The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual
Chapter 7, Problem 7.46

That this problem is in D^p is easy to see. The problem "is an instance of *SAT* satisfiable with any of its clauses removed?" is in NP, since a certificate need only list, for each clause of the instance, the truth assignment that satisfies the instance with that clause removed. We only need to intersect that problem with the problem "is an instance of *SAT* unsatisfiable?", a problem which is of course our most basic coNP-complete problem.

We begin with a polynomial-time many-one reduction from *3UNSAT* to our problem. Let f be the formula in the *3UNSAT* problem with n variables and k clauses. Consider the set S of all possible truth assignments to the n variables of f; each clause "covers" some part of that set through the truth assignments in which all three of its literals are set to false. f is satisfiable if the union of the regions covered by its clauses does not equal S; it is minimally unsatisfiable if the union does equal S, but each clause covers a part not covered by any other clause. We create a new formula, g, in $n+k$ variables, using all variables of f, call them x_i, plus one new variable for each clause of f, call these new variables y_j. We shall set up, for each clause in f, a clause in g which acts like the clause of f when all y_js are set to false. Specifically, if $c_j = \{\tilde{x}, \tilde{y}, \tilde{z}\}$ is a clause of f, we place a clause in g which is simply $\{\tilde{x}, \tilde{y}, \tilde{z}, y_1, \ldots, y_{j-1}, y_{j+1}, \ldots, y_k\}$. Call these clauses of type (a). Each such new clause covers some unique territory, since it can be made false while setting variable y_j to true. Now we need only cover the rest of the uncovered territory in such a manner that each new clause has its own exclusive "patch." We begin by covering the rest of the region where y_j is true and all other new variables are false: for each clause $c_j = \{\tilde{x}, \tilde{y}, \tilde{z}\}$, we place three new clauses $\{\overline{\tilde{x}}, y_1, \ldots, y_{j-1}, \overline{y_j}, y_{j+1}, \ldots, y_k\}$, $\{\overline{\tilde{y}}, y_1, \ldots, y_{j-1}, \overline{y_j}, y_{j+1}, \ldots, y_k\}$, and $\{\overline{\tilde{z}}, y_1, \ldots, y_{j-1}, \overline{y_j}, y_{j+1}, \ldots, y_k\}$. Call these clauses of type (b). Finally, we cover regions where two or more new variables are set to true: for every pair $1 \le l < m \le k$, we add a clause $\{\overline{y_l}, \overline{y_m}\}$. Call these clauses of type (c). (Note that any truth assignment that sets at most one new variable to true satisfies all clauses of this type.)

Note that, if f is satisfiable, so is g (simply set all new variables to false, which reduces g to f). We claim that, if f is unsatisfiable, g is minimally so. First note that, if f is unsatisfiable, so is g: we can set at most one of the new variables to true, thereby satisfying all but one of the clauses of type (a) and all but three of the clauses of type (b); but these four clauses now read $\{\tilde{x}, \tilde{y}, \tilde{z}\}$, $\{\overline{\tilde{x}}\}$, $\{\overline{\tilde{y}}\}$, and $\{\overline{\tilde{z}}\}$, which is a contradiction. Let us now verify minimality. Suppose we remove a single clause of type (c), say $\{\overline{y_{i_1}}, \overline{y_{i_1}}\}$; we can then set both of these variables to true, automatically satisfying all clauses of type (a) and all clauses of type (b). Suppose we remove a single clause of type (b), say clause $\{\overline{\tilde{y}}, y_1, \ldots, y_{j-1}, \overline{y_j}, y_{j+1}, \ldots, y_k\}$. We then set variable y_j to true and all other new variables to false, thereby satisfying all clauses of type (c), all but one clause of type (a)—the jth clause—and all but two clauses of type (b)—the two remaining clauses derived from the original jth clause. These three unsatisfied clauses now reduce to $\{\tilde{x}, \tilde{y}, \tilde{z}\}$, $\{\overline{\tilde{x}}\}$, and $\{\overline{\tilde{z}}\}$, which we can easily satisfy by setting $\tilde{x} =$ false, $\tilde{y} =$ true, and $\tilde{z} =$ false. Finally, suppose we remove a single clause of type (a), say the jth such; we then set y_j to true and all other new variables to false. This satisfies all remaining clauses of type (a) (they all include y_j), all clauses of type (c), and all but three clauses of type (b), which reduce to $\{\overline{\tilde{x}}\}$, $\{\overline{\tilde{y}}\}$, and $\{\overline{\tilde{z}}\}$, which we can satisfy by setting $\tilde{x} =$ false, $\tilde{y} =$ false, and $\tilde{z} =$ false.

We now augment this reduction to make it a valid many-one, polynomial-time reduction from *3SAT* to *Minimal Unsatisfiability* and thus provide the two pieces needed in reducing *SAT/UNSAT* to *Minimal Unsatisfiability* and proving the latter to be D^p-complete. The augmentation is really quite simple—we have done all the hard work above. We simply add a single clause $\{y_1, y_2, \ldots, y_k\}$ that includes all our new variables. Now our new g' is unsatisfiable: we must set at least one new variable to true (because of our new clause) and may set at most one new variable to true (because of clauses of type (c)); setting one new variable to true will satisfy all but one of the clauses of type (a) and all but three of the clauses of type (b), but these four clauses then read $\{\tilde{x}, \tilde{y}, \tilde{z}\}$, $\{\overline{\tilde{x}}\}$, $\{\overline{\tilde{y}}\}$, and $\{\overline{\tilde{z}}\}$, which is a contradiction. If f is not satisfiable, then g' is not minimally unsatisfiable: we can remove the new clause, thereby reducing g' to g, which is still unsatisfiable as shown earlier. On the other hand, if f is satisfiable, then g' is minimally unsatisfiable. Certainly, removing the new clause turns g' into g, which is satisfiable when f is, as noted earlier. Removing a clause of any other type was covered earlier; note that in all three cases we set at least one of the additional variables (the y_is) to true, which also satisfies our new clause.

Finally, we need to combine our two reductions into a single reduction; this will give us the desired many-one reduction from the D^p-complete problem *SAT-UNSAT* to *Minimal Unsatisfiability*. Given the two instances of *Minimal Unsatisfiability* produced by our two reductions, we combine them into a single instance of *Minimal Unsatisfiability* so as to preserve minimality as follows: if the clauses of of the first instances are $c_1, \ldots, c_m$ and those of the second instance are $d_1, \ldots, d_n$—on an entirely disjoint set of variables, then the clauses of our single new instances are all clauses of the form $c_i + d_j$, for all pairs $1 \leq i \leq m$, $1 \leq j \leq n$. Thus removing a single clause from the new instance is equivalent to removing one clause from each of our two intermediate instances and the new instance remains unsatisfiable, since we are using every possible pair.