Cybernetics and Systems Analysis, Vol. 46, No. 6, 2010

REOPTIMIZATION OF SET COVERING PROBLEMS

V. A. Mikhailyuk

UDC 519.854

Abstract. *If an element is inserted into or removed from a set, then the set covering problem can be reoptimized with some ratio* $\left(2-\frac{1}{\ln m+1}\right)$, *where m is the number of elements of the set. A similar result holds if an arbitrary number* $1<p<m$ *of elements of the set is inserted or removed.*

Keywords: *reoptimization, r-approximation algorithm.*

At the present time, in solving NP-hard problems, a new computational paradigm has arisen, namely, reoptimization [1–7]. The most well-known approach to the solution of such problems lies in the development and use of approximation algorithms that yield a solution close to optimal in polynomial time. The quality of an approximate solution (usually measured as the ratio of values of approximate and optimal solutions) must be guaranteed. The line of investigation connected with the development of best approximation algorithms and classification of problems that is based on the approximation quality that is reached in polynomial time occupy an important place in theoretical investigations of informatics over the past two decades.

The concept of reoptimization is as follows. Let Π be some NP-hard (or, possibly, NP-complete) problem, and let I be its initial instance whose optimal solution is known. A new instance I' of the problem Π is proposed that is obtained as a result of some "insignificant" changes in the instance I. How can knowledge on the optimal solution of I be efficiently used for the computation of some exact or approximate solution of the instance I'?

The objective of reoptimization in applying approximate methods is the use of knowledge on the solution of the initial instance I (a) to achieve a better approximation quality (approximation ratio) of I', (b) to create a more efficient (more fast) algorithm for determination of an optimal (or close to optimal) solution to I', or (c) to provide the fulfillment of the first and second conditions.

The following results on the reoptimization of discrete optimization problems are well known. After the insertion of an elementary disjunction, the reoptimization of Max Weighted Sat (the weighted maximum satisfiability problem) is approximable with the ratio 0.81 though Max Weighted Sat is approximable with the ratio 0.77 [7]. After the insertion of a vertex into a graph, the reoptimization of Min Vertex Cover (the minimum vertex cover problem) is approximable with the ratio 1.5, and Min Vertex Cover is approximable with the ratio 2 [7]. After the insertion of a (terminal or nonterminal) vertex, the reoptimization of Min Steiner Tree (the minimum Steiner tree problem) is approximable with the ratio 1.5 and Min Steiner Tree is approximable with the ratio $1+\ln(3)/2\approx 1.55$ [4].

Worthy of mention is a series of works on the reoptimization of the Traveling Salesman Problem (TSP) [1–3, 5]. For example, the minimum metric distance TSP (Min TSP) is approximable with the ratio 1.5, its reoptimization after the insertion of a new node is approximable with the ratio 1.34, and the reoptimization of this problem after changing distances is approximable with the ratio 1.4 [7]. Approximation estimates for the general TSP (Min TSP) and for various versions of reoptimization are unknown.

V. M. Glushkov Institute of Cybernetics, National Academy of Sciences of Ukraine, Kyiv, Ukraine, *vickm@univer.lutsk.ua*. Translated from Kibernetika i Sistemnyi Analiz, No. 6, pp. 27–31, November–December 2010. Original article submitted December 10, 2009.

PRELIMINARY INFORMATION ON APPROXIMATION ALGORITHMS AND APPROXIMATION CLASSES OF DISCRETE OPTIMIZATION PROBLEMS

We introduce some concepts to be used in what follows [7].

Definition 1 [7]. An NP-optimization (NPO) problem Π is defined as a quadruple $(E, \mathrm{Sol}, m, \mathrm{opt})$ such that

• E is the set of instances of Π that are recognizable in polynomial time;

• for a given $I \in E$, $\mathrm{Sol}(I)$ is the set of feasible solutions I; for each $S \in \mathrm{Sol}(I)$, $|S|$ (the size of S) is polynomial with respect to $|I|$ (the size of I); for any given I and any S (polynomial with respect to $|I|$), $S \in \mathrm{Sol}(I)$ can be determined in polynomial time;

• for given $I \in E$ and $S \in \mathrm{Sol}(I)$, $m(I, S)$ is the (numerical) value of S; m is polynomially computable and is called the objective function;

• $\mathrm{opt} \in \{\min, \max\}$ is the type of an optimization problem.

For a given NPO problem $\Pi = \{E, \mathrm{Sol}, m, \mathrm{opt}\}$, we denote an optimal solution to an instance I from Π by $S^*(I)$ and its value by $m(I, S^*(I)) - \mathrm{opt}(I)$.

