ACTA MATHEMATICA VIETNAMICA
Volume 19, Number 1, 1994, pp. 97–110

EFFICIENT METHODS FOR SOLVING CERTAIN BILINEAR PROGRAMMING PROBLEMS

LE DUNG MUU AND BUI THE TAM

Abstract. We specialize the algorithm in [7] for solving certain nonconvex programming problems which contain indefinite quadratic and linear complementarity problems as special cases. We show that indefinite quadratic problems with a few negative eigen values and rank k bilinear program with k is small can be solved efficiently by this algorithm. The method is also applied for solving linear complementarity problems without any assumption on the structure of the involved matrices.

1. Introduction

Due to the inherent difficulty of global optimization and the fact that general methods are efficient only for problems with a moderate numbers of variables, it is important to develope particular algorithms and to apply general methods for concrete problems taking into account their specific structures.

In this paper we consider the following global optimization problem (P)

$$\min\{f(x) := \sum_{i=1}^{k} f_i(x)g_i(x) + h(x) : x \in X\} \tag{P}$$

where X is a polyhedron in R^n given by a system of equalities and/or inequalities, $f_i, \quad g_i \quad (i = 1, \dots, k)$ are affine and h convex functions on X.

Several problems such as indefinite quadratic problems, bilinear programs, linear complementarity problems can be converted into the form of Problem (P). This problem with $k = 1$ and $h \equiv 0$ is often called a multiplicative

Received September 16, 1993.
This paper is supported by the National Basic Program in Natural Sciences and Applied Mathematics Program of NCSRV
Keywords: Indefinite quadratic problem, rank k bilinear program, complementarity problem, adaptive rectangular subdivision, decomposision, numerical efficiency.

program [5,8,9,12] which recently has been increasingly interested. For this case some efficient algorithms [5,8,9,12] are developed for obtaining a globally optimal solution. For solving rank two and rank three bilinear programming problems, which are special cases of Problems (P) with $k = 2$ and $k = 3$ respectively, recently Yajima and Konno [13] proposed a parametric method using one parameter for $k = 2$ and two parameters for $k = 3$. As it is reported in their paper [13] this method is quite efficient for $k \doteq 2$. For $k > 3$ their method, to our opinion, does not work efficiently, since it requires solving parametric linear programs with $(k-1)$ parameters in both the objective function and the constraints.

In this paper we show that Problem (P), due to its specific struture, could be efficiently solved by a method in [7] if k is small, say, less than 5. For linear complementarity problems a preliminary computational experiment shows that the method is efficient when k is much greater. The reason is that for complementarity problems one can check whether a given point is a solution or not.

2. Examples for the Problem

Below we give two examples which can be converted into the form of Problem (P).

2.1. Indefinite Quadratic Programming Problem.

It is well known that an indefinite quadratic form $d^T x + x^T C x$, by using only linear transformations, can be transformed into the form

$$q^T x + \sum_{i=1}^{k} \lambda_i x_i^2,$$

where k is the rank and λ_i are eigenvalues of the matrix C.

Let

$$I_+ := \{i : 1 \le i \le k, \lambda_i > 0\}$$

$$I_- := \{i : 1 \le i \le k, \lambda_i < 0\}$$

and take, for example,

$$f_i(x) = x_i,\ g_i(x) = \lambda_i x_i \quad (i \in I_-) \text{ and } h(x) = q^T x + \sum_{i \in I_+} \lambda_i x_i^2.$$

Then we see that an indefinite quadratic problem with linear constraints can be transformed into the form (P). From [10] we know that a linear constrained indefinite quadratic problem, even with one negative eigenvalue, is NP-hard.

It is well known that a bilinear programming problem [1,4] can be formulated as a quadratic program. A rank k bilinear programming problem in its canonical form, which has been considered in [13], is given by

$$\min\{f(u,v) := c_0^T u + d_0^T v + \sum_{i=1}^{k} c_i^T u\ d_i^T v\}$$

subject to

$$A_1 u \le b_1,\ A_2 v \le b_2,\ u \ge 0,\ v \ge 0$$

which is of the form (P) with $x = (u,v)$, $f_i(x) = c_i^T u$, $g_i(x) = d_i^T v$ and $h(x) = c_0^T u + d_0^T v$.

