Tạp chí Tin học và Điều khiển học, T. 14, S. 1 (1998) (34-39)

A MODIFICATION OF TUY'S ALGORITHM FOR CANONICAL DC PROGRAMMING PROBLEM

NGUYEN TRONG TOAN [1]

Asbtract: A version of outer - approximation method is presented for the Canonical DC optimization problem. Some computational experimens are described to compare it with other methods.

Keywods. Reverse Convex Programming Canonical DC Optimization. Outer Approximation. ε-Appoximate Feasible Solution.

1. INTRODUCTION.

In this paper, we are concerned with the Canonical DC opimization problem (CDC), also referred to as the reverse convex programming problem [3 - 7]:

Minimize f(x)

subject to:

$x \in D \setminus \text{int}G$,

Where $D = \{x: h(x) \le 0\}$ and $G = \{x: g(x) \ge)\}$; h(x) and g(x) are bounded convex functions in R^n; $f(x) = <c, x>$, $c, x \subset R^n$. Assume that D is bounded. It has been proved that any DC optimization problem can be reduced to the CDC (1).

CDC problem is a mathematical model for many practical applied problems. Besides, it plays an important role in the global optimization theory. Therefore, is has received much attention in recent years (see (1) and its references). The main difficulty for solving the problem is due to the presence of the reverse convex costraint $g(x) < 0$, which destroys the convexity and even the adjacency of the feasible set of the problem. Up to now, there were many different methods for solving CDC. However, several of them have not yet been interested sufficiently in their convergence, efficiency of computational test.

This paper includes 4 sections. After the introduction, the second section describes a typical outer-approximation algorithm for CDC, which presented by H. Tuy (see [1]). The thrid one presents our modification of Tuy's algorithm and its theoretical background. The last one presents some computational experiences of the algorithms.

2. TUY'S OUTER- APPROXIMATION ALGORITHM

To slove the problem (1), it often takes us a very great amount of calculation. Besides, to meet the application necessary of the problem we can be completely satisfied with an approximate optimal solution as follow:

[1] Department of Applied Mathematics Hanoi University of Technology

Definition: Given a sufficiently small positive number ε, vector $x_\varepsilon \in R^n$ is called ε-feasible solution of CDC if:

$$h(x_\varepsilon) \leq 0,\ g(x_\varepsilon) \leq 0.$$

And it is called ε-approximate optimal solution if:

$$h(x_\varepsilon) \leq 0,\ g(x_\varepsilon) \leq 0,\ f(x_\varepsilon) - f^* \leq 0;$$

where f^* is the optimal value of CDC.

It is clearly, when $\varepsilon_k \downarrow 0$ all cluster points of sequence $(x_{\varepsilon k})$ (ε_k-approximate optimal solutions of CDC) are exact optimal solutions of CDC. Furthermore, if an optimal solution w of the convex program Min $\{f(x): x \in D\}$ satisfies inequality $g(w) \leq 0$, then it must be an optimal solution for CDC (1) as well. Therefore, the condition $g(w) > 0$ is always assumed. By translating the origine if necessary, we can always suppose that:

$$0 \in \text{int } D \cap \text{int } G \quad (2)$$

ALGORITHM 1 (see [1])

Initialization.

Let $\gamma = \langle c, x^{*1} \rangle$, where x^{*1} is the current best solution (if there is on such solution then let $x^{*1} = 0$ and $\gamma = +\infty$). Let k = 1.

Build a polytope P_1 and its vertex set V_1, such that:

$$\{x \in D: \langle c, x \rangle \leq \gamma^1 - \varepsilon \subset P_1 \subset \{x: \langle c, x \rangle \leq \gamma^1 - \varepsilon\}.$$

Step k = 1, 2...

- Compute $x^k \in \arg\min \{g(x): x \in V_k\}$. If $g(x^k) > 0$ then terminate.

a. If $\gamma^k < +\infty$ x^{*k} is an ε-approximate optimal solution of CDC.

b. If $\gamma^k = +\infty$ then the problem has no feasible solution.

- Select $w^k \in V_k$ that $\langle c, w^k \rangle \leq \min \{\langle c, x \langle : x \in V_k + \varepsilon$. If $h(w^k) \leq \varepsilon$ and $g(w^k) \leq 0$ then terminate: w^k is an ε-approximate optimal solution of CDC.

