APPLICATION OF THE COMPUTATIONAL GEOMETRY IN LINEAR OPTIMIZATION

*Muzafer Saračević, **Aybeyan Selimi, p. 73-88

September, 2017; 2 (2)

ABSTRACT

Computers have an important role in the automated construction and production of various items and objects today. The production process is a mathematical model that develops methods for the best outcome. These models are formulated as the maximization or minimization of some target function along with given constraints and can also be observed as problems of computational geometry. Computational geometry develops efficient algorithms for optimizing these models. Computer models can be created based on objects that really exist or some imaginary object. In practice, experimenting with created models is make with imaginary objects because experimenting with them is easier than with a real object. In this paper, is given prune and search algorithm which is represents an example relation between linear programming and computational geometry.

Keywords: linear programming, feasible region, optimization, convex polygon.

***Assoc. Prof. Muzafer Saračević PhD**

International University Novi Pazar, Novi Pazar – Serbia

e - posta: muzafers@uninp.edu.rs

****Phd.Cand. Aybeyan Selimi**

International Vision University, Faculty of Informatic, Gostivar - Macedonia

e-mail: aybeyan.selim@vizyon.edu.mk

Article type:
1.02 General scientific articles

UDK: 519.852

Date of received:
July 5, 2017

Date of acceptance:
August 10, 2017

Declaration of interest:
M.S, A.S., the authors reported no conflict of interest related to this article..

1. INTRODUCTION

Linear programming is a branch of mathematics that deals with the technique for the optimization of a linear objective function, subject to linear equality and linear inequality constraints. The problem of linear programming is introduced from Leonid Kantorovich in 1939 as a method of solving the problem expenditures and returns to the army and increase losses incurred by the enemy. Also, in the United States, linear programming was developed during the Second World War primarily for problems of military logistics, such as optimizing the transportation of military and equipment to convoys. Linear programming as mathematical model is used in field of computational geometry. Computational geometry is a part of the field of algorithms and deals with the development and analysis of efficient algorithms and the structure of data suitable for geometric problems. As synthesis of geometry and computer sciences, computational geometry develops thanks to problems and applications, first of all in computer graphics, computer vision, robotics, databases, geographic information systems, Computer Aided Design / Computer Aided Manufacturing (CAD/CAM) systems, molecular biology, etc. Some of the concrete applications are applications in virtual reality, planning of movement, drug design, fluid dynamics, etc. The field of computer geometry usually deals with problems in the Euclidean plane or space and implies the availability of elementary operations such as: checking whether the point belong to line or circle, checking intersection of the lines or line segments [de Berg et al., 2008 :2]. In the introductory section we will give an overview of the optimization, focusing especially on convex optimization.

The mathematical problem of optimization, or just the problem of optimization, is problem from the following form

$$\min f(x) \qquad (1.1)$$

$$\text{Subject to} \qquad f_i\ (x) \le b_i \qquad i = 1,...m$$

The vector $x=\left(x_1,...,x_n\right)^T$ is a optimization variable of problem, the function $f:\mathbb{R}^n\rightarrow\mathbb{R}$ is the objective function, the functions $f_i:\mathbb{R}^n\rightarrow\mathbb{R}$, $i=1,...,m$, are the (inequality) constraint functions, and the constants $b_1,...,b_m$ are the limits, or bounds, for the constraints. A vector x^* is called optimal, or a solution of the problem (1.1), if it has the smallest objective value among all vectors that satisfy the constraints: $\forall z\in\mathbb{R}^n$ with $f_1(z)\le b_1,...,f_m(z)\le b_m$ we have $f(z)\ge f\left(x^*\right)$. A set $X=\left\{z\mid f_i\left(z\right)\le b_i, i=1,2,...,m\right\}$ is called feasible region of problem (1.1), and $z\in X$ is feasible point. If $X=\varnothing$ than the problem (1.1) is called infeasible optimization problem. If the objective function of problem (1.1) is unbounded than the (1.1) is unconstrained optimization problem. Usually the families or classes of the optimization problems are characterized with the certain forms of objective function and the functions of constraints. As a special case, the optimization problem (1.1) is a problem of linear programming if the objective function f and the functions of constraints $f_1,...,f_m$ are linear, that is, the equations

$$\begin{aligned} f\left(\alpha x+\beta y\right)&=\alpha f(x)+\beta f(y) \\ f_i\left(\alpha x+\beta y\right)&=\alpha f_i(x)+\beta f_i(y) \end{aligned} \tag{1.2}$$

for any $x,y\in X\subseteq\mathbb{R}^n$ and any $\alpha,\beta\in\mathbb{R}$. The convex programming problem is the one in which the objective function and the functions of constraints are convex functions, which means that the inequalities hold

$$\begin{aligned} f\left(\alpha x+\beta y\right)&\le\alpha f(x)+\beta f(y) \\ f_i\left(\alpha x+\beta y\right)&\le\alpha f_i(x)+\beta f_i(y) \end{aligned} \tag{1.3}$$

for any $x,y\in X\subseteq\mathbb{R}^n$ and any $\alpha,\beta\in\mathbb{R}$ such that $\alpha+\beta=1$, $\alpha\ge 0$, $\beta\ge 0$. If we compare (1.3) and (1.2) we can see that convexity is more general than linearity, equality is replaced by inequality, and the inequality must apply only to certain values of α and β. Since the problem of linear programming is simultaneously the problem of convex programming, we can consider convex programming as a generalization of linear programming [Boyd et al, 2005 :8].