Definition 2 [7]. For a given NPO problem $\Pi = (E, \mathrm{Sol}, m, \mathrm{opt})$, an approximation algorithm A is an algorithm that, for a given instance I of Π yields a feasible solution $S \in \mathrm{Sol}(I)$.

If A is executed in polynomial time with respect to $|I|$, then A is called a polynomial approximation algorithm for Π.

The quality of an approximation algorithm is estimated as the ratio $\rho_A(I)$ (approximation ratio) between the value of an approximate solution $m(I, A(I))$ and the value of the optimal solution $\mathrm{opt}(I)$. Thus, for minimization problems, the approximation ratio is within the limits $[1, \infty)$ and that for maximization problems is within the limits $[0, 1]$.

NPO problems are classified according to the quality of approximation algorithms as follows.

Definition 3 [7]. An NPO problem Π belongs to the class *APX* if there is a polynomial approximation algorithm A and a rational number r such that, for a given I of Π, $\rho_A(I) \le r$ (accordingly, $\rho_A(I) \ge r$) if Π is a minimization (accordingly, maximization) problem. In this case, A is called an r-approximation algorithm (the problem Π is r-approximable by the algorithm A).

Examples of combinatorial optimization problems belonging to the class *APX* are Max Weighted Sat, Min Vertex Cover, and Min TSP. For some problems from *APX*, a stronger form of approximationness can be introduced.

For such problems, for any rational $r > 1$ (or $r \in (0,1)$ for maximization problems), there is an algorithm A_r and a suitable polynomial p such that A_r is an r-approximation algorithm whose time is measured as p in $|I|$. The family of algorithms A_r (parametrized with the help of r) is called the Polynomial Time Approximation Scheme (PTAS).

Definition 4 [7]. An NPO problem Π belongs to the class *PTAS* if, for any rational $r > 1$ (accordingly, $r \in (0, 1)$) and any instance I of Π, there is a polynomial approximation scheme A_r such that $\rho_{A_r}(I) \le r$ (accordingly, $\rho_{A_r}(I) \ge r$) for a minimization (maximization) problem Π.

Note that, in the definition of *PTAS*, the execution time of the algorithm A_r is polynomial with respect to the size of its input, but it can be exponential with respect to $r-1$. A better situation arises when the execution time is polynomial with respect to the input size and with respect to $r-1$ (or $1-r$ for maximization problems). In this case, the corresponding algorithm is called a fully polynomial time approximation scheme (FPTAS).

Definition 5 [7]. An NPO problem belongs to the class *FPTAS* if it allows for the fully polynomial time approximation scheme.

If $P \ne NP$, then we have $FPTAS \subset PTAS \subset APX \subset NPO$.

REOPTIMIZATION OF THE SET COVERING PROBLEM AFTER CHANGING ITS CONSTRAINT MATRIX

We consider the set covering problem (the problem $\Pi(A, c)$) in following statement:

$$\min\left\{ f(x) = \sum_{i=1}^{n} c_i x_i \,\middle|\, x \in Q(A) \right\}, \qquad Q(A) = \left\{ x \in B^n \,\middle|\, \sum_{j=1}^{n} a_{ij} x_j \ge 1,\ i = 1, \dots, m \right\},$$

where $B^n = \{0,1\}^n$, $A = \{a_{ij}\}, i = 1, \dots, m,\ j = 1, \dots, n,\ A - (m, n)$ is a Boolean matrix $(m \le n)$, and $c_i > 0,\ i = 1, \dots, n.$

We will use the following results.

It is well known [8] that a greedy algorithm solves the problem $\Pi(A,c)$ with the accuracy of estimate $\sum_{t=1}^{c(A)} \frac{1}{t} \le \ln m + 1$, where $c(A)$ is the maximal number of unities in a column of the matrix *A*, i.e., a greedy algorithm is some $(\ln m + 1)$-approximation algorithm for the solution of $\Pi(A,c)$.

We introduce the following denotations for changed instances of the problem $\Pi(A,c)$: $\Pi(A_j^+, c)$ (accordingly, $\Pi(A_j^-, c)$) is the problem $\Pi(A,c)$ with the replacement of 0 by 1 (1 by 0) at an arbitrary position in the *j*th column of the matrix *A*; $\Pi(A_j^+(p), c)$ (accordingly, $\Pi(A_j^-(p), c)$) is the problem $\Pi(A,c)$ with the replacement of an arbitrary number $p(1 \le p < m)$ of 0 by 1 (1 by 0) in the *j*th column of the matrix *A*; $REOPT(\cdot)$ is the corresponding reoptimization algorithm destined for optimal solution of the problem $\Pi(A,c)$.

