APPROXIMATION OF LINEAR COST SUPPLY MANAGEMENT PROBLEM WITH LOWER-BOUNDED DEMANDS

SATYAVEER S. CHAUHAN[1], **ANTON V. EREMEEV**[2], **ANNA A. ROMANOVA**[3], **VLADIMIR V. SERVAKH**[2]

[1] INRIA-SAGEP, UFR Scientifique, Université de Metz, Ile du Saulcy, 57045 Metz, FRANCE, Singh.Chauhan@loria.fr
[2] Omsk Branch of Sobolev Institute of Mathematics SB RAS, Pevtsov street 13, 644099 Omsk, RUSSIA, {eremeev,svv}@iitam.omsk.net.ru
[3] Department of Mathematics, Omsk State University, Mira prospect 55A, 644077 Omsk, RUSSIA, romanova_ann@bk.ru

Abstract. We consider the problem of optimization of product delivery to the manufacturing units, given lower and upper bounds on shipment sizes, lower-bounded demands and linear delivery costs. We suggest a fully polynomial time approximation scheme for this problem with single manufacturing unit, and show its hardness for greater numbers of manufacturing units.

Key Words. Supply management, approximation scheme, complexity.

1 Introduction

In this paper, we consider the possibility of finding the solutions with guaranteed approximation ratio to a supply management problem where a set of providers supply some product to a set of manufacturing units, the quantity of product that can be delivered lies between the given minimum and maximum values, and the costs proposed by each provider are linear functions of the quantity being delivered. Formally, the problem is stated as follows.

$$\sum_{i=1}^{n}\sum_{j=1}^{m} k_{ij}(x_{ij}) \to \min, \qquad (1)$$

$$\sum_{i=1}^{n} x_{ij} \geq A_j,\ j = 1, ..., m, \qquad (2)$$

$$\sum_{j=1}^{m} x_{ij} \leq M_i,\ i = 1, ..., n, \qquad (3)$$

$$x_{ij} \in \{0\} \cup [m_{ij}, M_i],\ i = 1, ..., n;\ j = 1, ..., m, \qquad (4)$$

where n is the number of suppliers; m is the number of manufacturing units; x_{ij} is the real-valued quantity of product delivered from provider i to manufacturing unit j; A_j is the minimal amount of product required for the manufacturing unit j; m_{ij} is the minimum quantity the provider i is prepared to deliver to the manufacturing unit j; M_i is the maximum quantity the provider i is able to deliver to any manufacturing unit and to all of them in total. The shipment cost is

$$k_{ij}(x_{ij}) = \begin{cases} 0, & \text{if } x_{ij} = 0; \\ a_{ij} + b_{ij}x_{ij}, & \text{if } x_{ij} > 0. \end{cases}$$

The parameters $a_{ij}, b_{ij}, m_{ij}, M_i, A_j$ are supposed to be integer and non-negative for all i, j. In what follows, the cost of solution $x = (x_{ij})$ given by (1) will be denoted by $f(x)$.

This problem is a simplified modification of the concave cost supply management problem considered in [1, 2] where instead of inequality (2) the equality was required and the cost functions were nonnegative and concave. Finding any feasible solution to that problem is NP-hard, yet there exists a pseudopolynomial-time exact algorithm [2]. As we will see below, the relaxed version considered here presents more possibilities for approximation and is more interesting in this respect, although the problem remains NP-hard with respect to finding exact solution (for it contains the MINIMUM KNAPSACK problem as a special case).

In fact, a simple reduction from the SUBSET SUM problem shows that finding a feasible solution satisfying (2)-(4) is NP-hard even in the case $m = 2$. Indeed, consider the recognition version of the SUBSET SUM problem that consists in deciding whether a set of positive integer numbers $\{p_1, p_2, ..., p_N\}$ is dividable into two subsets with equal sums. If $\sum_{i=1}^{N} p_i$ is odd, the answer is obviously negative. Otherwise, assign $m = 2,\ n = N,\ m_{i1} = m_{i2} = M_i = p_i,\ i = 1, ..., N$ and $A_1 = A_2 = \frac{1}{2}\sum_{i=1}^{N} p_i$. In this case the system (2)-(4) is solvable iff the required partition exists.

