JOURNAL OF ALGORITHMS 2, 22–30 (1981)

The Complexity of Identifying Redundant and Essential Elements

SHLOMO MORAN

Department of Computer Science, University of Minnesota, Minneapolis, Minnesota 55455

AND

YEHOSHUA PERL

Department of Mathematics and Computer Science, Bar Ilan University, Ramat Gan, Israel

Received February 20, 1980

In many optimization problems a solution is a subset of optimum number of elements satisfying some desired property. An element is *redundant* if it does not belong to any solution of the problem. An element is *essential* if it belongs to every solution of the problem. We consider the complexity of indentifying redundant and essential elements in a sample of *NP*-Hard optimization problems. It is shown that these identification problems are also *NP*-Hard. The proofs are based on an analysis of the original reductions of Cook [The complexity of theorem proving procedures, *in* "Proceedings, Third Annual Assoc. Comput. Mach. Symposium on Theory of Computing," pp. 151–158, Assoc. Comput. Mach., New York, 1971] and Karp [Reducibility among combinational problems, *in* "Complexity of Computer Computations" (R. E. Miller and J. W. Thatcher, Eds.), pp. 85–104, Plenum, New York, 1972].

1. INTRODUCTION

In many optimization problems a solution is a subset of optimum number of elements satisfying desired property. Two examples are the minimum set cover problem and the maximum clique problem [4] (formal definitions are given in the next section). In the minimum set cover problem the subsets $S_1, S_2, \ldots, S_n$ of a set A are the elements of the problem, and in the maximum clique problem the vertices of a graph $G = (V, E)$ are the elements of the problem. These two optimization problems, as many others, are *NP*-Hard, since the corresponding decision problems are *NP*-Complete [4].

A possible approach in treating such a difficult problem is reducing the given problem to a problem of fewer elements. An element is *redundant* if it does not belong to any solution of the problem. An element is *essential* if

0196-6774/81/010022-090$02.00/0

it belongs to every solution of the problem. Clearly such an optimization problem can be reduced by eliminating the redundant elements. The problem can be further reduced, using the information about the essential elements. If, for example, $S_i \subset A$ is an essential subset in a minimum set cover problem then the reduced problem is finding a minimum cover of A-S_i by the subsets $S_j, j \neq i$. Or if a vertex v is an essential vertex in a maximum clique problem then the reduced problem is finding a maximum clique in the induced subgraph of the neighbors of v.

Such an approach is used in Switching Theory in minimization of switching functions by the Quine–McCluskey method [6]. Our work was motivated by a recent work of Choueka and Goldberg [2]. They apply the minimum set cover problem in automatic text analysis [7] to identify the relevant meanings of the words in the text out of the possible meanings of a word given in a dictionary.

In this work we consider the complexity of identifying redundant and essential elements in a sample of *NP*-Hard optimization problems. Note that it is easy to see that the problem of determining the number of the redundant elements for a solution containing k-elements is *NP*-Hard. For example for the set k-cover problem it implies from the equivalence of this problem to the problem: Is the number of redundant subsets for a k-cover smaller than n? That is, there exists a nonredundant subset for a k cover. However, the complexity analysis of the problem of identifying the number of redundant subsets for a minimum set cover is much more difficult.

The problems of identifying the redundant and essential elements for an optimization problem of n given elements can be presented in either a "constructive" manner or an "existential" manner, as follows:

The constructive problem: "Given an integer i, $1 \leq i \leq n$, is the ith element redundant (essential)?"

The existential problem: "Given an integer i, $1 \leq i \leq n$, are there i redundant (essential) elements?" Note that it is not clear whether these problems are in *NP*. Thus we shall only prove that these problems are *NP*-Hard. For this purpose it is enough to prove that the existential problems are *NP*-Hard, since this clearly implies that the constructive problems are *NP*-Hard too. However, we shall prove the *NP*-Hardness of both the constructive and the existential problems, for a sample of *NP*-Complete optimization problems, since the proof for the constructive problems serves as an introduction to the proof for the existential problem.