We interpret a feasible solution $S = \{j_1, \ldots, j_k\}$ to the problem $\Pi(A,c)$ as a collection of columns $\{j_1, \ldots, j_k\}$ (subsets of the set $\{1, \ldots, m\}$) that form the covering of the matrix *A* (i.e., to $S = \{j_1, \ldots, j_k\}$ corresponds a feasible vector $x = (x_1, \ldots, x_n) \in Q(A)$ such that $x_{j_1} = x_{j_2} = \ldots = x_{j_k} = 1$; the other components are equal to 0). The weight $c(S)$ of the solution *S* is understood to be $c(S) = c_{j_1} + \ldots + c_{j_k}$.

THEOREM 1. There is $REOPT(\Pi(A_j^+, c))$, which is a $\left(2 - \frac{1}{\ln m + 1}\right)$-approximation algorithm.

Proof. We use the technique from [7] for proving the reoptimization of Min Vertex Cover (the minimum vertex cover problem) after inserting a new vertex. We construct $REOPT(\Pi(A_j^+, c))$.

Let S^* be an optimal solution to the problem $\Pi(A,c)$, let $S_{I'}^*$ be an optimal solution to $\Pi(A_j^+, c)$, and let $S^* \cup \{j\}$ be a feasible solution to $\Pi(A_j^+, c)$.

If $S_{I'}^*$ contains *j*, then $S^* \cup \{j\}$ is a solution.

Assume that $S_{I'}^*$ does not contain *j*; then $S^* \cup \{j\}$ is a feasible solution $\Pi(A_j^+, c)$ and since $c(S^*) \le c(S_{I'}^*)$, we have

$$c(S^* \cup \{j\}) \le c(S_{I'}^*) + c_j. \qquad (1)$$

To construct a feasible solution S_1 to the problem $\Pi(A_j^+, c)$, we

— apply a ρ-approximation algorithm to the problem $\Pi(A_j^+, c)$ with the eliminated *j*th column (set) (the corresponding $x_j = 0$ in $\Pi(A,c)$);

— add the column (set) *j* to the obtained solution.

Thus, we have

$$c(S_1) \le \rho(c(S_{I'}^*) - c_j) + c_j = \rho c(S_{I'}^*) - (\rho - 1)c_j \qquad (2)$$

(as above, $S_{I'}^*$ does not contain *j*). We choose the best solution (*S*) (with the least weight value) among solutions $S^* \cup \{j\}$ and S_1. Additionally multiplying inequality (1) by $\rho - 1$ and adding the result to inequality (2), we obtain

$$(\rho - 1)c(S^* \cup \{j\}) + c(S_1) \le (\rho - 1)c(S_{I'}^*) + \rho c(S_{I'}^*) = (2\rho - 1)c(S_{I'}^*).$$

Since $(\rho - 1)c(S^* \cup \{j\}) + c(S_1) \ge (\rho - 1 + 1)\min\{c(S^* \cup \{j\}), c(S_1)\} = \rho c(S)$, we have

$$\rho c(S) \le (2\rho - 1)c(S_{I'}^*).$$

We obtain an approximate solution S to the problem $\Pi(A_j^+, c)$ for which we have

$$c(S) \le \frac{2\rho-1}{\rho} c(S_{I'}^*) = \left(2 - \frac{1}{\rho}\right) c(S_{I'}^*).$$

Putting $\rho = \ln m + 1$ (i.e., applying a greedy algorithm in forming S_1), we obtain, as a result, the proof of the following theorem.

THEOREM 2. When $1 < p < m$, there is some $REOPT(\Pi(A_j^+(p), c))$ that is a $\left(2 - \frac{1}{\ln m + 1}\right)$-approximation algorithm.

The proof is completely similar to the proof of Theorem 1.

THEOREM 3. There is some $REOPT(\Pi(A_j^-, c))$ that is a $\left(2 - \frac{1}{\ln m + 1}\right)$-approximation algorithm.

After some modifications, we prove the theorem by analogy with the previous proof. We construct the corresponding $REOPT(\Pi(A_j^-, c))$.

Let S^* be an optimal solution to the problem $\Pi(A, c)$, and let $S_{I'}^*$ be an optimal solution to the problem $\Pi(A_j^-, c)$.