In what follows, by ρ-approximation algorithm we mean an algorithm that always returns a feasible solution whose cost at most ρ times exceeds the cost of the optimum (if the problem is solvable). Such solutions are called ρ-approximate solutions. By a fully polynomial time approximation scheme (FPTAS) we mean a family of $(1+\varepsilon)$-approximation algorithms over all factors $\varepsilon > 0$ with polynomial running times in problem input size and $1/\varepsilon$. The main results of the paper are formulated in the following two theorems.

Theorem 1 *In the case $m = 1$ there exists a fully polynomial time approximation scheme for problem (1)-(4) with time complexity $O(n^3\frac{1}{\varepsilon} + n^2)$.*

Theorem 2 *Problem (1)-(4) is NP-hard in the strong sense, and so is the problem of finding a $\mathrm{p}(n, m)$-approximate solution to it for any polynomial $\mathrm{p}(n, m)$ with positive coefficients.*

For the sake of notation simplicity let us write the problem (1)-(4) in the case $m = 1$ in a compact form:

$$\sum_{i=1}^{n} k_i(x_i) \to \min, \tag{5}$$

$$\sum_{i=1}^{n} x_i \geq A, \tag{6}$$

$$x_i \in \{0\} \cup [m_i, M_i],\ i = 1, 2, ..., n. \tag{7}$$

The shipment cost is

$$k_i(x_i) = \begin{cases} 0, & \text{if } x_i = 0; \\ a_i + b_i x_i, & \text{if } x_i > 0, \end{cases} \tag{8}$$

where $a_i \geq 0$ and $b_i \geq 0$, $i = 1, \ldots, n$.

In order to obtain an FPTAS for problem (5)-(8) in Sect. 4, we will first construct an efficient greedy 2-approximation algorithm in Sect. 2 and a pseudo-polynomial dynamic programming algorithm in Sect. 3. This approach is analogous to that for the minimum knapsack problem [4]. The last section contains the proof of Theorem 2.

2 2-Approximation Greedy algorithm

In this section, we propose a greedy algorithm for a generalization of problem (5)-(8), assuming that the manufacturing costs $k_i(x_i), i = 1, ..., n$ are given by

$$k_i(x_i) = \begin{cases} 0, & \text{if } x_i = 0; \\ a_i + g_i(x_i), & \text{if } x_i > 0, \end{cases} \tag{9}$$

where for all i we have $a_i \geq 0$ and $g_i(x_i)$ is a concave non-negative non-decreasing function when $x_i \in [0, M_i]$. By y we will denote the approximate solution constructed in the greedy algorithm; L is the amount of product to be necessarily added on subsequent iterations.

Algorithm Greedy.

1. Set $y_k := 0,\ k = 1, ..., n$.
2. Assume $L = A - \sum_{k=1}^{n} y_k$ and choose $j \in \{1, ..., n\}$ so that $y_j = 0$ and
$$\frac{k_j(\min\{\max\{L, m_j\}, M_j\})}{\min\{L, M_j\}} \tag{10}$$
$$\leq \frac{k_s(\min\{\max\{L, m_s\}, M_s\})}{\min\{L, M_s\}}$$
for all s such that $y_s = 0$.
3. Set $y_j := \min\{\max\{L, m_j\}, M_j\}$.
4. If $L - y_j > 0$ then go to Step 2, otherwise stop.

If after the last iteration we have $\sum_{k=1}^{n} y_k < A$, this means that no feasible solutions exist. The running time is obviously $O(n^2) \cdot T_g$, where T_g is an upper bound on time complexity of computing $g_i(x_i)$ for any i and x_i.

Theorem 3 *The Greedy algorithm is a 2-approximation algorithm for problem (5)-(7),(9).*

Proof. Note that on Step 3 variable y_j is assigned the value M_j in all iterations, except for the last one, when it equals either L or m_j (here and below by L we mean the value variable L takes when the Greedy algorithm is stopped). Without loss of generality we assume that the variables are ordered so that in the greedy solution $y_j = M_j$ for all $j \le i$, $y_{i+1} < M_{i+1}$ and $y_j = 0$ for all $j > i+1$.

