One algorithm for branch and bound method for solving concave optimization problem

A A Andrianova[1], **A A Korepanova**[2] **and I F Halilova**[2]

[1]Department of System Analysis and Information Technologies, Kazan (Volga Region) Federal University, Kazan, Kremlevskaya st. 18, Russia
[2]Master program Fundamental Informatics and Information Technologies, Kazan (Volga Region) Federal University, Kazan, Kremlevskaya st. 18, Russia

E-mail: Anastasiya.Andrianova@kpfu.ru

Abstract. The article describes the algorithm for branch and bound method for solving the concave programming problem, which is based on the idea of similarity the necessary and sufficient conditions of optimum for the original problem and for a convex programming problem with another feasible set and reverse the sign of the objective function. To find the feasible set of the equivalent convex programming problem we construct an algorithm using the idea of the branch and bound method. We formulate various branching techniques and discusses the construction of the lower objective function evaluations for the node of the decision tree. The article discusses the results of experiments of this algorithm for some famous test problems of a particular form.

1. Introduction

The concave programming problem has many applications in various fields of human activity (medicine, biology, technology, economics). Very often a model of concave programming problem encountered in the decision theory, concave programming problems applied for solving equilibrium problems, complementarity problems and others. Thus, it can be stated a serious interest in the construction an effective methods [1-4].

The main methods for the concave programming problem are based on the ideas of the cutting-plane methods, element methods and techniques of branches and borders based on the decomposition of the admissible set. However, most of these methods do not provide a solution in a reasonable time for practical problems [1,4].

In this paper we propose a different approach of branch and border method. The idea of this approach is based on the theoretical result - the similarity of the necessary and sufficient conditions for optimum some convex programming problem and original concave programming problem. Thus, the branch and bound method is only necessary in order to build this equivalent auxiliary convex programming problem. This theoretical result for the concave programming problem with linear constraints has been proposed and substantiated in [5,6].

So, we suggest a generalization of this theoretical result to the case of the feasible set defined by a system of inequalities with concave functions, formulate a general scheme of branch and bound method for solving the concave programming problem, an one algorithm for the special form of a concave programming problem and shows the results of experiments on a series of well-known test problems [7].

2. Theoretical aspects

Suppose that in Euclidean space R_n we consider the problem

$$\min_{\boldsymbol{x}\in D} \to f(\boldsymbol{x}), \tag{1}$$

where $D=\{\boldsymbol{x}\in R_n \mid f_i(\boldsymbol{x})\le 0, i\in I\}$, $I=\{1,2...m\}$, $f(\cdot)$, $f_i(\cdot)$ $i\in I$ are smooth concave functions. As known, this problem can be several local minima. The global minimum is among them.

We formulate the necessary conditions for a local minimum the problem (1) known as Lagrange principle (for example, [8]). Each point which satisfies the necessary conditions for a local minimum is called a stationary point.

Theorem 1. *If the point $\boldsymbol{x}^*\in D$ is a local minimum for problem (1), then there are exists numbers $y_i^*\ge 0$, $i=0,1...m$, such that:*

$$y_0^* f(\boldsymbol{x})+\sum_{i=1}^{m} y_i^* f_i(\boldsymbol{x}^*)=\boldsymbol{0}, \tag{2}$$

$$y_i^* f_i(\boldsymbol{x}^*)=0, \quad i=1...m. \tag{3}$$

It is obvious that the conditions (2) - (3) are equivalent to the following condition

$$y_0^* f(\boldsymbol{x})+\sum_{i\in I(x^*)} y_i^* f_i(\boldsymbol{x}^*)=\boldsymbol{0}, \tag{4}$$

where $I(\boldsymbol{x}^*)=\{\mathrm{i}\in \mathrm{I} \mid f_i(\boldsymbol{x}^*)=0\}$ is the active indices set in point $\boldsymbol{x}^*$. Note that finding the number y_0^* can be neglected, if the problem (1) satisfies the regularity conditions: gradients of functions in stationary point $\left\{ f_i'(\boldsymbol{x}^*)\right\}_{i=1}^{m}$ is a linearly independent system..