- If $h(u^k) \geq \varepsilon/2$ then

a. Let $x^{*k+1} = x^{*k}$, $x^{k+1} = \gamma^k$;

b. Let $p^k \in \partial h(w^k)$ (such a p^k exists because h(.) is convex, so $\partial h(w^k) \neq \varnothing$).

$$l_k(x) = \langle p^k, x - w^k \rangle + h(w^k) \quad (3)$$

c. Compute the vertex set V_{k+1} of the polytope $P_{k+1} = P_k \cap \{x: l_k(x) \leq 0\}$.

d. Go to step k + 1.

- Select $\gamma^k \in |w^k, x^k|$ so that $g(\gamma^k) = \varepsilon$ (γ^k exists since $g(x^k) \leq 0$ and $g(w^k) > 0$). If $h(\gamma^k) > \varepsilon$ then:

a. Let $x^{*k+1} = x^{*k}$, $\gamma^{k+1} = \gamma^k$;

b. Select $u^k \in |w^k, x^k|$ such that $h(u^k) = \varepsilon$ (γ^k exists since $h(w^k) \leq \varepsilon/2$ and $h(\gamma^k > \varepsilon)$:

Let $p^k \in \partial h(u^k)$, and:

$$l_k(x) = \langle p^k, x - u^k) \quad (4)$$

c. Compute the vertex V_{k+1} of the polytope $P_{k+1} = P_k \cap \{x: l_k(x) \leq 0\}$;

d. Go to step k + 1.

- If $h({}_{\gamma}^{k}) \leq \varepsilon$ then let $x^{*k+1} = x^{*k}, {}_{\gamma}^{k+1} = \langle c, {}_{\gamma}^{k}\rangle$.

a. If $\langle c, w^k - {}_{\gamma}^{k} \rangle \leq 0$ then terminate: x^{*k+1} is an ε-approximate optimal solution of CDC;

b. Otherwise, let

$$l_k(x) = \langle c, x - {}_{\gamma}^{k}\rangle + \varepsilon \quad (5)$$

c. Compute the vertex set V_{k+1} of the polytope $P_{k+1} = P_k \cap \{x: l_k(x) \leq 0)$

d. Go to step k + 1.

The finiteness of the algorithm is guaranteed by the following theorem:

Theorem 1 ([1]). The algorithm 1 terminates after a finitely many steps by an ε-approximate optimal solution or by the evidence that the problem has no feasible solution.

Remark

- Algorithm 1 uses a large number of cut-hyperplanes of different types in the solution process. Therefore, the total number of vertices P_k may quickly become quite large, and it makes increasing the computational cost and amount of memory for its storage. It causes a certain difficulty in using this algorithm.

- Algorithm 1 pays a great attention in solving be convex programming problem min $\{\langle c, x\rangle: x \in D\}$. Only if it had found such w^k that $h(w^k) < \varepsilon/2$ (it means w^k is the ε-approximate optimal solution of the above convex program) and $g(w^k) > 0$, then ${}_{\gamma}^{k}$ or u^k in turn are calculated and their cuts are built. That maybe unreasonable if the solution w did not satisfy the reverse - convex constraint.

3. THE MODIFICATION ALGORITHM

The following algorithm is a modification of the above algorithm. In order to prevent the number of verties of the approximate polytopes P_k from increasing, too quickly, in each step, after finding x^k and w^k as the above algorithm we solve equation $g(x) = 0$ on $|w^k, x^k|$ to find vector u^k (or let $u^k = w^k$ if $g(w^k) \leq 0$) and to cut it from P_{k+1} if $h(u^k) > \varepsilon$. The selection of such u^k bases on the following property of CDC.

Theorem 2. ([1]). If convex programming problem Min $\{f(x): x \in D\}$ has a optimal solution w, satisfies $g(w) > 0$, and the CDC problem (1) has a feasible solution, then there exists such an optimal solution x^* of CDC that:

$$g(x^*) = 0 \quad (6)$$

Furthermore, since h(x) is convex, so $h(u^k) \leq \max \{h(w^k), h(x^k)\}$; and if $h(u^k) > \varepsilon$, then either x^k or w^k is cut from P_{k+1} as u^k. That is also the reason we attemps to find an ε-approximate optimal solution satisfies the equality (6).

ALGORITHM 2

Initialization.

Build a polytope $P_1 \supset D$ and its vertex set V_1. Select $\varepsilon > 0$.