Let f^* be the cost of optimal solution to (5)-(7),(9). Let us denote

$$h(x) = \sum_{j=1}^{n} \frac{(a_j + g_j(M_j))x_j}{M_j}$$

and consider the problem of minimization of $h(x)$ on the set defined by (6)-(7). Looking at the optimum $\hat{x}$ of the linear relaxation of this problem i.e. the problem where (7) is substituted by $0 \le x_j \le M_j,\ j = 1, 2, ..., n$, we conclude that

$$h(\hat{x}) \ge \sum_{j=1}^{i} a_j + g_j(M_j) = \sum_{j=1}^{i} k_j(M_j).$$

From the definition of $h(x)$, we have $h(x) \le f(x)$ for all x satisfying (6)-(7). So,

$$f^* \ge \sum_{j=1}^{i} k_j(M_j).$$

Therefore, if $y_{i+1} = 0$ (i.e. $L = M_i$ on the last iteration of the Greedy algorithm), then y is optimal.

Now we will see that if $y_{i+1} > 0$ then $f^* \ge k_{i+1}(y_{i+1})$. Indeed, suppose there exists a feasible solution z such that

$$\sum_{j=1}^{n} k_j(z_j) < k_{i+1}(y_{i+1}). \qquad (11)$$

Obviously, since $L = A - \sum\limits_{k=1}^{i} M_k$,

$$\sum_{s>i} z_s \ge A - \sum_{j \le i} z_j \ge L. \qquad (12)$$

Multiplying (11) by $L/k_{i+1}(y_{i+1})$ and removing the first i terms we obtain

$$\sum_{s>i} \frac{Lk_s(z_s)}{k_{i+1}(y_{i+1})} < L. \qquad (13)$$

The greedy rule (10) applied on the last iteration ensures that for all $s > i$

$$\frac{k_{i+1}(y_{i+1})}{L} = \frac{k_{i+1}(y_{i+1})}{\min\{L, M_{i+1}\}} \qquad (14)$$

$$\le \frac{k_s(\min\{\max\{L, m_s\}, M_s\})}{\min\{L, M_s\}}.$$

Thus, (13) yields

$$\sum_{s>i} \frac{\min\{L, M_s\} k_s(z_s)}{k_s(\min\{\max\{L, m_s\}, M_s\})} < L. \qquad (15)$$

Let us consider the terms of this sum with $z_s > 0$ (i.e. $z_s \ge m_s$). If $L < m_s$ then by (14) we have $k_s(m_s)/L \ge k_{i+1}(y_{i+1})/L$, which contradicts with inequality $k_s(z_s) < k_{i+1}(y_{i+1})$ implied by (11). Hence, the terms with $L < m_s$ are absent in (15).

If $m_s \le L \le M_s$ then the term in (15) turns into $Lk_s(z_s)/k_s(L)$ and we have two cases:

(a) $z_s \ge L$. Since $k_s(x)$ is a non-decreasing function so $k_s(z_s) \ge k_s(L)$. By (14) we have $k_s(L) \ge k_{i+1}(y_{i+1})$, thus $k_s(z_s) \ge k_{i+1}(y_{i+1})$, which contradicts with (11) again. Hence, the terms with $z_s \ge L$ are absent in (15) also.

(b) $z_s < L$. Using the concavity of k_s we obtain $Lk_s(z_s)/k_s(L) \ge z_s$.

Finally, if $L > M_s$, the terms in (15) turn into $M_s k_s(z_s)/k_s(M_s)$, and due to the concavity of k_s we have $M_s k_s(z_s)/k_s(M_s) \ge z_s$. Thus,

$$\sum_{s>i} z_s \le \sum_{s>i} \frac{\min\{L, M_s\} k_s(z_s)}{k_s(\min\{\max\{L, m_s\}, M_s\})}, \qquad (16)$$

and by (15) and (16) we have $\sum\limits_{s>i} z_s < L$, which contradicts with (12). So, $f^* \ge k_{i+1}(y_{i+1})$ and

$$2f^* \ge \sum_{j=1}^{i} k_j(M_j) + k_{i+1}(y_{i+1}).$$

$\square$

3 Pseudo-Polynomial Exact Algorithm

In this section, we will use a restricted version of the problem, where the formulation (5)-(8) is augmented with the following condition:

$$x_i \in \mathbf{Z},\ i = 1, ..., n. \qquad (17)$$

Here $\mathbf{Z}$ is the set of integers. Using Result 1 from [1] it is easy to obtain