Choose a subset of indices $I_0\subset I$. Define an auxiliary convex programming problem

$$\max_{\boldsymbol{x}\in D_0} \to f(\boldsymbol{x}), \tag{5}$$

where $D_0=\{\boldsymbol{x}\in R_n \mid f_i(\boldsymbol{x})\ge 0, i\in I_0\}$.

Theorem 2. *The solution of problem (1) is the solution of problem (5) with $I_0=I(\boldsymbol{x}^*)$.*

Prof. The problem (5) is a convex programming problem, therefore, any of its local minimum points is a global minimum point. For convex programming problem stationary point conditions are also sufficient conditions for the global optimum ([8]):

$$y_0^* f(\boldsymbol{x}^*)+\sum_{i\in I_0} y_i^* f_i(\boldsymbol{x}^*)=\boldsymbol{0}, \tag{6}$$

$$y_i^* f_i(\boldsymbol{x}^*)=0, \quad i\in I_0. \tag{7}$$

So, for $I_0=I(\boldsymbol{x}^*)$ conditions (2)-(3) and (6)-(7) are the same. It is obvious that any local minimum point of problem (1) is a solution some problem of form (5). Conversely, if the point $\boldsymbol{x}^*$ solves the problem (5) and $\boldsymbol{x}^*\in D$, then $\boldsymbol{x}^*$ is the stationary point of the problem (1).

Consequently, the solution of the problem (1) can be found as a point of local minimum obtained as the solution of the problem (5) for some index set, with a minimum value of the objective function. Therefore, it can be found by trying all possible subsets of indices and solution for their auxiliary problem (5).

It should be noted that the problem (5) for some subset $I_0 \subset I$ may have empty feasible set. Consequently, the method of solving the auxiliary problem should have a way to determine this fact.

An effective way to solve the auxiliary problem (5) can be a solution of its dual problem:

$$\min_{y \in Y} \to \varphi(\boldsymbol{y}), \tag{8}$$

where objective function defines as $\varphi(\boldsymbol{y}) = \sup_{x \in R_n} L(\boldsymbol{x}, \boldsymbol{y})$ with $L(\boldsymbol{x}, \boldsymbol{y}) = y_0 f(\boldsymbol{x}) + \sum_{i \in I_0} y_i f_i(\boldsymbol{x})$ and feasible set has the form $Y = \{\boldsymbol{y} \in R_{m+1} \mid y_i \geq 0, i \in I_0, y_i = 0, i \notin I_0\}$. A simple form of this set may allow the use of less sophisticated computational methods for solving the problem (8). In addition, by the solving of the dual problem (8) we can conclude about the incompatibility of the restrictions of problem (5): if objective function of problem (8) is unbounded from bottom in its feasible set, then the problem (5) has no solutions.

3. The general scheme of branch and bound algorithm

So, as noted in the previous section, the solution of problem (1) can be found by searching all possible subsets $I_0 \subseteq I$. Thus, it is possible to organize this search with using the branch and bound method is constructed as follows.

General scheme. We denote $I_0 = \varnothing$, $f_{rec} = +\infty$. In each node of the decision tree following actions are performed:

1. The convex programming problem (5) is solved (for example, using the dual problem (8)). Note this solution as $\boldsymbol{x}'$. If found that the feasible set of the problem (5) is empty, the branching is not performed and returns to the parent node of the decision tree. Note that in the root node of the decition tree point $\boldsymbol{x}'$ will be the point of the absolute maximum of the function $f(\cdot)$ if such maximum exists.
2. Calculate the lower bound evaluation f'_{low} of objective function for node and all its child subtree.
3. If $f'_{low} > f_{rec}$, the branching is not performed and returns to the parent node of the decision tree.
4. If $\boldsymbol{x}' \in D$, then $\boldsymbol{x}'$ is the stationary point pf the problem (1). If $f_{rec} > f(\boldsymbol{x}')$, then $f_{rec} = f(\boldsymbol{x}')$ and we memorize a point $\boldsymbol{x}'$ as a possible solution of problem (1).
5. We perform the branching. Choosing indexes are not yet included in the feasible set $J = \{i \mid i \in I, i \notin I_0\}$. For each index $j \in J$ we construct child node with $I_0 = I_0 \cup \{j\}$. So, a count of child nodes for current neode is equal to the cardinality of the set of indices J. The order of enumeration indices form set J is not significant.

