $m + 1$.

Ok great, we now have produced numbers $w_1, \ldots, w_m$ for each triple $t_1, \ldots, t_m \in T$. We still need to decide what W in order to have a complete instance of SS. We construct the following instance:

- For each triple $t = (x_i, y_j, z_k) \in T$, we construct a number in base $m + 1$ as defined above.
- We define W to be the number in base $m + 1$ with $3n$ digits, each of which is equal to 1:

$$W = \sum_{i=0}^{3n-1} (m + 1)^i$$

We now claim that the set of triple T contains a perfect 3D matching iff there exists a subset of the numbers $\{w_t\}$ that adds up to exactly W.

First, suppose we have a 3D matching that consists of triples $t_1, t_2, \ldots, t_n$. In the sum $w_{t_1} + w_{t_2} + \ldots + w_{t_n}$, there is a single 1 in each of the $3n$ digit positions by construct, and so the result is equal to W (by construction of W also).

Conversely, suppose now we have a set of numbers $w_{t_1}, w_{t_2}, \ldots, w_{t_k}$ that adds up to W. Since each w_{t_i} has 3 1s in its representation, and there are no carries, then $k = n$. It follows that for each of the $3n$ digit positions, exactly 1 of the w_{t_i} has a 1 in that position, therefore $t_1, \ldots, t_n$ is a perfect 3D matching. $\square$