The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual

Chapter 7, Problem 7.1

- We prove this problem NP-complete (membership in NP is obvious) by transforming *1in3SAT* to it. We have four cases to consider, according to how many complemented literals the *1in3SAT* clause has. If no complemented literals appear, then we leave the clause unchanged and place it in the set of clauses for *Positive 1in3SAT*. The other three cases require changes.

  1. One complemented literal: $\{\overline{a}, b, c\}$. Replace by the clauses $\{a, \alpha, \beta\}$, $\{a, \alpha, \gamma\}$, $\{\beta, \gamma, \delta\}$, and $\{b, c, \alpha\}$. The first three clauses force $\beta = \gamma =$ False and $\delta =$ True, which in turn forces $a = \overline{\alpha}$, so that the last clause becomes exactly the same as the replaced clause.
  2. Two complemented literals: $\{\overline{a}, \overline{b}, c\}$. Replace by the clauses $\{a, \alpha, \beta\}$, $\{b, \alpha, \gamma\}$, $\{c, \beta, \gamma\}$, $\{\alpha, \delta, \epsilon\}$, $\{\alpha, \delta, \zeta\}$, and $\{\delta, \epsilon, \zeta\}$. The last three clauses force $\delta =$ True and $\epsilon = \zeta =$ False, which in turn forces $\alpha =$ False, which, through the first two clauses, forces $a = \overline{\beta}$ and $b = \overline{\gamma}$. Thus the third clause becomes exactly the same as the replaced clause.
  3. Three complemented literals: $\{\overline{a}, \overline{b}, \overline{c}\}$. Replace by the clauses $\{a, \alpha, \beta\}$, $\{b, \alpha, \gamma\}$, $\{c, \alpha, \delta\}$, and $\{\beta, \gamma, \delta\}$. Taken with the first three clauses, the fourth clause forces $\alpha =$ False; through the first three clauses, this in turn forces $a = \overline{\beta}$, $b = \overline{\gamma}$, and $c = \overline{\delta}$, so that the last clause becomes exactly the same as the replaced clause.

  Each clause has its own set of new variables. It is easily verified by exhaustion that the new set of clauses admits a solution if and only if the original one does.

  Note that our strategy for clauses with two complemented literals suggests another transformation: since we can use three clauses to force some of its variables to True and some to False, we can then use one of the variables forced to False, call it ϵ, in clauses of the type

$\{x, x', \epsilon\}$, which forces x and x' to behave as complements of each other, so that we can now replace every complemented occurrence of x by x'. In fact, we can simplify the whole thing quite a bit: simply set up variables x and x' for each original variable x and add a single new variable, d. For each original clause, write a corresponding clause where complemented literals have been replaced by the prime variables; for each original variable x, write a new clause $\{x, x', d\}$. Note that d cannot be set to True, since that would force all primed and unprimed variables to False; thus d must be set to False, which means that x and x' must have opposite truth values.

- We prove this problem NP-complete (membership in NP is obvious) by transforming *NAE3SAT* to it. We have four cases to consider, according to how many complemented literals the *NAE3SAT* clause has. If no complemented literals appear, then we leave the clause unchanged and place it in the set of clauses for *Positive NAE3SAT*. The other three cases require changes.

  1. One complemented literal: $\{\overline{a}, b, c\}$. Replace by the clauses $\{a, \alpha, \beta\}$, $\{a, \alpha, \gamma\}$, $\{a, \alpha, \delta\}$, $\{\beta, \gamma, \delta\}$, and $\{b, c, \alpha\}$. The first four clauses force $a = \overline{\alpha}$, so that the last clause becomes exactly the same as the replaced clause.
  2. Two complemented literals: $\{\overline{a}, \overline{b}, c\}$. Simply replace by the clause $\{a, b, \overline{c}\}$ with one complemented literal and apply the replacement given above.
  3. Three complemented literals: $\{\overline{a}, \overline{b}, \overline{c}\}$. Simply replace by the clause $\{a, b, c\}$.