Lemma 1 *A solvable problem (5)-(8) has an optimal solution $z \in \mathbf{Z}^n$ where $z_i \in \{0, m_i, M_i\}$ for all $i = 1, ..., n$, except for at most one coordinate.*

In view of Lemma 1, we reduce the set of feasible solutions to the set (6)-(7),(17). In the next paragraph for constructing the FPTAS it will be convenient to use a formulation of (5)-(8),(17) with the following change of variables:

$$\sum_{i=1}^{n}(\alpha_i u_i + \beta_i v_i) \to \min, \tag{18}$$

$$\sum_{i=1}^{n}(m_i u_i + (M_i - m_i)v_i) \geq A, \tag{19}$$

$$v_i \leq u_i,\ i = 1, 2, ..., n. \tag{20}$$

$$u_i \in \{0,1\}, v_i \in \left\{0, \frac{1}{M_i - m_i}, \frac{2}{M_i - m_i}, ..., 1\right\}, \tag{21}$$

$$i = 1, 2, ..., n.$$

Here the problem parameters are assumed integer and nonnegative. Note that now all values of the variables are bounded between 0 and 1 and the original problem (5)-(8),(17) transforms to the form (18)-(21) where

$$\alpha_i = a_i + b_i m_i,\ \beta_i = b_i(M_i - m_i),$$

$$x_i = u_i m_i + (M_i - m_i)v_i,$$

$i = 1, ..., n$. For brevity let us denote

$$\gamma(u, v) = \sum_{i=1}^{n}(\alpha_i u_i + \beta_i v_i)$$

and

$$w(u, v) = \sum_{i=1}^{n}(m_i u_i + (M_i - m_i)v_i).$$

We will also use a supplementary problem where (21) is substituted with

$$u_i \in \{0,1\}, v_i \in \{0,1\},\ i = 1, 2, ..., n, \tag{22}$$

i.e. the shipment sizes are allowed to take only the extreme values. Assume we have an a priori upper bound U on the cost of solutions to problem (18)-(20), (22). Let us define for each $k = 1, \ldots, n$ and $c = 0, \ldots, U$ a set

$$S(k, c) = \mathrm{argmax}_{u,v\in\{0,1\}^n}\{w(u, v) \mid \gamma(u, v) = c,$$

$$u_i = v_i = 0\ \forall\ i > k\}.$$

Now for any pair (k, c) if $S(k, c) \neq \emptyset$ then we will assume $\varphi(k, c) = w(u, v)$, where $(u, v) \in S(k, c)$. If $S(k, c) = \emptyset$ let $\varphi(k, c) = -\infty$. Then the initial stage of the dynamic programming is defined as follows

$$\varphi(1, c) = \begin{cases} 0, & \text{if } c = 0; \\ m_1, & \text{if } c = \alpha_1; \\ M_1, & \text{if } c = \alpha_1 + \beta_1; \\ -\infty, & \text{otherwise.} \end{cases}$$

The stages with $k = 2, ..., n$ are processed using the recursive equation

$$\varphi(k, c) =$$

$$\begin{cases} \varphi(k-1, c), & \text{if } c < \alpha_k; \\ \max\{\varphi(k-1, c), m_k + \varphi(k-1, c-\alpha_k)\}, & \\ & \text{if } c \geq \alpha_k,\ c < \alpha_k + \beta_k; \\ \max\{\varphi(k-1, c), m_k + \varphi(k-1, c-\alpha_k), & \\ \quad M_k + \varphi(k-1, c-\alpha_k-\beta_k)\}, & \\ & \text{if } c \geq \alpha_k + \beta_k. \end{cases}$$

Therefore, if γ' is the optimal goal function value for problem (18)-(20), (22) and $\gamma' \leq U$ then $\gamma' = \min\{c|\varphi(n, c) \geq A\}$ and the set $S(n, \gamma')$ contains the optimal solution (or solutions) to this problem.