The technique we apply for proving the *NP*-Hardness of the above problems is embedding the problems of identifying the redundant and essential elements in the original reductions of Cook [3] and Karp [4]. Our proofs show that the properties of redundancy and essentiality of elements are actually conserved by Cook's and Karp's reductions. Thus adding to the extra properties conserved by those reductions, as investigated in [5, 8].

The conclusion of our result is that the problem of reducing some *NP*-Hard optimization problems to a smaller problem, using redundant and essential elements, is in general *NP*-Hard by itself.

Section 2 contains preliminary definitions. *NP*-Hardness of the constructive problems and the existantial problems is shown in Section 3 and 4, respectively.

2. Preliminary Definitions

Let $x_1, x_2, \ldots, x_n$ be Boolean variables. A *literal* σ is either a variable x_i or its negation $\bar{x}_i$. Let $L = C_i \cdot C_2 \cdot \cdots \cdot C_k$ be a logical formula in Conjunctive Normal Form (CNF), where each C_i is a clause presented as a sum of literals.

A formula L is *satisfiable* (*i-satisfiable*) if there exists a truth assignment for the variables of L satisfying all (at least i) of the clauses of L.

A literal σ in L is *i-essential* (*i-redundant*) if each truth assignment, satisfying at least i clauses of L, assigns σ the value 1 (0).

A literal σ is L is *essential* (*redundant*) if each truth assignment, satisfying the maximum possible number of clauses of L, assigns σ the value 1 (0). Note that a literal σ is essential (i-essential) if and only if its negation $\bar{\sigma}$ is redundant (i-redundant).

An occurrence of an essential (i-essential) literal σ in a clause C is an *essential occurrence* (*i-essential occurrence*) if all other literals in C are redundant (i-redundant). In particular, an occurrence of an essential literal σ in a clause C_i is an essential occurrence in case σ is the only literal occurring in C_i.

Example. Let $L = (x + y)(x + \bar{y})$. Only x is an essential literal in L, but none of its two occurrences is an essential occurrences.

Denote by $e(L)$ ($e_i(L)$) the number of essential occurrences (i-essential occurrences) of literals in a formula (i-satisfiable formula) L.

Note that for redundant literals there is no corresponding definition for the term of essential occurrence of a literal. The necessity of the term of essential occurrences of a literal for our proofs is demonstrated in connection with the proof of Corollary 1 in the next section. This fact explains also the difference in the formulation of Theorems 1 and 2 in the next section.

Let $r(L)$ ($r_i(L)$) denote the number of occurrences of redundant (i-redundant) literals in a formula (i-satisfiable formula) L.

Let L' be a subformula of a formula L. The definitions of L'-essential, L'-redundant L'-essential occurrence, $e(L')$ and $r(L')$ are as the above similar definitions for L.

Let $S_1, S_2, \ldots, S_n$ be subsets of a set A. Let $I \subset \{1, 2, \ldots, n\}$. A family $\{S_i | i \in I\}$ of subsets is a *set cover* of A if $\cup_{i \in I} S_i = A$. A *minimum set cover* (*set k-cover*) is a set cover of minimum number of (k) subsets.

Let $G = (V, E)$ be an undirected graph where V is the set of vertices and E is the set of edges of G. A *maximum clique* in G is a complete subgraph of G with the maximum number of vertices.

A *minimum vertex cover* in G is a subset $V' \subset V$, of minimum number of vertices, such that each edge in E is adjacent to at least one vertex of V'.

3. The Complexity of the Constructive Problems

In this section we prove that the constructive problem: "Is the ith element redundant (essential) for a given optimization problem?" is *NP*-Hard for a sample of problems.

The proofs are based on observations of the following reductions, the first of which is due to Cook [3] and the others are slight modifications of Karp's [4] reductions.

R_1. *Input*: (W, T, P) where W is a word over a given alphabet Σ, T is (the encoding of) a nondeterministic Turing machine (NDTM) and P is (the encoding of) a polynomial.

Output: A logical formula L is CNF, such that W is accepted by T in time $P(l(W))$, where $l(W)$ denotes the length of W, if and only if L is satisfiable.

