

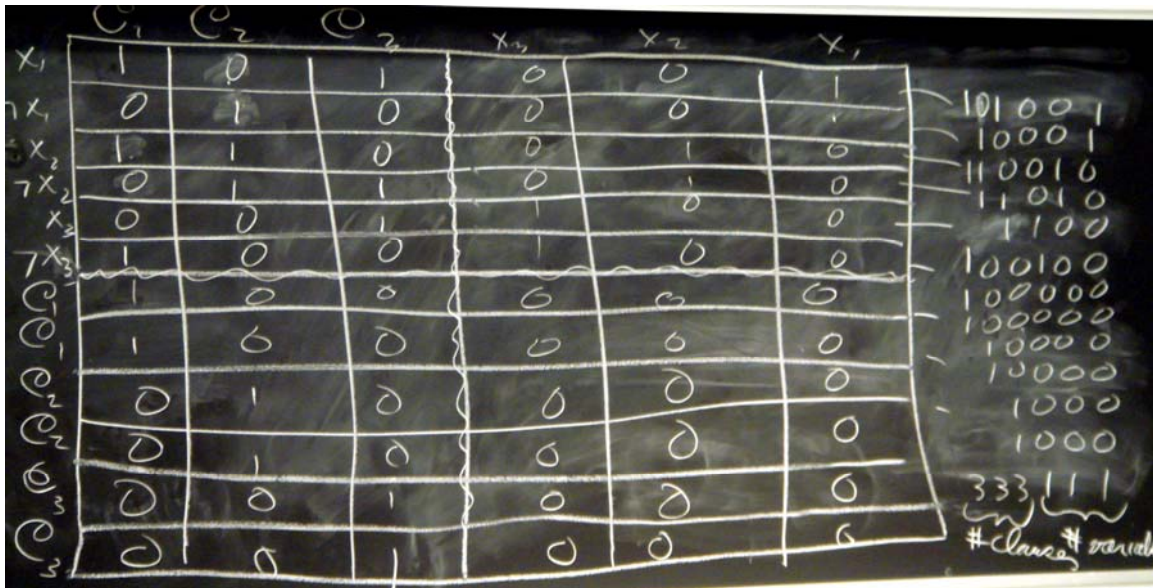
CSCI 303 Introduction to Algorithms
 Spring 2007
 February 26th, 2007 class notes

The midterm may cover all of the material taught in class so far, including the current topic: NP-completeness. HW1-7.

From last class:

	C_1	C_2	C_3	$x_3 \neg x_3$	$x_2 \neg x_2$	$x_1 \neg x_1$
x_1	1	0	1	0	0	1
$\neg x_1$	0	1	0	0	0	1
x_2	1	1	0	0	1	0
$\neg x_2$	0	1	1	0	1	0
x_3	0	0	1	1	0	0
$\neg x_3$	1	0	0	1	0	0
C_1	1	0	0	0	0	0
C_1	1	0	0	0	0	0
C_2	0	1	0	0	0	0
C_2	0	1	0	0	0	0
C_3	0	0	1	0	0	0
C_3	0	0	1	0	0	0

Note that bottom right is all 0s, top right and bottom left are “double diagonals” and the top left (yellow) has a 1 in cell C_i, l_j if literal l_j appears in clause C_i , and a 0 otherwise.