Note that the general scheme is principled. To implement it, we need a way for solving the problem (5), the ability to determine that the feasible set is inconsistent, the method of calculating the lower bound evaluation of the objective function for the node and its child subtree. The process of branching as described in step 5 is very general and does not preclude re-examination of the same set under a different order enabling the restrictions in it. However, this method guarantees branching viewing subsets of this.

We should also discuss the construction of the lower bound evaluation of the objective function for the node. Traditionally, for construction of the branch and bound methods is expected that the optimum value of the auxiliary problem solved in a node can be used as an evaluation for the node. But in this case it is not.

Indeed, let the auxiliary convex programming problem is solved with its dual problem. Let the fixed node of the tree is constructed for a certain set of indices $I_1 \neq \varnothing$ and to branching some index $t \notin I_1$ is added. So, $I_2 = I_1 \cup \{t\}$ and we have two duals problems $\min\limits_{y \in Y_1} \varphi_1(\boldsymbol{y})$ and $\min\limits_{y \in Y_2} \varphi_2(\boldsymbol{y})$ where

$$\varphi_1(\boldsymbol{y}) = \sup_{x \in R_n} L_1(\boldsymbol{x}, \boldsymbol{y}),\ \varphi_2(\boldsymbol{y}) = \sup_{x \in R_n} L_2(\boldsymbol{x}, \boldsymbol{y}),$$

$$L_1(\boldsymbol{x}, \boldsymbol{y}) = y_0 f(\boldsymbol{x}) + \sum_{i \in I_1} y_i f_i(\boldsymbol{x}),$$

$$L_2(\boldsymbol{x}, \boldsymbol{y}) = y_0 f(\boldsymbol{x}) + \sum_{i \in I_2} y_i f_i(\boldsymbol{x}) = y_0 f(\boldsymbol{x}) + \sum_{i \in I_1} y_i f_i(\boldsymbol{x}) + y_t f_t(\boldsymbol{x}),$$

$$Y_1 = \{\boldsymbol{y} \in R_{m+1} \mid\ y_i \geq 0, i \in I_1, y_i = 0, i \notin I_1\},\ Y_2 = \{y \in R_{m+1} \mid\ y_i \geq 0, i \in I_2, y_i = 0, i \notin I_2\}.$$

Obviously, $Y_1 \subseteq Y_2$. So, for $y \in Y_1$ we have $\varphi_1(\boldsymbol{y}) = \sup\limits_{x \in R_n} L(\boldsymbol{x}, \boldsymbol{y})$ and for $y \in Y_2$ - $\varphi_2(\boldsymbol{y}) = \sup\limits_{x \in R_n} L(\boldsymbol{x}, \boldsymbol{y})$, where the Lagrangian $L(\boldsymbol{x}, \boldsymbol{y}) = y_o f(\boldsymbol{x}) + \sum\limits_{i \in I} y_1 f_1(\boldsymbol{x})$ is constructed for full set of indeces of the problem (1). Consequently, $\min\limits_{y \in Y_2} \varphi_2(\boldsymbol{y}) \leq \min\limits_{y \in Y_1} \varphi_1(\boldsymbol{y})$. The same happens with the values of the objective function of the primal problem (5). Thus, the addition of new constraint can reduce the objective function value at the optimum auxiliary problem in the node.

4. An algorithm of branch and bound method

Consider the simplest form of the problem (1):

$$\min_{x \in D} \to f(\boldsymbol{x}) \tag{9}$$

where $f(\boldsymbol{x}) = 0.5\langle Q\boldsymbol{x}, \boldsymbol{x}\rangle + \langle d, \boldsymbol{x}\rangle$ - quadratic function with symmetric negative definite $n \times n$ matrix Q, $d \in R_n$ and feasible set has a form $D = \{\boldsymbol{x} \in R_n \mid\ \alpha_i \leq x_i \leq \beta_i, i = 1..n\}$. It should be noted that despite its simplicity, this type of problem is computationally difficult to solve, because it has 2^n local minima. Nevertheless, this type of problem allows us to formulate specific rules for the branch and bound method.