R_2. *Input*: A logical formula $L = C_1 \cdot C_2 \cdot \cdots \cdot C_k$ presented in CNF.

Output: A graph $G = (V, E)$ such that $V = \{V_{\sigma,i} | \sigma$ is a literal occurring in the clause $C_i\}$ $E = \{(V_{\sigma,i}, V_{\zeta,j}) | i \neq j, \sigma \neq \bar{\zeta}\}$ such that for each integer i, L is i-satisfiable if and only if G contains a clique of i vertices.

R_3. *Input*: A graph $G = (V, E)$

Output: The complementary graph $\bar{G} = (V, E)$ such that for each integer i, G contains a clique of i vertices if and only if $\bar{G}$ contains a vertex cover of $|V| - i$ vertices.

R_4. *Input*: A graph $G = (V, E)$

Output: Subsets $S_1, S_2, \ldots, S_{|V|}$ of E defined as follows: $S_i = \{e | e \in E$ and is adjacent to $V_i\}$, such that for each integer i G contains a vertex cover of i vertices if and only if E has a set cover of i subsets.

THEOREM 1. *The problem of verifying whether a given occurrence of a literal σ, in a logical formula L given in* CNF, *is an essential occurrence, is NP-Hard.*

THEOREM 2. *The problem of verifying whether a given literal σ, in a logical formula L given in* CNF, *is a redundant literal, is NP-Hard.*

Before presenting the proofs of these two theorems, we bring three corollaries easily observed from the reductions R_2, R_3 and R_4.

COROLLARY 1. *The problems of verifying whether a vertex in a graph G is an essential (redundant) vertex for the maximum clique problem, is NP-Hard.*

Proof. The proof for essential vertices implies from Theorem 1 and the following observation concerning the reduction R_2: The vertex $V_{\sigma,i}$ is an essential vertex for the maximum clique problem in G if and only if the occurrence of the literal σ in the clause C_i of the logical formula L given in CNF is an essential occurrence.

The proof for redundant vertices implies from Theorem 2 and the following observation concerning the reduction R_2: The vertex $V_{\sigma,i}$ is a redundant vertex for the maximum clique problem in G if and only if σ is a redundant literal in L. □

In order to demonstrate the necessity of using the term of essential occurrence of a literal in the proof of Corollary 1 consider the logical formula $L = (x + y)(x + \bar{y})$. The graph G constructed by the reduction R_2 is given in Fig. 1.

Only x is an essential literal in L but none of its two occurrences is an essential occurrence. In the corresponding graph G there are three maximum cliques of two vertices each, but no vertex is essential for the maximum clique problem in G.

COROLLARY 2. *The problems of verifying whether a vertex in a graph G is an essential (redundant) vertex for the minimum vertex cover problem, is NP-Hard.*

Proof. Implies from Corollary 1 and the following observation concerning the reduction R_3: A vertex V is an essential (redundant) vertex for the minimum vertex cover problem in $G = (V, E)$ if and only if V is a redundant (essential) vertex for the maximum clique problem in the complementary graph $\bar{G} = (V, \bar{E})$. □

Note that the reduction R_3 transfers essential vertices to redundant vertices and vice versa.

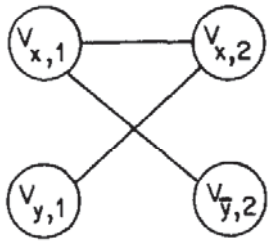

FIGURE 1

COROLLARY 3. *The problems of verifying whether a subset is an essential (redundant) subset for the minimum set cover problem is NP-Hard.*

Proof. Implies from Corollary 2 and the following observation concerning the reduction R_4: A vertex V_i in a graph G is an essential (redundant) vertex for the minimum vertex cover problem if and only if the corresponding subset S_i is an essential (redundant) subset for the minimum set cover problem defined in the reduction R_4. □

Similar results can be shown for more problems, for example, the feedback vertex set and feedback edge set problems [4].