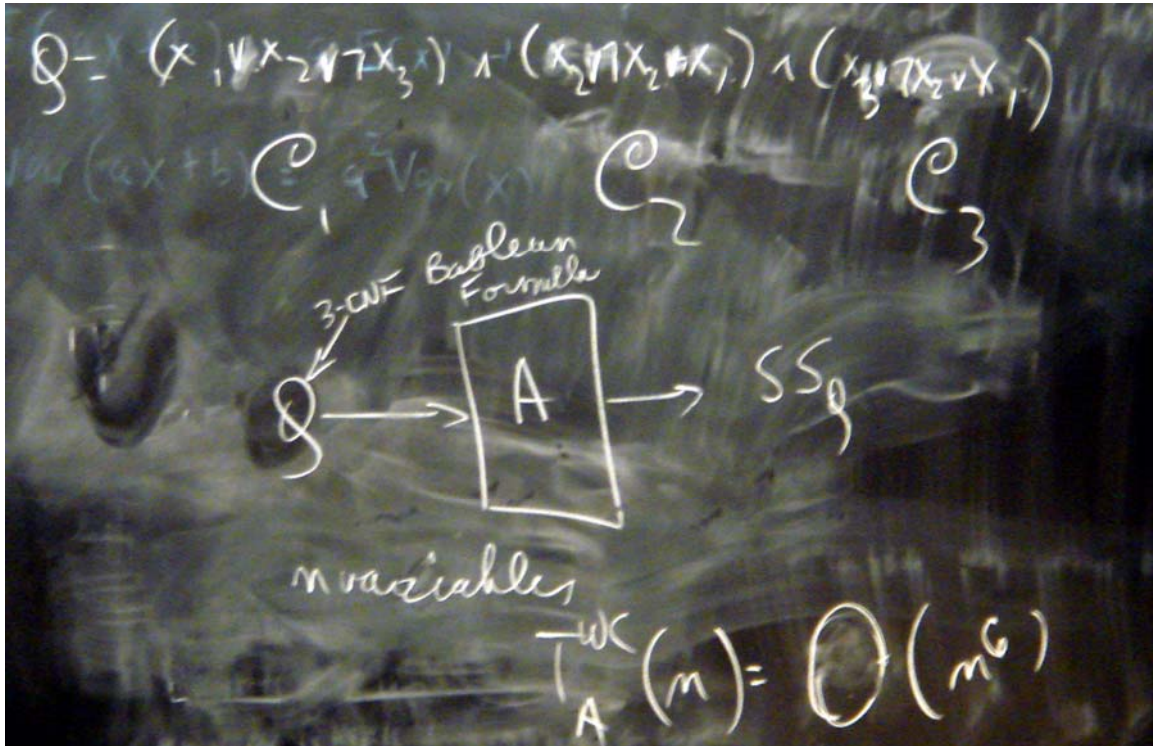


Target number: a 3 for every clause followed by a 1 for every variable. In this case: 333111.

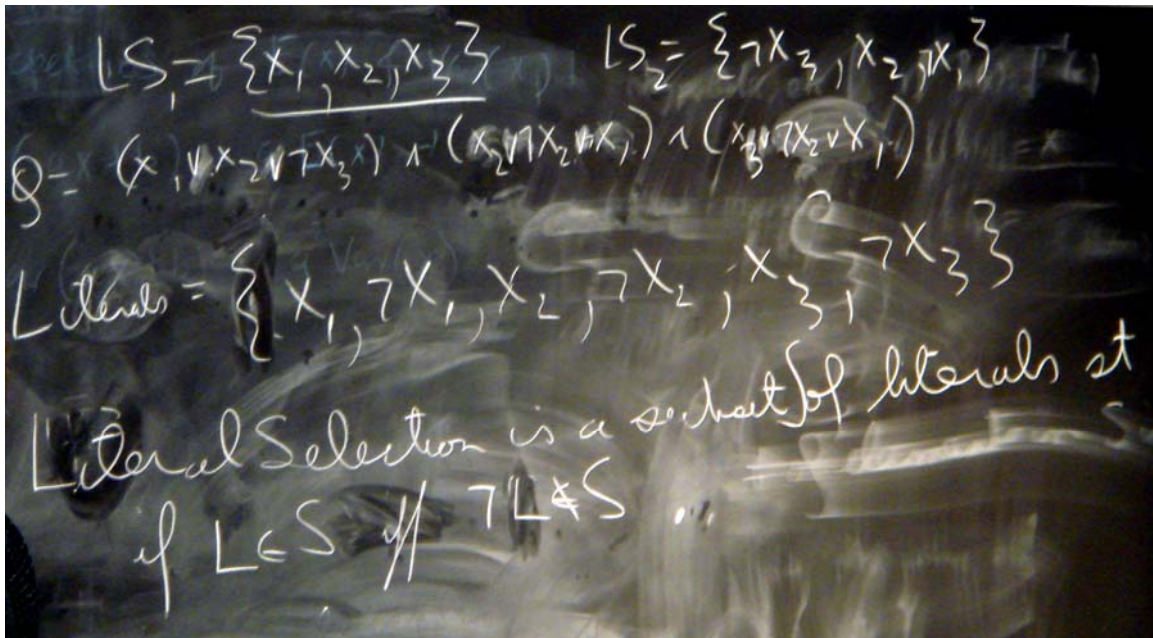
Let algorithm A convert Φ with n variables into the corresponding Subset Sum problem. Each Φ has 3 literals, so there are $8n^3$ different clauses. So the number of columns is

less than or equal to $8n^3 + n$ and the number of rows is less than or equal to $16n^3 + 2n$. So the size of the table is $O(n^6)$. So A can run in worst case $O(n^6)$ time.

$$T_A^{WC}(n) = O(n^6)$$



Literal selection is a subset S of the literals such that literal $l \in S \Leftrightarrow \neg l \notin S$.