  Each clause has its own set of new variables. It is easily verified by exhaustion that the new set of clauses admits a solution if and only if the original one does.

  We can also use systematically variables x' and x'' to represent literals x and $\overline{x}$; for each clause, say with variables x, y, and z (regardless of complementation), we introduce auxiliary clauses $\{x', x'', y'\}$, $\{x', x'', y''\}$, $\{y', y'', z'\}$, $\{y', y'', z''\}$, $\{z', z'', x'\}$, and $\{z', z'', x''\}$—and, needless to add, we rewrite the original clause itself of in terms of our new variables. Successive clauses require some care to avoid multiple copies of the same clause.

- We prove this problem NP-complete (membership in NP is obvious) by transforming *3SAT* to it.

  Our transformation produces instances with all the variables used in the original instance, plus additional variables as needed. Given a clause in *3SAT*, two possibilities arise: (i) the clause is already in monotone form, in which case it is written without change into the transformed instance; and (ii) it includes both complemented and uncomplemented literals, in which case we need to modify it. For each clause of this second type, we use two additional variables and produce a total of three clauses; we distinguish two cases. In the first case, the clause contains one complemented and two uncomplemented literals, say $c = \{\overline{x}, y, z\}$; we then produce two clauses with uncomplemented literals and one with complemented literals: $\{a_c, y, z\}$, $\{b_c, y, z\}$, and $\{\overline{a}_c, \overline{b}_c, \overline{x}\}$. (This is really just splitting the original clause into two pieces connected by one auxiliary variable, as in the reduction from *SAT* to *3SAT*; the second auxiliary variable is a redundant copy of the first, used solely to ensure that the split clauses all have three literals each.) In the second case, proportions are reversed; from a clause $c = \{\overline{x}, \overline{y}, z\}$ with two complemented literals, we produce two clauses with complemented literals, $\{\overline{x}, \overline{y}, \overline{a}_c\}$, and $\{\overline{x}, \overline{y}, \overline{b}_c\}$, and one with uncomplemented literals, $\{z, a_c, b_c\}$. The transformation is easily carried out in polynomial time. The reader can verify that the transformed instance admits a solution if and only if the original one does.

- We prove this problem NP-complete by transforming *NAE3SAT* to it.

  Given an instance of *NAE3SAT* with variables $V = \{x_1, \dots, x_n\}$ and clauses $C = \{\{\hat{x}_i, \hat{y}_i, \hat{z}_i\}, 1 \le i \le r\}$ (where $\hat{x}$ is used to denote one of x or $\overline{x}$), we construct an instance of *2SAT* with the same variables and where each clause, $\{\hat{x}, \hat{y}, \hat{z}\}$, is replaced by 6 clauses as follows:
  $\{\hat{x}, \hat{y}\}$ $\{\hat{x}, \hat{z}\}$ $\{\hat{y}, \hat{z}\}$ $\{\overline{\hat{x}}, \overline{\hat{y}}\}$ $\{\overline{\hat{x}}, \overline{\hat{z}}\}$ $\{\overline{\hat{y}}, \overline{\hat{z}}\}$

  Observe that, whenever the original clause has a satisfying assignment (1 or 2 literals set to true), then exactly 5 of the 6 constructed clauses are satisfied, whereas only 3 of the 6 clauses are satisfied when the original clause has no satisfying assignment (all literals true or all literals false). Thus we should set K to $5r$. The only problem left is repetitions: our construction can easily produce multiple copies of the same 2-clause. We can handle this minor problem in a very simple way: whenever variable x is involved in copies, create variables x_i (as many as needed to handle all copies), use the x_i in the copy rather than the x, and add the clauses $\{x, \overline{x}_i\}$ and $\{\overline{x}, x_i\}$, both of which are forced to true by the bound (i.e., the bound is increased by two for each new x_i). The two clauses are true if and only if x and x_i are equivalent and the proof follows immediately.

The NAE version of this problem is also NP-complete: indeed, the transformation given above, but starting from *Positive NAE3SAT* and with a bound set to $4r$ (plus repeats) rather than $5r$ (plus repeats), will prove the assertion.