Proof of Theorem 1. The proof is based on an analysis of Cook's reduction R_1. We refer to the presentation in [1 pp. 379–383]. Let (W, T, P) and L_w be the input and output for the reduction R_1, respectively. Let the logical formula L_w given in CNF contain k clauses. For each computation of T on input W, there corresponds a truth assignment for the variables of L_w (described in [1]) satisfying the followings:

At least $k - 1$ out of the k clauses of L_w are satisfied. The only clause which is probably not satisfied is the "accepting" clause G (see [1]) containing a single literal σ which is satisfied if and only if at the end of the computation T is in the "accepting state."

Thus if T accepts W in time $P(l(W))$ then the occurrence of the literal σ in the clause G is an essential occurrence since each truth assignment satisfying all k clauses of L_w assigns σ the value 1. On the other hand if T does not accept W in time $P(l(W))$ then no truth assignment satisfies all k clauses of L_w but there are truth assignments satisfying $k - 1$ clauses of L_w which assign σ the value 0. Thus, in this case the literal σ is not essential. Hence, the occurrence of σ is an essential occurrence if and only if T accepts W in time $P(l(W))$. Therefore the problem of verifying whether a given occurrence of a literal, in a logical formula L given in CNF, is an essential occurrence is *NP*-Hard. □

Proof of Theorem 2. As for Theorem 1 the proof is based on an analysis of Cook's reduction R_1. First we note that given a set A in *NP*, there exists a NDTM T' accepting A in polynomial time P having the following property: T' has a certain nonaccepting state q_N such that there is a legal computation of T' on each $W \in \Sigma^*$ for which T' enters the state q_N at the beginning of the computation and stays at q_N indefinitely.

Let (W, T', P) and L_w be the input and output for the reduction R_1, respectively. Let L_w contains k clauses. The formula L_w contains a literal σ which is satisfied if and only if T' is in the state q_N at the end of the computation.

There exists a truth assignment satisfying $k - 1$ out of the k clauses of L_w (excluding the accepting clause G) assigning σ the value 1, since there

exist a legal computation of T' on w in which T' enters the state q_N and stays at this state. If T' does not accept W, then L_w is $(k-1)$-satisfiable but not k-satisfiable and σ is not a redundant literal. On the other hand, if T' accepts W, then there exists a truth assignment satisfying all k clauses of L_w. Any such truth assignment assigns σ the value 0 since at the end of the computation T' is not at the state q_N but at the accepting state, and thus σ is a redundant literal.

Thus σ is a redundant literal if and only if T' does not accept W in time $P(l(W))$. Hence, the problem of verifying whether a given literal, in a logical formula given in CNF, is a redundant literal, is *NP*-Hard. □

4. The Complexity of the Existential Problems

In this section we show that not only the constructive problems, but even the simple existential problems: "Given an integer $i, 1 \leq i \leq n$, are there i essential (redundant) element?" are *NP*-Hard for a sample of optimization problems.

Let L' be a subformula of a formula L. Let $L - L'$ denote the formula obtained by deleting the subformula L' from L.

Lemma 1. *Let (W, T, P) and L_w be input and output of the reduction R_1, where T accepts the set $A \subset \Sigma^*$ in polynomial time P. Let G be the accepting clause of the formula L_w containing k clauses. Then*

(a) *If $W \in A$ then $e(L_w) > e(L_w - G)$.*

(b) *For each $W \in \Sigma$, $^*e(L_w - G) \geq e_{k-1}(L_w)$*

Proof. (a) The formula L_w is satisfiable since $W \in A$. Thus for each essential occurrence of a literal in $L_w - G$ there exists an essential occurrence of the same literal in L_w. On the other hand L_w contains one essential occurrence of the unique literal σ in the "accepting" clause G which does not appear in $L_w - G$. Hence $e(L_w) > e(L_w - G)$.

(b) Let ζ be an essential literal for the satisfiability of any $k-1$ clauses out of the k clauses of L_w. The set of truth assignments satisfying any $k-1$ clauses of L contains the set of truth assignments satisfying the $k-1$ clauses of $L_w - G$. Thus ζ is also an essential literal for the satisfiability of $L_w - G$. This implies that $e(L_w - G) \geq e_{k-1}(L_w)$. □

Theorem 3. *The following problem is NP-Hard: "Given a logical formula L in* CNF *and an integer i, does L contain i essential occurrences of literals?"*

Proof. Let A be a set in *NP* accepted by a NDTM in time P. The proof implies from the following reduction from the *NP*-Complete problem of recognizing A to the problem of finding the number of essential occurrences of literals in a formula L given in CNF, which is polynomially

equivalent to the decision problem of verifying, for a given integer i, whether L contains i essential occurrences of literals.