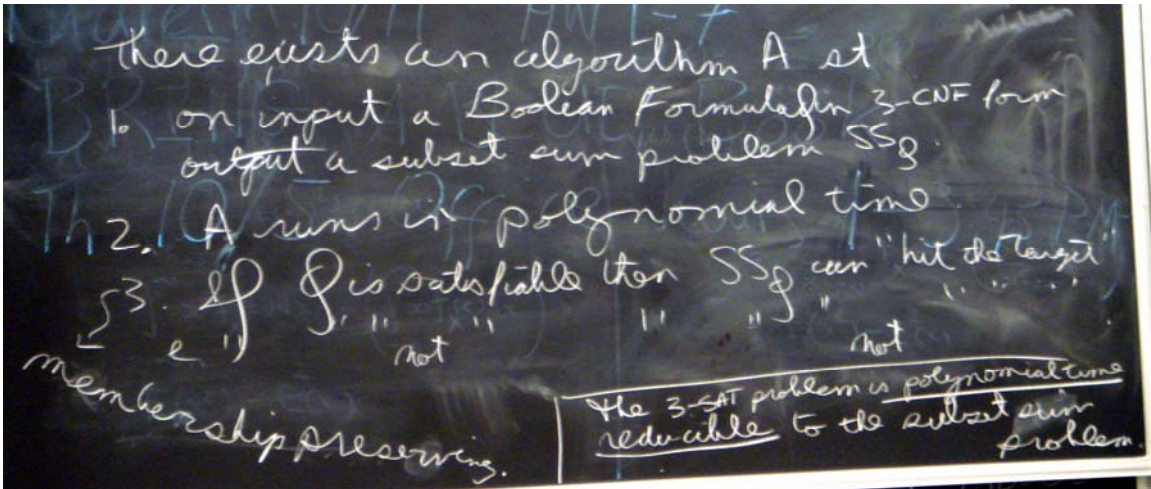
Given a literal selection, we can pick the rows of the matrix that correspond to those literals from the top part of the table, and extra number from the bottom part to complete the sum. If our literal assignment did not satisfy Φ , then we'll have a 0 in one of the "3" columns and we won't be able to complete the sum.



So there exists an algorithm A such that:

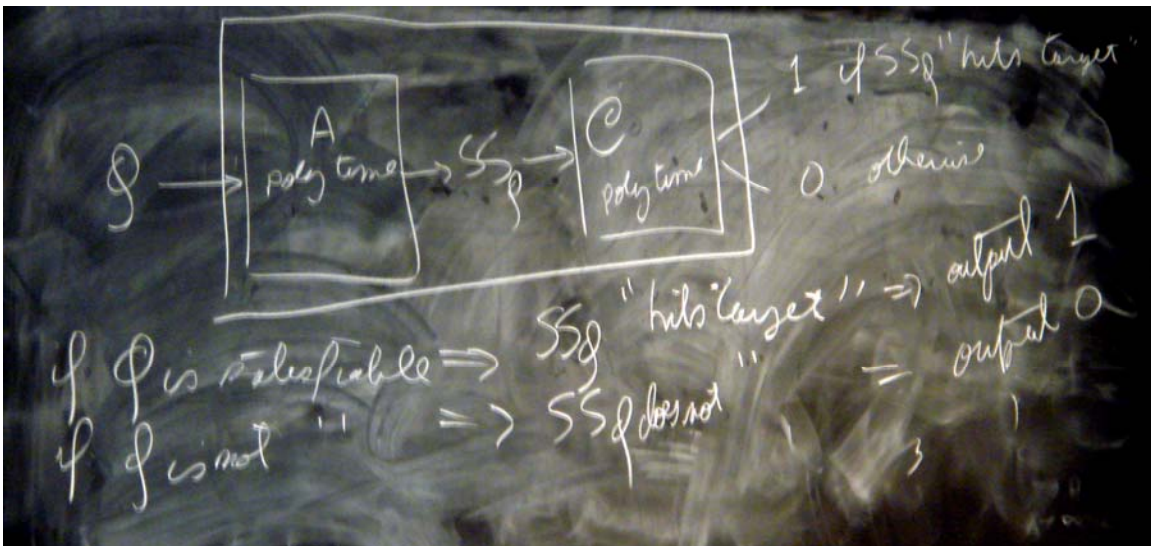
1. On input a Boolean formula Φ in 3-CNF, A outputs a subset sum problem SS_Φ .
2. A runs in polynomial time.
3. If Φ is satisfiable then SS_Φ can "hit the target" and if Φ is not satisfiable then SS_Φ cannot "hit the target"

(condition 3 is known as membership preserving)



We say that 3-SAT problem is polynomial time reducible to the Subset Sum problem.

So, if we find a polynomial time algorithm C to solve Subset Sum, we can convert it to solve 3-SAT in polynomial time by first translating the Φ to SS_Φ and then using C on SS_Φ .



All NP-Complete problems are polynomial time related, so they are either all solvable in polynomial time or none of them are.