Algorithms

NP-completeness of the Subset Sum problem

The Subset Sum (SS) problem is: Given a set of \( n+1 \) integers \( a_1, a_2, \ldots, a_n, b \) is there a subset of \( a_1, a_2, \ldots, a_n \) that sums exactly to \( b \)? It is easy to show that this problem is in NP.

We now prove that CNF-SAT is reducible to SS. Suppose we are given a Boolean formula \( F(x_1, x_2, \ldots, x_n) \). For convenience, we let \( x_{i+n} = \overline{x_i} \) for \( i = 1, 2, \ldots, n \). We introduce variables \( X_1, X_2, \ldots, X_{2n} \) corresponding respectively to the Boolean variables \( x_1, x_2, \ldots, x_{2n} \). The integer variables will assume values 0 or 1; 0 will correspond to Boolean False and 1 to Boolean True. Then, we write the following system of inequalities and equations in the \( X_1, X_2, \ldots, X_{2n} \) which we will show is valid if \( F \) is satisfiable:

\[
egin{align*}
1 &\geq X_i \geq 0 \text{ for } i = 1, 2, \ldots, 2n \\
X_i + X_{i+n} &= 1 \text{ for } i = 1, 2, \ldots, n \\
\text{For each clause } C \text{ in } F, \sum_{x_i \in C} X_i &\geq 1.
\end{align*}
\]

The last condition gives us one inequality per clause. I have used the somewhat loose notation \( x_i \) “belongs” to \( C \) to denote that \( x_i \) is one of the disjuncts involved in \( C \).

Now consider the decision problem: Does there exist a set of integers \( X_1, X_2, \ldots, X_{2n} \) satisfying the system of inequalities? We will reduce this problem in turn to Subset Sum. First, we introduce some new integer variables called “slack variables” to convert the inequalities corresponding to the clauses into equations. So \( X_{i_1} + X_{i_2} + \ldots + X_{i_k} \geq 1 \) will be replaced by \( X_{i_1} + X_{i_2} + \ldots + X_{i_k} - Y_1 - Y_2 - \ldots - Y_{k-1} = 1 \); \( 1 \geq Y_1, Y_2, \ldots, Y_{k-1} \geq 0 \) where \( Y_1, Y_2, \ldots, Y_{k-1} \) are new variables not used elsewhere. Let us call the vector...
of all variables including the slack variables \( z \). Then we can write the new system in matrix notation (with a suitable matrix \( A \) and a suitable vector \( c \)) as:

\[
Az = c \quad 1 \geq z \geq 0.
\]

(Note : I am using 1,0 to indicate the vector of all 1 ’s and all 0’s respectively in the last line.) \( z \) has at most \( l = 2nk \) components where \( k \) is the number of clauses in \( F \). We will now “aggregate” all the equations into one to get a SubsetSum problem. Note that the entries of \( A \) are all \( 0, \pm 1 \).

Let \( m \) be the number of rows of \( A \) and for \( i = 1, 2, \ldots m \), let \( a_i \) denote the \( i \) th row of \( A \). For any \( z \) with 0-1 components, the dot product of \( a_i \) and \( z \) is an integer between \(-l\) and \(+l\). So for 0-1 vectors \( z \), the vector \( Az \) has \( m \) components each between \(-l\) and \( l \). The following lemma will be used directly:

**Lemma** Suppose two integer vectors \( u \) and \( v \) each has \( m \) integer components in the range \(-l\) to \( l \). Then \( u = v \) iff

\[
\sum_{i=1}^{m} (2l+1)^i u_i = \sum_{i=1}^{m} (2l+1)^i v_i.
\]

**Proof** : \( (\Rightarrow) \) is obvious.

To prove the other way, suppose

\[
w = \sum_{i=1}^{m} (2l+1)^i u_i - \sum_{i=1}^{m} (2l+1)^i v_i = 0.
\]

Further, for contradiction, assume that \( u \neq v \). Let \( k \) be the largest index \( i \) so that \( u_i \neq v_i \). Without loss of generality, assume that \( u_k > v_k \). Then this contributes at least \((2l+1)^k\) to \( w \). This contribution must be cancelled by \( i < k \) to get \( w \) to be zero. But

\[
| \sum_{i=1}^{k-1} (2l+1)^i u_i - \sum_{i=1}^{k-1} (2l+1)^i v_i | \leq (2l) \sum_{i=1}^{k-1} (2l+1)^i = (2l+1)^k - 1.
\]
So cancellation is not possible proving the lemma by contradiction.

[Here is the intuition behind the lemma: think of \(u, v\) as integers represented to the base \(2l + 1\). The sums in the lemma are precisely their values as integers.]

Now, using the lemma, it is clear that the system

\[
Az = c \quad 1 \geq z \geq 0
\]

has an integer solution iff the following one equation has a 0-1 solution:

\[
\sum_{i=1}^{m}(2l + 1)^i(a_i \cdot z) = \sum_{i=1}^{m}(2l + 1)^i c_i.
\]

The last is a SubsetSum Problem. (Why?)