2.2. Linear Complementarity Problem.

Let $f, g : R^n \to R^k$ be two given affine mappings. The linear complementarity problem [6], denoted by (LP), is the problem of finding $x \in R^n$ such that

$$x \ge 0,\ f(x) \ge 0,\ g(x) \ge 0,\ \langle f(x), g(x)\rangle = 0. \tag{LP}$$

Let $f(x) = (f_1(x), \ldots, f_k(x))$, $g(x) = (g_1(x), \ldots, g_k(x))$. Then solving this complementarity problem is reduced to solving the following problem, denoted by (CP)

$$\min \sum_{i=1}^{k} f_i(x) g_i(x) \tag{CP}$$

subject to

$$x \ge 0,\ f_i(x) \ge 0,\ g_i(x) \ge 0,\ (i = 1, \ldots, k)$$

which is again of the form of Problem (P). Note that Problem (LP) is solvalable if and only if the optimal value of (CP) equals zero. It is clear that if $\bar{x}$ is a feasible point of (CP) and $\sum_{i=1}^{k} f_i(\bar{x}) g_i(\bar{x}) = 0$ then $\bar{x}$ is a solution of (LP). In what follows we shall refer to this program as a rank k linear complementarity problem.

3. Description of the Algorithm

This section we refine upon the decomposition branch and bound algorithm proposed in [7] for solving Problem (P). We show that for this problem one can perform the rectangular bisection in a k-dimensional space avoiding the vertex searching which is the main computational burden in the method proposed in [7]. The algorithm that we are going to describe is a form of the Prototype Branch and Bound Scheme given in [3,4] but here for the convergence an infinite sequence of nested rectangles does not necessarily tend to a singleton. A preliminary numerical experiment on a personal computer shows that the method is quite efficient for $k \leq 3$ and resonnably efficient with k is small, say, less than 5. To rank k linear complementarity problems the method could be used efficiently with much larger k.

Assume that X is bounded and that for each i the sign of f_i is unchanged in X, i.e., $f_i(x) \geq 0$ or $f_i(x) \leq 0$ for every $x \in X$.

We shall argree, as usually, that $0(+\infty) = 0$. Denote by f_* the optimal value of Problem (P). The algorithm then can be described as follows

ALGORITHM 1.

Choose a tolerance $\epsilon \geq 0$.

Initialization. For each $i = 1, \ldots, k$ calculate

$$\underline{\xi}_j := \min\{g_j(x) : x \in X\}, \overline{\xi}_j := \max\{g_j(x) : x \in X\}.$$

Let

$$R_0 := \{y = (y_1, \ldots, y_k) \in R^k : \underline{\xi}_j \leq y_j \leq \overline{\xi}_j \ (j = 1, \ldots, k)\}.$$

Define $\beta(R_0)$ as a lower bound of f over X with respect to R_0, i.e.,

$$\beta(R_0) \leq \min\{\sum_{k=1}^{k} f_i(x)g_i(x) + h(x) : x \in X, \ g(x) \in R_0\}.$$

Set $\mathcal{R}_0 := \{R_0\}$, $\beta_0 = \beta(R_0)$, α_0 an upper bound for f_* (we see that α_0 is obtained as $\beta(R_0)$ is calculated), and $x^0 \in X$ such that $f(x^0) = \alpha_0$. Take $i = 0$ and go to iteration i.

<u>*Iteration i*</u> $(i = 0, \ldots,)$

a) If $\alpha_i - \beta_i \leq \epsilon\alpha_i$, terminate: x^i is an ϵ- solution of (P).

b) If $\alpha_i - \beta_i > \epsilon\alpha_i$, then select $R_i \in \mathcal{R}_i$ such that

$$\beta_i := \beta(R_i) = \min\{\beta(R) : R \in \mathcal{R}_i\}.$$

Bisect R_i into two rectangles R_i^1 and R_i^2. Compute $\beta(R_i^1)$ and $\beta(R_i^2)$. As these numbers are computed we obtain new feasible points of Problem (P). Let x^{i+1} be the best feasible point among x^i and the newly generated feasible points.

Set $\alpha_{i+1} = f(x^{i+1})$ and

$$\mathcal{R}_i' := (\mathcal{R}_i \backslash \{R_i\}) \cup \{R_i^1, R_i^2\},$$

$$\mathcal{R}_{i+1} := \mathcal{R}_i' \backslash \{R \in \mathcal{R}_i' : \beta(R) > \alpha_{i+1}\}.$$

Let

$$\beta_{i+1} := \min\{\beta(R) : R \in \mathcal{R}_{i+1}\}.$$

Encrease i by 1 and go to iteration i.