1. An easy way to solve the auxiliary problem (5) in the node tree. The dual problem to (5) also has the form of quadratic programming problem and there is an explicit formula for obtaining the solution of the direct problem (5) $\boldsymbol{x}'$ with solution of dual problem $\boldsymbol{y}'$: $\boldsymbol{x}' = -Q^{-1}[d + A^{\mathrm{T}}\boldsymbol{y}']$ ([5]).
2. Another rule for branching. The decision tree will be in binary tree form. In each node of this tree we choose the next variable for branching, for example x_t. In one of child nodes we solve the problem with additional constraint $x_t \leq \alpha_t$ and in another child node - $x_t \geq \beta_t$. So, in the step 5 of general scheme we choose a variable, but not constraint. Moreover, because the variable is not found in other limitations, there is no possibility of constructing a problem (5) incompatible. So, step 1 of general scheme will be easier.
3. The rule for calculate of lower bound evaluation. This rule is based on using known bound values of each variable. Let we have the solution of problem (5) $\boldsymbol{x}'$ in some node Write the objective function:

$$f(\boldsymbol{x}) = 0.5\sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij} x_i x_j + \sum_{i=1}^{n} b_i x_i$$

It can be seen that the lower value of the function in stationary points obtained in the child subtree, we can estimate the sum of lower bound evaluation of each term. Thus, the lower bound evaluation is the same sum, in which coordinates uncommitted variables are replaced with the bound values that give a minimum term. For example, if the only uncommitted variable x_t evaluation will be as follows:

$$f_{low} = 0.5\sum_{i=1,i\neq t}^{n}\sum_{j=1,j\neq t}^{n} q_{ij}x_i'x_j' + \sum_{i=1,i\neq t}^{n} b_i x_i' +$$

$$+\sum_{j=1,j\neq t}^{n}\min\{q_{tj}x_j'\alpha_t, q_{tj}x_j'\beta_t\} + 0.5\min\{q_{tt}\alpha_t^2, q_{tt}\beta_t^2\} + \min\{b_t\alpha_t, b_t\beta_t\}.$$

5. Results of experiments

We provide the experiment on some known test problems in form (9) which often used in research of concave optimization methods [7]:

Type 1. Q is diagonal matrix, $d=\mathbf{0}$, $q_{jj}=-(n-1-0.1j)$, $\alpha_j=-1-j$, $\beta_j=1+5j$, $j=1..n$;

Type 2. Q is diagonal matrix, $d_j=1$, $q_{jj}=-1$, $\alpha_j=-j$, $\beta_j=[n/j]$, $j=1..n$;

Type 3. Q is diagonal matrix, $d=\mathbf{0}$, $q_{jj}=-1$, $\alpha_j=-(n-j+1)$, $\beta_j=n+j/2$, $j=1..n$;

Тип 4. Q is diagonal matrix, $d_j=-j$, $q_{jj}=-1$, $\alpha_j=1-j$, $\beta_j=2j$, $j=1..n$.

Each problem is solved with $n=3...12$. In solving was measured the time of operation process, the percentage of nodes in the tree, in which auxiliary convex programming problem (5) did not solved ($P_{nosolve}$), the percentage of nodes in obtaining the global optimum (P_{rec}).The Table 1 will be given the maximum, minimum and average values of parameters $P_{nosolve}$ and P_{rec} for each type of problem.

We considered several rules for selecting the order of the variables:

Rule 1. Direct order variables, i.e. first fixed variable number is $i=1$, then – with number $i=2$. The last variable will be with number $i=n$. So, the left child node is corresponded the constraint $x_i\le\alpha_i$ and the right child node – $x_i\ge\beta_i$.

Rule 2. Reverse order variables, i.e. first fixed variable number is $i=n$, then – with number $i=n-1$. The lasr variable will be with number $i=1$. So, the left child node is corresponded the constraint $x_i\ge\beta_i$ and the right child node – $x_i\le\alpha_i$.

Rule 3. More intellectual case based on the analysis of the deviation of the boundary values of each variable from its value at the absolute maximum point of function $f(\cdot)$. Note x_{abs} is the absolute maximum point of function $f(\cdot)$. The order for variables is defined by sorted order of values $\rho_i=\max\{|x^i{}_{abs}-\alpha_i|,|x^i{}_{abs}-\beta_i|\}$. In this case left child node will be corresponded to those constraint $x_i\le\alpha_i$ or $x_i\ge\beta_i$, which bound has maximum deviation of value of i variable form absolute maximum point $x^i{}_{abs}$.