As it follows from Lemma 1, solving problem (18)-(21) it suffices to search each coordinate v_j, $j = 1, ..., n$ within the set $\{0, \frac{1}{M_j - m_j}, \frac{2}{M_j - m_j}, ..., 1\}$ with $u_j = 1$, while the rest of the variables are optimized with the help of the dynamic programming algorithm described above.

Suppose we are given an upper bound $UB \geq \gamma^*$, where γ^* is the cost of optimal solution to (18)-(21). For this problem we suggest the following algorithm.

Algorithm A.
Set $\tilde{\gamma} = UB$.
For $j = 1$ to n do:

1. Apply the dynamic programming algorithm with $U = UB$ to problem (18)-(20), (22) where only m_j and M_j are substituted by 0.
2. Keep the values $\varphi_j(n, c)$, $c = 0, ..., UB$ and the representatives $(u^{(0)}, v^{(0)})$, ..., $(u^{(UB)}, v^{(UB)})$ of sets of solutions $S(n, 0), ..., S(n, UB)$. If for some c holds $S(n, c) = \emptyset$, then $(u^{(c)}, v^{(c)})$ is arbitrary.
3. For $c = 0$ to UB do:

    If $A \leq \varphi_j(n, c)$ and $c \leq \tilde{\gamma}$ then
    update $\tilde{\gamma} := c$, $\tilde{u} := u^{(c)}$, $\tilde{v} := v^{(c)}$.

    If $\varphi_j(n, c) < A < m_j + \varphi_j(n, c)$ and $c + \alpha_j \leq \tilde{\gamma}$ then
    update $\tilde{\gamma} := c + \alpha_j$; $\tilde{u}_i := u_i^{(c)}$, $\tilde{v}_i := v_i^{(c)}, i \neq j$; $\tilde{u}_j := 1$, $\tilde{v}_j := 0$.

    If $\varphi_j(n, c) + m_j \leq A \leq \varphi_j(n, c) + M_j$ and

    $$c + \alpha_j + \beta_j \frac{A - m_j - \varphi_j(n, c)}{M_j - m_j} \leq \tilde{\gamma}$$

    then update

    $$\tilde{\gamma} := c + \alpha_j + \beta_j \frac{A - m_j - \varphi_j(n, c)}{M_j - m_j};$$

$$\tilde{u_i} := u_i^{(c)}, \tilde{v_i} := v_i^{(c)}, i \neq j;$$
$$\tilde{u_j} := 1,\ \tilde{v_j} := \frac{A - m_j - \varphi_j(n,c)}{M_j - m_j}.$$

Return the optimal solution $(\tilde{u}, \tilde{v})$.

Theorem 4 *The algorithm* **A** *solves problem (18)-(21) in time* $O(n^2 UB)$.

Proof. Correctness of the algorithm follows by Lemma 1 and the properties of the dynamic programming scheme discussed above (note that the assumption $U = UB$ in the supplementary problem does not exclude the partial solution corresponding to the optimum). The time complexity bound is obvious. □

4 Fully Polynomial-Time Approximation Scheme

Now we can apply the "rounding-the-input" approach so that the complexity of the dynamic programming procedure is reduced to polynomial in input length.

Proof of Theorem 1. First of all, let us exclude the case $f^* = 0$, for it is easy to identify and to obtain the optimal vector. In what follows, we assume $f^* > 0$.

Taking the cost of greedy 2-approximate solution as an upper bound UB^* for the optimum to problem (5)-(8) we also get its lower bound $LB^* = UB^*/2 \leq f^*$ (note that $LB^* > 0$ since $0 < f^* \leq UB^*$).