To complete the description of this algorithm one has to give rules for computing lower bound $\beta(R)$ and for bisecting a rectangle into two subrectangles. The convergence and efficiency of the algorithm crucially depend on these bounding and branching operations. The bounding and branching operations that we are going to describe are essentially the same as those in [7], but here taking into account the specific structure of Problem (P) the vertex searching required in [7] is avoided.

3.1. Bounding Operation.

Let $R = \{y = (y_1, \ldots, y_k) \in R^k : a_i \leq y_i \leq b_i,\ i = 1, \ldots, k\}$ be a nonempty subrectangle of R_0. For each $i = 1, \ldots, k$ let

$$\delta_i^R := \begin{cases} a_i & \text{if} \quad f_i(x) \geq 0 \\ b_i & \text{if} \quad f_i(x) < 0 \end{cases} \tag{3.1}$$

(Note that by our assumption the sign of f_i is unchanged on X).

Define $\beta(R)$ as the optimal value of the following convex program, denoted by (RP),

$$\min\left(\sum_{i=1}^{k}\delta_i^R f_i(x)+h(x)\right), \tag{RP}$$

subject to

$$x\in X,\ g_i(x)=y_i,\ a_i\le y_i\le b_i\ (i=1,\dots,k).$$

As usually we let $\beta(R)=+\infty$ if this program has nonfeasible points.

LEMMA 3.1. *Let*

$$R=\{(y_1,\dots,y_k)\in R_0 : a_j^R\le y_j\le b_j^R\ (j=1,\dots,k)\}$$

where a_j^R and b_j^R $(j=1,\dots k)$ are given.

Then

$$\beta(R)\le\alpha(R):=\min\{\sum_{i=1}^{k}f_i(x)g_i(x)+h(x) : x\in X, g(x)\in R\}.$$

PROOF. Since the feasible domain of (RP) is compact, this problem has an optimal solution whenever its feasible set is nonempty.

For each $j=1,\dots,k$ let $y_j=g_j(x)$ and $y=(y_1,\dots,y_k)$. Then

$$\begin{aligned}
\alpha(R)=&\min_{(x,y)}\{\sum_{j=1}^{k}f_j(x)y_j+h(x) : x\in X,\ g_j(x)=y_j,\ y\in R\}\\
\ge&\min_{(x,y)}\{\sum_{j=1}^{k}f_j(x)y_j+h(x) : x\in X,\ g(x)=z,\ y,z\in R\}\\
=&\min_{x}\{\min_{y\in R}(\sum_{j=1}^{k}f_j(x)y_j+h(x)) : x\in X,\ g(x)=z,\ z\in R\}\\
=&\min_{x}\{\sum_{j=1}^{k}\min_{a_j^R\le y_j\le b_j^R}f_j(x)y_j+h(x)\} : x\in X,\ g(x)=z, z\in R\}\\
=&\min_{x}\{\sum_{j=1}^{k}\delta_j^R f_j(x) : x\in X,\ g(x)=z,\ z\in R\}=\beta(R).
\end{aligned}$$

If the feasible set of (RP) is empty, then $\beta(R)=\alpha(R)=+\infty$. $\square$

Lemma 3.1 allows us to calculate the lower bound $\beta(R)$ by solving convex program (RP) which is linear if h is affine. Let (x^R, y^R) be an optimal solution of (RP). Since the feasible set of Problem (RP) is contained in X, x^R is a feasible point of Problem (P). If $\delta_j^R = g_j(x^R)$ for every j, then it is clear that $\beta(R) = \alpha(R)$. In this case the rectangle R may be deleted from further consideration.

3.2 Branching Operation.

Assume that we are in iteration i. As we have remarked, if $\delta^{R_i} = g(x^{R_i})$ then

$$\beta(R_i) = \alpha(R_i) = \sum_{i=1}^{k} f_i(x^{R_i})g_i(x^{R_i}) + h(x^{R_i}).$$

This and the fact that $\beta(R_i)$ is a lower bound of f_* imply that x^{R_i} is a global optimal solution of (P). Thus if $\delta^{R_i} \neq g(x^{R_i})$ then it is nature to bisect the rectangle R_i into subrectangles in a way such that as the algorithm proceeds the distant between δ^{R_i} and $g(x^{R_i})$ tends to zero. Such a subdivision can be described briefly as follows.