Let $w^1 = \arg\min \{<c, x>: x \in V_1\}$,

$x^1 = \arg\min \{g(x): x \in V_1\}$,

$\gamma^1 \geq \max \{<c, x>: x \in D\} + \varepsilon$

Step k = 1, 2...

If $g(x^k) > 0$ then terminate

If $g(w^k) \leq 0$ let $u^k = w^k$; otherwise compute $u^k \in |w^k, x^k|$ such that $g(u^k) = 0$ (u^k exists because $g(w^k) > 0$ and $g(x^k) \leq 0$). There are two cases:

a. If $h(u^k) \leq \varepsilon$, let $\gamma^{k+1} = <c, u^k>$, $x^* = u^k$, $P_{k+1} = w^k$. $x^{k+1} = \arg\min \{g(x): x \in P_{k+1}, <c, x> \leq \gamma^{k+1} - \varepsilon\}$ (7)

and go to step k + 1.

b. If $h(u^k)$ ε, let $p_k \in \partial h(u^k)$ (since h(.) is convex, $\partial h(u^k) \neq \emptyset$),

$\gamma^{k+1} = \gamma^k$,

$l_k(x) = <p^k, x - u^k> + h(u^k)$ (8)

Compute the vertex set V_{k+1} of the polytope

$$P_{k+1} = P_k \cap \{x: l_k(x) \leq 0\}$$

If $l_k(x^k) \leq 0$ then let $x^{k+1} = x^k$, otherwise compute:

$$x^{k+1} = \arg\min \{g(x): x \in P_{k+1}, <c, x> \leq \gamma^{k+1} - \varepsilon\} \quad (9)$$

If $l_k(w^k) \leq 0$ then let $w^{k+1} = w^k$, otherwise compute:

$$w^{k+1} = \arg\min \{<c, x>: x \in V_{k+1}\}.$$

Then go to step k + 1.

Theorem 3. The algorithm 3 terminates after a finite number of step and yields either ε-approximate aptimal solution or an evidence that the problems has no feasible solution.

Proof. Suppose that the algorithm is infinite. Clearly, $P_1 \supset P_2 \supset .... \supset P_k .. \supset D$.

From (7) and (9), it implies $<c, w^k> \leq \gamma^{k+1} = <c, u^k> \leq <c, x^k> \leq \gamma^k - \varepsilon$. It is easy to see that the case a never occurs more than $|(\gamma^1 - <x, w^1>)||\varepsilon| + 1$ time. So that, the case b must occur infiniely many times. Because P_1 is bounded, there exists a convergent subsequence of the sequence $\{u^k\}$. It means that there exist two sufficiently large number k and s ($s \geq k + 1$) such that $<pk, u^s - u^k> > -\varepsilon$. But $h(u^k)$. ε, it confliets with (8):

$$u^s \in P_{k+1}, l_k(u^s) = <p^k, u^s - u^k> + h(u^k) \leq 0.$$

By the above contradiction, it is evident that the algorithm must be finite.

Suppose now that the algorithm terminates at step k. Since (7) and (9), P_k has no such feasible solution x that $g(x) \le 0$ and $<c, x> \le \gamma^k - \varepsilon$. If the case a has ever occurred then the recent x^* is an ε-approximate optimal solution. Otherwise, it means that $\gamma^k = \gamma^1$; since $P_k \supset D$, it shows that the problem has no feasible solution.

The theorem has been completely proved.

4. COMPUTATIONAL EXPERIENCE

The algorithms were coded in PASCAL and run on a personal computer AT 386DX to test 12 different problems. The result is described in the following table:

	Dimension			Algorithm 1						Algorithm 2					
	N	M1	M2	STEP	VER 1	VER 2	CUT 1	CUT 2	TIME	STEP	VER 1	VER 2	CUT 1	CUT 2	TIME
1	2	3	0	2	3	5	1	0	0	2	3	5	1	2	0
2	8	6	0	4	45	69	1	2	50	3	24	27	1	3	16
3	2	4	0	4	3	5	0	1	6	1	3	3	0	1	5
4	2	5	0	2	3	5	0	1	22	1	3	3	0	1	17
5	3	8	0	9	12	32	5	3	11	6	7	19	4	6	5
6	2	5	0	13	9	29	4	8	11	11	5	9	3	11	5
7	2	1	2	15	7	31	12	2	22	11	6	15	6	9	11
8	2	4	0	9	5	19	2	6	11	8	4	7	2	8	6
9	3	1	2	64	123	380	51	12	9079	28	24	52	13	25	160
10	3	1	2	54	99	282	48	5	3877	27	40	85	20	25	396
11	3	3	1	3	4	10	1	1	5	2	4	7	1	2	5
12	5	6	1	28	20	301	8	19	401	6	22	55	5	5	50