Suppose that $0 < \varepsilon < 2$ is given (if $\varepsilon \geq 2$ the greedy solution is sufficient) and assume $\delta = \max\left\{\frac{\varepsilon LB^*}{2n}, 1\right\}$. Let us apply the algorithm **A** to the instance of problem (18)-(21) with

$$\alpha_i = \left\lceil \frac{a_i + b_i m_i}{\delta} \right\rceil, \beta_i = \left\lceil \frac{M_i - m_i}{\delta / b_i} \right\rceil, i = 1, \dots, n, \tag{23}$$
$$UB = \frac{UB^*}{\delta} + 2n.$$

The upper bound UB is valid because the cost of optimum in (18)-(21) with α_i, β_i defined in (23) is at most $f^*/\delta + 2n$. Now note that if $\frac{\varepsilon LB^*}{2n} > 1$ then $UB = 4n/\varepsilon + 2n$, and otherwise, we still have $UB \leq 4n/\varepsilon + 2n$. So by Theorem 4 the computation in **A** and Greedy will require $O(n^3 \frac{1}{\varepsilon} + n^2)$ operations.

We claim that the resulting solution

$$\tilde{x}_i = m_i \tilde{u}_i + (M_i - m_i)\tilde{v}_i,\ i = 1, ..., n$$

is a $(1+\varepsilon)$-approximate solution to (5)-(8). Indeed, if $\delta = 1$ then $\tilde{x}$ is an optimum.

If $\delta > 1$, let us denote $\gamma(x) = \gamma(u(x), v(x))$, where

$$u(x)_i = \text{ sign } x_i,\ v(x)_i = \frac{x_i - m_i u(x)_i}{M_i - m_i},$$

$i = 1, ..., n$, i.e. $\gamma(x)$ is the cost of solution corresponding to x in problem (18)-(21). Let x^* be optimal in (5)-(7),(8). Then

$$\frac{f(x^*)}{\delta} =$$
$$= \sum_{i=1}^{n} \frac{a_i + b_i m_i}{\delta} u(x^*)_i + \sum_{i=1}^{n} \frac{b_i(M_i - m_i)}{\delta} v(x^*)_i$$
$$\geq \sum_{i=1}^{n} \left\lceil \frac{a_i + b_i m_i}{\delta} \right\rceil u(x^*)_i$$
$$+ \sum_{i=1}^{n} \left\lceil \frac{b_i(M_i - m_i)}{\delta} \right\rceil v(x^*)_i - 2n = \gamma(x^*) - 2n.$$

Thus, since $(\tilde{u}, \tilde{v})$ is an optimum for (18)-(21), so $\delta\gamma(\tilde{x}) \leq \delta\gamma(x^*) \leq f^* + 2n\delta$, and

$$\frac{|f(\tilde{x}) - f^*|}{f^*} \leq \frac{\delta\gamma(\tilde{x}) - f^*}{f^*} \leq \frac{2n\delta}{LB^*} = \varepsilon.$$

□

5 Hardness of Approximation in the Case of Multiple Units

In this section we will see that in the general case problem (1)-(4) is NP-hard in the strong sense for approximation with any "reasonable" approximation factor. To this end, we will use the recognition version of the VERTEX COVER problem (RVCP): let $G = (V, E)$ be a graph with the set of vertices $V = \{1, ..., s\}$ and the edge set $E = \{e_1, ..., e_r\}$. A subset $C \subseteq V$ is called a vertex cover of G if every edge has at least one endpoint in C. The RVCP is given a graph G, decide whether there exist a cover of cardinality at most K in it. In what follows the degree of vertex j will be denoted by $\deg(j)$.

The reduction starts with identifying the vertices of G with manufacturing units and assigning edges to providers. Besides that let us introduce an additional provider number $r + 1$. Thus, we have

$n = r + 1$ and $m = s$. Let each provider $i = 1, ..., r$ have $M_i = 1$ and $m_{ij} = 1$, $a_{ij} = 0$ for all $j = 1, ..., s$, and assume that it charges $b_{ij} = 1$ for serving the two "privileged" manufacturing units with indices belonging to e_i. For the rest of the clients provider i $(i = 1, ..., r)$ will charge $b_{ij} = \Omega$, where $\Omega > r + s^2$ is a sufficiently large value. In other words, each provider $i = 1, ..., r$ is able to send one unit of product at low price just to one of its "privileged" customers. The provider $r + 1$ will have greater capacities: $m_{r+1,j} = s$ for all $j = 1, ..., s$, $M_{r+1} = Ks$ and it will always charge $x_{r+1,j}$ for its shipment, i.e. $b_{r+1,j} = 1, j = 1, ..., s$. The demands of the manufacturing units are set equal to the degrees of the corresponding vertices: $A_j = \deg(j)$ for all $j = 1, ..., s$.