Let (W, T, P) and L_w be input and output for the reduction R_1. Denote by s and t the number of essential occurrences of literals in L_w and $L_w - G$, respectively, where G is the accepting clause of L_w. If $W \in A$ then $s = e(L_w)$ and if $W \notin A$ then $s = e_{k-1}(L_w)$. Thus by Lemma 1 $s > t$ if and only if $W \in A$, since if $W \in A$ then by (a) $s > t$ and if $W \notin A$ then by (b) $s \leq t$.

Hence, the problem of verifying whether $W \in A$ is reduced to the problem of finding the numbers s and t of essential occurrences of literals in two formulas given in CNF. □

LEMMA 2. *Let (W, T', P) and L_w be the input and output of the reduction R_1, where the* NDTM *T' accepting the set $A \in \Sigma^*$ is a special* NDTM *as described in the proof of Theorem 2. Let G be the accepting clause of the formula L_w containing k clauses. Then*

(a) *If $W \in A$ then $r(L_w) > r(L_w - G)$.*

(b) *For each $W \in \Sigma^*$, $r(L_w - G) \geq r_{k-1}(L_w)$*

Proof. (a) Each redundant literal of $L_w - G$ is also a redundant literal of L. Thus $r(L_w) \geq r(L_w - G)$.

As in the proof of Theorem 2, L_w contains a literal σ which is satisfied if and only if T' is at the state q_N in the end of the computation. This literal σ is a redundant literal of L_w but not a redundant literal of $L_w - G$. Hence $r(L_w) > r(L_w - G)$.

(b) A redundant literal for the satisfiability of $k - 1$ clauses out of the k clauses of L_w is also a redundant literal of the formula $L_w - G$ containing $k - 1$ clauses. Hence $r(L_w - G) \geq r_{k-1}(L_w)$. □

Similar to the proof of Theorem 3 one can apply Lemma 2 to prove the next Theorem.

THEOREM 4. *The following problem is NP-Hard: "Given a logical formula L in* CNF *and an integer i, does L contain i occurrences of redundant literals in L?"*

Theorems 3 and 4 can be applied now to prove the following corollary similarly to the proofs of Corollaries 1, 2, and 3.

COROLLARY 4. *The existential problems for the maximum clique problem the minimum vertex cover problem and the minimum set cover problem are NP-Hard.*

REFERENCES

1. A. V. AHO, J. E. HOPCROFT, AND J. D. ULLMAN, "The Design and Analysis of Computer Algorithms," Addison–Wesley, Reading, Mass., 1974.

2. Y. CHOUEKA AND D. GOLDBERG, Mechanical resolution of lexical ambiguity: A combinatorial approach, *in* "Proceedings, International Conference on Literary and Linguistic Computing, Israel, 1978, in press.
3. S. A. COOK, The complexity of theorem proving procedures, *in* "Proceedings Third Annual ACM Symposium on Theory of Computing," pp. 151–158, Assoc. Comput. Mach., New York, 1971.
4. R. M. KARP, Reducibility among combinatorial problems, *in*, "Complexity of Computer Computations" (R. E. Miller and J. W. Thatcher, Eds.) pp. 85–104, Plenum, New York, 1972.
5. N. LINCH AND R. J. LIPTON, "On Structure Preserving Reductions," Yale University, Technical Report 85.
6. E. J. MCCLUSKEY, "Introduction to the Theory of Switching Circuits," McGraw–Hill, New York, 1965.
7. G. SALTON, "Automatic Information Organization and Retrieval," McGraw–Hill, New York, 1968.
8. J. SIMON, "On Some Central Problems in Computational Complexity," Cornell University, Technical Report 75–224, 1975.