Assume that we want to bisect a rectangle $R \subset R^k$ into two subrectangles. Let $u, v \in R$, $u \neq v$. Select an index $j_R \in \{1, \dots, k\}$ such that

$$|(u - v)_{j_R}| = \max_{1 \le j \le k} |(u - v)_j|.$$

Define

$$R_1 := \{y \in R : y_{j_R} \le (u_{j_R} + v_{j_R})/2\},$$

$$R_2 := \{y \in R : y_{j_R} \ge (u_{j_R} + v_{j_R})/2\}.$$

It is clear that R_1, R_2 are nonempty subrectangles of R. This branching operation has the following property which was proved in [7]. For the convenience we shall call u, v the division (or bisection) points of R.

Lemma 3.2. *Let $\{R_q\}$ be an infinite sequence of nested rectangles generated in the above branching operation. Let u^q and v^q be the division points of R_q. Then*

$$\lim_{q \to \infty} (u^q - v^q) = 0$$

provided this limit exists.

REMARK 3.1.

It is easy to give an example for which an infinite sequence of nested rectangles generated by the above bisection does not tend to a single point. Thus this branching operation is, in general, not necessarily exhaustive [4,11]. However it is sufficient to guarantee the convergence as the following theorem shows.

THEOREM 3.1. *Assume that the bounding and branching operations described above are used in Algorithm 1, then we have*

(a) If the algorithm terminates at iteration i, then

$$f(x^i) - f_* \le \epsilon f(x^i).$$

(b) If the algorithm is infinite, then $\beta_i \nearrow f_*$, $\alpha \searrow f_*$ as $i \to \infty$, and every limit point of the sequence $\{x^i\}$ solves (P).

PROOF. We observe that, since X is compact, the convex program (RP) has an optimal solution for every rectangle $R \subseteq R_0$, whenever its feasible domain is nonempty. Hence $\beta(R) > -\infty$ for every $R \subseteq R_0$, and therefore the branching operation is well defined.

(a) If the algorithm terminates at iteration i, then $\alpha_i - \beta_i \le \epsilon\alpha_i$. Since $\beta_i \le f_*$ we have $\alpha_i - f_* = f(x^i) - f_* \le \epsilon\alpha_i = \epsilon f(x^i)$.

(b) If the algorithm is infinite, then there exists an infinite sequence $\{R_q\}$ of nested rectangles, i.e., $R_{q+1} \subset R_q$ for every q. Since the division points $u^q := \delta^{R_q}, v^q := g(x^{R_q})$ belong to R_0, we may assume, taking subsequences if necessary, that $u^* = \lim u^q$, $v^* = \lim v^q$ exist. By viture of Lemma 3.2 we have $u^* = v^*$.

On the other hand

$$0 \le \alpha_q - \beta_q \le \sum_{j=1}^{k} f_j(x^{R_q})g_j(x^{R_q}) + h(x^{R_q}) - \sum_{j=1}^{k} u_j^q f_j(x^{R_q}) - h(x^{R_q})$$
$$= \sum_{j=1}^{k} \left(g_j(x^{R_q}) - u_j^q\right) f_j(x^{R_q}) = \sum_{j=1}^{k} (v_j^q - u_j^q) f_j(x^{R_q}).$$

Hence, letting $q \to \infty$ we obtain $\lim_{q\to\infty}(\alpha_q - \beta_q) = 0$. This and monotonicity of the sequences $\{\alpha_i\}$, $\{\beta_i\}$ imply $\lim_{i\to\infty} \alpha_i = \lim_{i\to\infty} \beta_i$. By the definitions $\beta_i \le f_* \le \alpha_i$ for every i, we have $\lim \alpha_i = \lim \beta_i = f_*$. Note that $\alpha_i = f(x^i)$ for every i, and therefore every limit point of the sequence $\{x^i\}$ solves Problem (P).

4. The Case of Linear Complementarity Problem

In this section we apply Algorithm 1 for solving linear complementarity problem (LP) stated in Section 1. As mentioned solving (LP) is reduced to solving the problem

$$\min \sum_{i=1}^{k} f_i(x) g_i(x),$$

subject to

$$x \ge 0,\ f_i(x) \ge 0,\ g_i(x) \ge 0\ (i = 1, \ldots, k).$$

Since our task now is to find only a feasible point $\overline{x}$ satisfying $\sum_{i=1}^{k} f_i(x) g_i(x) = 0$, rather than an optimal solution of this problem, we may take zero as both upper and lower bounds. Thus a rectangle R is deleted if the optimal value of the relaxed problem (RP) is greater than zero, and therefore only rectangles for which $\beta(R) = 0$ are still of interest. Algorithm 1 for this case reads as follows:

Algorithm 2

<u>*Initialization.*</u> For each $j = 1, \ldots, k$ take

$$\underline{\xi}_j := \min\{g_j(x) :\ x \ge 0,\ f_i(x) \ge 0,\ g_i(x) \ge 0,\ i = 1, \ldots, k\}$$

$$\overline{\xi}_j := \max\{g_j(x) :\ x \ge 0,\ f_i(x) \ge 0,\ g_i(x) \ge 0,\ i = 1, \ldots, k\}.$$

Let $\mathcal{R}_0 := \{y = (y_1, \ldots, y_k) \in R^k :\ \underline{\xi}_j \le y_j \le \overline{\xi}_j,\ j = 1, \ldots, k\}$. Set $i = 0$, $\mathcal{R}_0 := \{R_0\}$ and go to iteration i.

<u>*Iteration i*</u> $(i = 0, \ldots,)$

For each rectangle $R = \{y = (y_1, \ldots, y_k) : a_j^R \le y_j \le b_j^R,\ j = 1, \ldots, k\} \in \{\mathcal{R}_i\}$ solve the linear program (RL) given by

$$\min \sum_{j=1}^{k} a_j^R f_j(x),$$

subject to

$$x \geq 0,\ f_j(x) \geq 0,\ g_j(x) = y_j,\ a_j^R \leq y_j \leq b_j^R,\ j = 1, \ldots, k.$$

Let (x^R, y^R) denote the obtained solution of this program

a) If x^R satisfies $\sum_{j=1}^k f_j(x)g_j(x) = 0$, then terminate: x^R is a solution of the complementarity problem (LP).

b) Otherwise, delete from $\mathcal{R}_i$ all rectangles R for which $\beta(R) > 0$. Let $\mathcal{R}'_i$ be the set of the remainning rectangles.

b1) If $\{\mathcal{R}'_i\} = \emptyset$, then (LP) has no solution.

b2) If $\{\mathcal{R}'_i\} \neq \emptyset$, then choose a rectangle $R_i \in \mathcal{R}'_i$ such that

$$\max_{1\leq j\leq k} |(y^{R_i} - a^{R_i})_j| \geq \max_{1\leq j\leq k} |(y^R - a^R)_j|\ \forall R \in \mathcal{R}'_i.$$

Using y^{R_i} and a^{R_i} as division points and bisect the rectangle R_i into the two rectangles $R_{i1},\ R_{i2}$ by the above described rectangular bisection. Form

$$\mathcal{R}_{i+1} := (\mathcal{R}'_i \setminus \{R_i\}) \cup \{R_{i1}, R_{i2}\}.$$

Increase i by 1 and go to iteration i.

The validity of this algorithm is clear from the following remarks.

REMARKS.

4.1. If for some rectangle R we have that $y^R = a^R$ and that x^R is not a solution of (LP), then $\beta(R) > 0$, and there R is deleted.

4.2. In the algorithm a rectangle is eliminated from further consideration only if $\beta(R) > 0$. Thus if at some iteration i the case b1) occurs, i.e., $\mathcal{R}'_i = \emptyset$, then $\beta(R) > 0$ for every encountered rectangle which implies that (LP) has no solution.

4.3. This algorithm does not require any assumption on the structure of the involved functions f and g, except the one which ensures that $\overline{\xi}_j < +\infty$ for every j. Note that the linear program (RL) is always solvable, since its linear objective function is bounded from below by zero.

4.4. The algorithm could be applied when f_j is a concave function for each $j = 1, \ldots, k$. In this case (RL) is an ordinary convex program, since $a_j^R \geq 0$ for every $R \subset R_0$.

5. Preliminary Computational Experiments

In order to obtain a preliminary evaluation of the performance of the proposed algorithms we have written and tested a C computer code that implements the algorithms. To test the code we use it to solve twenty radomly generated problems on an ARC 286 personal computer.

Tables 1 and 2 contain respectively the computational results for rank k bilinear problems of the form

$$\min \sum_{i=1}^{k} \langle F^i, x\rangle \ \langle G^i, x\rangle$$

subject to

$$Ax \leq b, \ x \geq 0,$$

and for rank k linear complementarity problems of the form

$$x \in R^n, \ x \geq 0, \ Cx + c^0 \geq 0, \ Dx + d^0 \geq 0, \ \langle Cx + c^0, Dx + d^0\rangle = 0$$

where all of entries of the matricies A, C, D and the vectors b, c^0, d^0, F^i, G^i ($i = 1, ..., k$) of the test problems were radomly drawn from the integers in the intervals [- 10, 10]. The ordinary simplex method was used for computing lower bounds.