Lemma 2 *a) If G has a cover of size K or less, then the optimal cost value in the corresponding supply management problem is at most $r + s^2$.*
b) If the constructed supply management problem has a feasible solution with cost less than Ω, then there exists a cover of size at most K in G.

Proof. (a) Given a cover C, let us set for all $j = 1, ..., s$ the shipment size $x_{r+1,j} = s$ iff $j \in C$ and $x_{r+1,j} = 0$ otherwise. Then a feasible solution to problem (1)-(4) is completed for all $i = 1, ..., r, j = 1, ..., s$ by setting $x_{i,j} = 1$ if $j \in e_i$ and $j \notin C$; otherwise $x_{i,j} = 0$. Now since all shipments of providers $i = 1, ..., r$ are directed to the "privileged" units, and provider $r + 1$ sends not more than s^2 units of product in total, so $f(r) \leq r + s^2$.

(b) Given a feasible solution x to (1)-(4), $f(x) \leq \Omega$ let us denote $C(x) = \{j : x_{r+1,j} > 0\}$. Here $|C(x)| \leq K$ since $x_{r+1,j} \geq s$ for all $j \in C(x)$, and $x_{r+1,1} + ... + x_{r+1,n} \leq M_{r+1} = Ks$. If $f(x) \leq \Omega$, then the first r providers supply only the "privileged" units. $C(x)$ must be a cover for if on some edge $e_i = (u, v)$ $x_{r+1,u} = 0$ and $x_{r+1,v} = 0$, then

$$\sum_{i=1}^{n} x_{iu} + \sum_{i=1}^{n} x_{iv} \leq \deg(u) + \deg(v) - 1 < A_u + A_v,$$

which is in contradiction with (2). $\square$

Proof of Theorem 2. Given an RVCP with graph G, assuming $\Omega = 1 + \mathrm{p}(n, m)(r + s^2)$, by the above transformation we construct a supply management instance with numeric parameters polynomially bounded in encoding length of instance of (1)-(4).

Let x^* denote an optimum to (1)-(4) and let x_T be a $\mathrm{p}(n, m)$-approximate solution to (1)-(4). On one hand, if a cover of cardinality at most K exists, then by Lemma 2 (a) we have $f(x^*) \leq r + s^2$ and $f(x_T) \leq \mathrm{p}(n, m)(r + s^2)$. On the other hand, if $f(x_T) \leq \mathrm{p}(n, m)(r + s^2)$ then $f(x^*) < \Omega$, and a cover of at most K vertices exists by Lemma 2 (b).

So by finding a $\mathrm{p}(n, m)$-approximate solution and answering positively if $f(x_T) < \Omega$ and negatively otherwise, one will solve the RVCP. $\square$

Acknowledgments

This research has been supported by the INTAS grants 00-217 and 03-51-5501 and RFH grant 04-02-00238a. The authors also would like to thank A.A. Kolokolov, A.V. Kononov and M.Y. Kovalyov for the helpful discussions.

References

1. Chauhan S.S., Proth J.-M.: The concave cost supply problem, *European Journal of Operational Research* 148. (2) (2003), 374–383.

2. Chauhan S.S., Eremeev A.V., Kolokolov A.A., Servakh V.V.: Concave cost supply management problem for single manufacturing unit. In: Supply chain optimisation. Ed. by A. Dolgui, J. Soldek, O. Zaikin. Kluwer. Acad. Pbs., 2004. 167–174.

3. Ibarra O., Kim C.E.: Fast approximation algorithms for the knapsack and sum of subset problems, *Journ. of the ACM* 22 (1975), 463–468.

4. Gens G.V., Levner E.V.: Computational complexity of approximation algorithms for combinatorial problems. Mathematical Foundations of Computer Science. Proc. 8th Symp., Olomouc, Czech., *Lecture Notes Comput. Sci.* 74. Springer, Berlin (1979), 292–300.