Table: Computational result of the algorithm 1 & 2

Where:

- **N**: Number of variables;
- **M1**: Number of linear constrains, sign constrains not include;
- **M2**: Number of convex constrains;
- **STEP**: Number of iterations;
- **VER 1:** Maximal number of vertices of polytope P_k;
- **VER 2:** Surn of generated vertices;
- **CUT 1**: Number of cuts (3) and (4);
- **CUT 2:** Number of cuts by levels of the function f(x) (in (5), (7), and (9)).
- **TIME**: CPU time in % of second; I/O time not includes.

From the above table we note that: because the cuts by levels of the objectie function f(x) (CUT 2) are not used to make new polytopes algorithms 2, this leads to lower VER 1 and TIME. In the case wher f(x) is convex, the problem may be formulated as: Minimize t, $h_l(x) = f(x) \leq 1$ and the old constraints. Two among the tested problems cited in the above table are of this form:

Problem 9. $$f(x) = (x_1 - 3.69)^2 + (x_2 - 12)^2 \to \min$$

$$x_1 + x_2 \leq 30,\ x_1 \geq 0,\ x_2 \geq 0$$

$$h(x) = -x_1 + 18x^2_2/484 - 10 \leq 0$$

$$g(x) = -x^2_1 - x^2_2 + 484 \leq 0$$

The optimal solution $x^* = (6.48079, 21.02378)$, $f^* = 89.21686$ with $\varepsilon = 0.001$.

Problem 10. $$f(x) = (x_1 - 2)^2 + (x_2 - 1)^2 \to \min$$

$$x_1 + x_2 \leq 5,\ x_1 \geq 0, x_2 \geq 0$$

$$h(x) = x^2_1 - 4.4.x^1 + x^2_2 - 2.4x_2 + 4.03 \leq 0$$

$$g(x) = 4x_1 - x^2_1 - 0.36x^2_2 - 2.56 \leq 0$$

The optimal solution $x^* = (2.77534, 1.526646)$, $f^* = 0,87743$ with $\varepsilon = 0.0001$.

REFERENCES

1. **H. Tuy**. *Canonical DC Programming Problem: Outer Approximation Methods Revisited.* Operation Research Letters 18 (1995) p. 99 - 106.

2. **H. Tuy**. *Convex Program with An Additional Reverse Convex Constraint.* J. Optim Theory Appl. 52 (1987) p. 463 - 485.

3. **L. D. Muu.** *A Convergent Algorithm for Solving Linear Programming with An Additional Convex Constraint*. Kybernetika. 21 (1985). p. 438 - 435.

4. **N. D. Nghia. N. D. Hieu.** *A Method for Solving Reverse Convex Programming Problems.* Acta Math. Vietnamica. 2 (1986) p. 241 - 252.

5. **N. D. Nghia, N. D. Hieu.** *Computational Testing Procedure of Tuy's Method for Soling Reverse Convex Programming Problems.* J. Math. 2 (1987), (in Vietnamese).

6. **N. D. Nghia, N. D. Hieu.** *An Algorithm for Linear Programs with A Reverse Convex Constraint.* Volume of Math. Res. Works, Hanoi University of Technology, 1984 (in Vietnamese).

7. **N. T. Toan, N. D. Nghia.** *Testing Compution, Comparison and Modification of Several Algorithms for Canonical Reverse Convex Programming Problem.* Proc of 1st National Conf. On Optimization and Control, Quinhon, 1996 (in Vietnamese).

8. **R. Horst and H. Tuy.** *Global Optimization (deterministic approaches).* (1st ed. 1990) 2nd ed. Springer, Gerlin, 1993.

Địa chỉ liên hệ:

1. Nguyễn Trọng Toàn, Khoa Cơ Bản Học Viện Không quân

54c Đường Trường Chinh Tel: 8.535 143

2. Nguyễn Đức Nghĩa, Khoa Công Nghệ Thông Tin ĐHBK Hà nội.

Tel (Home) 8.520 674