In Tables 1 and 2 we use the following headings:

- N: the number of variables,

- M: the number of constraints (without $x \geq 0$),

- K: the rank,

-ITE: the number of iterations,

-MNR: the number of the rectangles stored in the memory,

-TIME: the CPU time (in second).

For bilinear problems we take the tolerance $\epsilon = 10^{-2}$, for complementarity problems $\epsilon = 10^{-6}$.

The results in these tables show that Algorithm 1 could be used for solving rank k bilinear programming problem up to $k = 5$ on an ARC 286 personal computer, while Algorithm 2 is efficient for rank k linear complementarity problems with k is larger. The reason is that for complementarity problems one can check whether a given point is solution or not. We hope that on a fast computer the proposed algorithms could be used efficiently for larger k.

In the both tables it appears that the running time is much more sensitive to grow in the rank than to grow in the number of constraints or variables. The required memory however increases very slowly as the program runs, since a large percentage of the generated rectangles is eliminated from further consideration.

Table 1 (for *rank* k bilinear problems)

Prob.	M	N	K	ITE	MNR	TIME (second)
1	5	7	3	64	16	3.23
2	5	15	3	142	99	9.91
3	5	20	3	334	89	24
4	10	20	3	124	44	27.08
5	10	30	3	132	32	31.2
6	10	50	3	241	65	86.92
7	8	10	4	54	13	11.47
8	10	10	4	108	68	65.85
9	10	20	4	344	105	118
10	5	10	5	459	219	121
11	6	20	5	1000	229	140.6
12	10	10	5	518	247	171.0

Table 2 (for *rank k* linear complementarity problems)

Prob.	M	N	K	ITE	MNR	TIME (second)
1	5	15	3	4	3	0.96
2	10	50	3	12	9	6.2
3	8	10	4	3	6	1.6
4	15	20	5	9	4	12
5	6	20	10	2	4	6
6	10	50	10	5	4	12.1
7	10	50	50	12	9	19.7
8	20	50	30	9	6	23.1

References

[1] F.A Al-Khayyal, and J. Falk, *Jointly constrained biconvex programming*, Math. Oper. Res. 8 (1983), 273-386.

[2] C.A. Floudas and P.M. Pardalos, "A Collection of Test Problems for Constrained Global Optimization Algorithms," Springer-Verlag, Berlin, 1990..

[3] R. Horst, *Deterministic global optimization with partition set whose feasibility is not known. Application to concave minimization, reverse convex constraints, DC programming and Lipschitzian optimization*, Journal Optim. Theory Appl. 58 (1988), 11-37.

[4] R. Horst and H. Tuy, "Global Optimization: Deterministic Approaches," Springer-Verlag, Berlin, 1990.

[5] H. Konno, Y. Yajima and T. Matsui, *Parametric simplex algorithm for solving a special class of nonconvex minimization problems*, Journal of Global Optimization 1 (1991), 65-81.

[6] K.G. Murty, "Linear Complementarity Linear and Nonlinear Programming," Heldermann Verlag, Berlin, 1983..

[7] L.D. Muu and W. Oettli, *Method for minimizing a convex-concave function over a convex set*, Journal Optim. Theory Appl. 70 (1991), 377-384.

[8] L.D. Muu and B.T. Tam, *Minimizing the sum of a convex function and the product of two affine functions over a convex set*, Optimization (to appear)..

[9] P.M. Pardalos, *Polynomial time algorithms for some classes of constrained nonconvex quadratic problems*, Optimization 21 (1990), 843-853.

[10] P.M. Pardalos and S.A. Vavasis, *Quadratic programming with one negative eigenvalue is NP-hard*, Journal of Global Optimization 1 (1991), 15-22.

[11] N.V. Thoai and H. Tuy *Convergent algorithms for minimizing a concave function*, Math. Oper. Res. **5** (1980), 556-566.

[12] H. Tuy, *Polyhedral annexation, dualization and dimension reduction technique in global optimization*, Journal of Global Optimization **1** (1991), 229-244.

[13] Y. Yajima and H. Konno, *Efficient algorithms for solving rank two and rank three bilinear programming problems*, Journal of Global Optimization **1** (1991), 155-171.

INSTITUTE OF MATHEMATICS
P.O. BOX 631, BOHO, 10 000 HANOI, VIETNAM