The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual
Chapter 7, Additional Problem 1

The problem is clearly in NP: we need only guess a truth assignment and count the number of clauses it satisfies in NAE fashion, then compare the result with the bound. We prove the problem NP-complete by reducing *NAE3SAT* to it. Given an instance of *NAE3SAT* with variables $V = \{x_1, \ldots, x_n\}$ and clauses $C = \{\{\hat{x}_i, \hat{y}_i, \hat{z}_i\}, 1 \leq i \leq r\}$ (where $\hat{x}$ is used to denote one of x or $\overline{x}$), we construct an instance of *NAEMax2SAT* with the same variables and where each clause, $\{\hat{x}, \hat{y}, \hat{z}\}$, is replaced by 6 clauses as follows: $\{\hat{x}, \hat{y}\}$ $\{\hat{x}, \hat{z}\}$ $\{\hat{y}, \hat{z}\}$
Observe that, whenever the original clause has a satisfying assignment (1 or 2 literals set to true), then exactly 2 of the 3 constructed clauses are satisfied in NAE fashion (i.e., have exactly one literal set to true), whereas none of the 3 clauses is satisfied in that fashion when the original clause has no satisfying assignment (all literals true or all literals false). Thus we should set K to $2m$, where m is the number of clauses in the *NAE3SAT* instance. The only problem left is repetitions: our construction can easily produce multiple copies of the same 2-clause (not within a single group of 3, but between groups). We can handle this minor problem in a very simple way: whenever variable x is involved in copies, create variables x_i (one for each copy) use the x_i in the copy rather than x; for each x_i thus created, add the clause $\{x, \overline{x}_i\}$, and increase the bound by 1. By increasing the bound by 1, we force the additional clause to be satisfied in NAE fashion (because there is no way to satisfy more than 2 out of the 3 clauses in each original group, no matter whether the variable used there is called x or x_i), i.e., we force x and x_i to have the same truth value, so that every appearance of x_i is equivalent to an appearance of x. The proof follows immediately.

A brute-force approach to the same problem simply creates m new variables for each original variable of *NAE3SAT*; given the original clause $c = \{\hat{x}, \hat{y}, \hat{z}\}$, we replace it by 3 clauses as follows: $\{\hat{x}_c, \hat{y}_c\}$ $\{\hat{x}_c, \hat{z}_c\}$ $\{\hat{y}_c, \hat{z}_c\}$
We then add clauses of the type $\{x, \overline{x}_i\}$, $1 \leq i \leq m$, and increase the bound by m for each original variable, or by mn overall. This transformation creates mn variables and $m(n+3)$ clauses, and sets the bound to $K = m(n+2)$; it is clearly doable in polynomial time, in spite of the typically large number

of useless variables (a lot of them never appear in any of our groups of 3) created.

Somewhat harder would be to reduce something like *Max2SAT* to our problem: this would entail introducing symmetry and thus much the same trick as used in reducing *3SAT* to *NAE3SAT* in the notes.