A detailed analysis of the overall process time showed the presence of proportion to the percentage of nodes in the tree, in which auxiliary convex programming problem (5) are solved $(1-P_{nosolve})$. In general, the calculation time ranged from a few seconds for $n=3..6$ up to 8 hours at $n=12$. The sharp increase in overall process time observed since $n=9$. Obviously, this was due to an increase in the computational complexity of solving the auxiliary convex programming problem in the tree nodes.

The results of experiments were carried out in the Table 1. So, we make the following conclusions on the basis of the experiment:

1. The solution of Type 1 and Type 4 problems is primarily for upper bounds of variables. So, value of P_{rec} for Rule 1 is large, while for Rule 2 it is sufficiently small. Thus, in this case, Rule 2 of the order of enumeration variables is preferable.
2. The solution of Type 2 problems is at the lower boundaries of variables. Therefore, Rule 1 of the order of enumeration variables is preferable.
3. The Rule 1 and Rule 2 of the order of enumeration variables do not account for problems specific. Therefore, the values P_{rec} are very different depending on the type of problem. When using Rule 3 is visible that indicator P_{rec} is stabilized regardless of the type of problem. On average for the optimal solution of problem (1) is sufficient to look a little less than 10% of the decision tree.
4. Dynamics of the indicator $P_{nosolve}$ does not allow to make qualitative conclusions about the effectiveness of Rule 3. For example, in the problems of Type 2 and Type 4, percentage of nodes in the tree, in which auxiliary convex programming problem (5) is not solved, is large enough, and for problems of Type 3 - is very small. This is due to low lower bound evaluation.

Table 1. A results of experiments.

	$P_{nosolve}$			P_{rec}		
	min	avg	max	min	avg	max
Type 1:						
Rule 1	0	0,04393	0,0967	0,8895	0,9636	1
Rule 2	0,258	0,3851	0,4211	0,0053	0,0803	0,0952
Rule 3	0,258	0,3851	0,4211	0,0053	0,0803	0,0952
Type 2:						
Rule 1	0,2666	0,798	0,988	0,0053	0,108	0,266
Rule 2	0	0	0	0,933	0,9838	1
Rule 3	0,5333	0,7251	0,8725	0,0053	0,0847	0,266
Type 3:						
Rule 1	0	0,0119	0,0293	0,4904	0,5333	0,5056
Rule 2	0,1333	0,165	0,1935	0,0097	0,091	0,266
Rule 3	0,1333	0,1502	0,18	0,0097	0,091	0,266
Type 4:						
Rule 1	0	0	0	1	1	1
Rule 2	0,4	0,729	0,6132	0,0097	0,091	0,266
Rule 3	0,4	0,729	0,6132	0,0097	0,091	0,266

6. Conclusions

In general, the experiments leads to the conclusion that the approach used in the article in the construction algorithm of branch and bound method, can be quite competitive in solving practical problems of concave programming. Prospects for the development of this method are related to the construction of good lower bounds in the tree nodes and the possible use of parallel computing.

References

[1] Strekalovsky A S 2003 *Principles of Nonconvex Optimization* (Novosibirsk, Nauka)
[2] Strekalovsky A S 2014 Modern Methods for Solving Nonconvex Optimal Control Problems *IIGU Ser. Matematika* **8** 141–163
[3] Gruzdeva T V, Strekalovsky A S, Orlov A V, Druzinina O V 2011 Nonsmooth minimization problems for the difference of two convex functions *Vychisl.Metody Programm* **12(4)** 384-396
[4] Audet C, Hansen P, Savard G 2005 *Essays and Surveys in Global Optimization* (New York, Springer) 312
[5] Konnov I V 2010 Sign reversion approach to concave minimization problems *Optim.Lett.* **4** 491-500
[6] Andrianova A A, Konnov I V 2014 The branch and bound method for concave optimization problem in *Problem of Theoretical Cybernetics* (Kazan, Otechestvo) 23-26
[7] Chinchuluun A, Pardalos P M, Enkhbat R 2005 *Global minimization algorithms for concave quadratic programming problems* Optimization **54** 627-639
[8] Sukharev A G, Timokhov A V, Fedorov V V 2005 *A course in optimization methods* (Moscow, Nauka) chapter 4 147-160