If S^* does not contain j, then we have $S_{I'}^* = S^*$. If S^* contains j, then S^* is not necessarily is a feasible solution to $\Pi(A_j^-, c)$. In this case, we construct a suitable approximation of the solution to the problem $\Pi(A_j^-, c)$.

For a fixed j, we put $i_0 = \arg\min\{c_i : a_{ij} = 1\}$ (i.e., some i_0, $1 \le i \le n$, such that c_{i_0} is minimal and $a_{ij} = 1$, $1 \le i \le n$). Then, in any case, $S^* \cup \{i_0\}$ is a feasible solution to $\Pi(A_j^-, c)$. Since $c(S^*) \le c(S_{I'}^*)$, we have

$$c(S^* \cup \{i_0\}) \le c(S_{I'}^*) + c_{i_0}. \tag{3}$$

A feasible solution S_1 to the problem $\Pi(A_j^-, c)$ is constructed as follows: we

— apply a ρ-approximation algorithm to the problem $\Pi(A_j^-, c)$ with the eliminated column (set) i_0 (the corresponding $x_{i_0} = 0$ in $\Pi(A, c)$);

— add the column (set) i_0 to the obtained solution.

Thus, we have

$$c(S_1) \le \rho(c(S_{I'}^*) - c_{i_0}) + c_{i_0} = \rho c(S_{I'}^*) - (\rho - 1)c_{i_0}. \tag{4}$$

Among solutions $S^* \cup \{i_0\}$ and S_1, we choose the best solution (S) (with the least weight value). Additionally multiplying inequality (3) by $\rho - 1$ and adding the result to equality (4), we obtain

$$(\rho-1)c(S^* \cup \{i_0\}) + c(S_1) \le (\rho-1)c(S_{I'}^*) + \rho c(S_{I'}^*) = (2\rho-1)c(S_{I'}^*).$$

Since

$$(\rho-1)c(S^* \cup \{i_0\}) + c(S_1) \ge (\rho - 1 + 1)\min\{c(S^* \cup \{i_0\}), c(S_1)\} = \rho c(S),$$

we have $\rho c(S) \le (2\rho - 1)c(S_{I'}^*)$.

We have obtained an approximate solution S to the problem $\Pi(A_j^-, c)$ and, for this solution, we have

$$c(S) \le \frac{2\rho-1}{\rho} c(S_{I'}^*) = \left(2 - \frac{1}{\rho}\right) c(S_{I'}^*).$$

Putting $\rho = \ln m + 1$ (i.e., using a greedy algorithm to form S_1), we obtain the proof of the theorem.

THEOREM 4. For $1 < p < m$, there is $REOPT(\Pi(A_j^-(p), c))$ that is some $\left(2 - \frac{1}{\ln m + 1}\right)$-approximation algorithm.

The proof is similar to the proof of Theorem 3.

REFERENCES

1. G. Ausiello, B. Escoffier, J. Monnot, and V. Th. Paschos, “Reoptimization of minimum and maximum traveling salesman’s tours,” in: Proc. SWAT 2006; LNCS, **4059**, 196–207, Springer, Berlin (2006).
2. H. J. Bockenhauer, L. Forlizzi, J. Hromkovic, et al., “On the approximability of TSP on local modifications of optimal solved instances,” Algorithmic Oper. Res., **2**(2), 83–93 (2007).
3. H. J. Bockenhauer, J. Hromkovic, T. Momke, and P. Widmayer, “On the hardness of reoptimization,” in: Proc. 34th Intern. Conf. on Current Trends in Theory and Practice of Computer Science (SOF-SEM 2008); LNCS, **4910**, 50–65, Springer, Berlin (2008)
4. B. Escoffier, M. Milanic, and V. Th. Paschos, “Simple and fast reoptimizations for the Steiner tree problem,” Algorithmic Oper. Res., **4**(2), 86–94 (2009).
5. C. Archetti, L. Bertazzi, and M. G. Speranza, “Reoptimizing the traveling salesman problem,” Networks, **42**(3), 154–159 (2003).
6. C. Archetti, L. Bertazzi, and M. G. Speranza, “Reoptimizing the 0-1 knapsack problem,” Manuscript (2008).
7. G. Ausiello, V. Bonifaci, and B. Escoffier, “Complexity and approximation in reoptimization,” in: Computability in Context: Computation and Logic in the Real World, Computability in Europe (CiE) Conference 2007 (June, 2007), Imperial College Press (2010), pp. 24–33.
8. V. A. Chvatal, “A greedy heuristic for the set covering problem,” Math. Oper. Res., **4**, No. 3, 233–235 (1979).