The solving method of optimization problem is an algorithm that calculates the solution of the problem to a certain accuracy. The efficiency of these algorithms, that is, the ability to solve the optimization problem (1.1), varies greatly and depends on factors such as certain types of objective function. The solving of problem (1.1) means that one of the following four conditions is fulfilled:

- The optimal solution of (1.1) is found
- It's shown that (1.1) is unbounded from the down on X
- It's proved that $x^* = \inf_{x \in X} f(x)$ doesn't exist
- It's proved that (1.1) is infeasible problem.

2. CONCEPTUALLY AND METHODOLOGICALLY DETERMINATION

The research methodology consists of three sections: Preliminaries, Algorithm and Application.

2.1 Preliminaries

Linear programming is an optimization problem which maximizes a linear objective function under linear inequality constraints:

$$\max f(x_1, x_2, \ldots, x_n) = c_1x_1 + c_2x_2 + \ldots + c_nx_n \tag{2.1}$$

$$\text{Subject to} \quad \begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &\le b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &\le b_2 \\ \ldots \qquad\qquad & \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &\le b_m \end{aligned} \tag{2.2}$$

$$x_1 \ge 0, x_2 \ge 0, \ldots, x_n \ge 0 \tag{2.3}$$

The objective function (2.1) is linear, where $c_j,\ j = 1, 2, \ldots, n$ are coefficients $x_j,\ j = 1, 2, \ldots, n$ are structural variables of objective function. The optimization problem (2.1) – (2.3) can be written on matrix form as a

$$\max f = c^T x$$

Subject to $\quad Ax \le b \quad$ (2.4)

$$x \ge 0$$

$S = \{x | x \in \mathbb{R}^n, Ax \ge b, x \ge 0\}$ is set of possible solutions (feasible region) of problem (2.4). Feasible solution $x^* \in S$ for which $f(x) \le f(x^*)$, $\forall x \in S$ is optimal solution of problem (2.4), while $f(x^*)$ is the optimal value of objective function.

Definition 2.1 The set $C \subseteq \mathbb{R}^n$ is convex if the line segment between any two points from C completely lies in C, for any $x_1, x_2 \in C$ and any θ, $(0 \le \theta \le 1)$ we have $\theta x_1 + (1-\theta)x_2 \in C$.

Definition 2.2 A hyper-plane is a set of the form

$$\{x \mid a^T x = b\},$$

where $a \in \mathbb{R}^n$, $a \ne 0$ and $b \in \mathbb{R}$.

Definition 2.3 A closed halfspace is a set of the form

$$\{x \mid a^T x \le b\},$$

where where $a \ne 0$, *i.e.*, the solution set of one (nontrivial) linear inequality.

Definition 2.4 A polyhedron is the solution set of a finite number of linear equalities and inequalities:

$$\{x \mid a^T x \le b,\ c^T x = d\}$$

where $a, c \in \mathbb{R}^n$, $a, c \ne 0$ and $b, d \in \mathbb{R}$.

From definition we obtain that the polyhedron is intersection of a finite number of half-spaces and hyper-planes. Hyper-plane, halfspace and polyhedron are convex sets. A bounded polyhedron is called a polytope. Optimization problem which minimized the linear objective function is the dual problem of (2.4) and can be written in the form

$$\min f = b^T \lambda$$

$$\text{Subject to} \quad A^T \lambda \geq c \tag{2.5}$$

$$\lambda \geq 0$$

2.2 Linear Programming in Computational Geometry

Often in the application of computational geometry, the problems of linear programming appear [Dyer, 1995 :346]. In computational geometry, randomized algorithms are used that give the possibility of treating geometric problems in the general case [Seidel, 1991 :424]. Let $f(x_1, x_2, ..., x_n)$ is objective function of LP problem an let half-space of LP problem is defined by non-zero vector $a_i = \{a_{i1}, a_{i2}, ..., a_{in}\}$ and real number b_i such that $s_i = \{x | a_i^T x \leq b_i\}$ for $i = 1, 2, ..., n$. We partition the set of half-spaces into three sets S^-, S^0 and S^+ such that $s_i \in S^-$ if $a_i^T x < 0$, $s_i \in S^0$ if $a_i^T x = 0$ and $s_i \in S^+$ if $a_i^T x > 0$.

Definition 2.5 $P(L) = \bigcap_{i=1}^{n} s_i$ is the feasible domain of LP problem, and the range of LP problem is

$$\Sigma(L) = \bigcap_{s \in S^0} s.$$

If $P(L) \neq \emptyset$ we said LP problem is feasible, and if $P(L) = \emptyset$ than LP problem is infeasible. A linear programming problem in two dimensions is one which involves only two variables, x_1 and x_2 where each constraint is a half-plane in E^2 [Chen et al. 2002 :158]. In this paper is given the prune-and-search paradigm algorithm who solves a linear program defined by n half-planes in time O(n). The global structure of the algorithm follows the search step who decreases the range of possible solutions, and a prune step eliminates data which is irrelevant in this range [Imai, 1991 :12]. After a search and a prune step, we simply recur with the smaller set of data until the problem becomes trivial. A prune step is typically straightforward, while search steps require the sophistication for design. The main purpose in a search step is to decrease the range of possible solutions in a way that allows us to eliminate a proportional amount of the data. Thus, a search step consists of two steps which may be iterated a constant number of times: first, we find a suitable test, and second, we answer this test. In our case, a test comes as a vertical line, and we decide on which side of this line we are going to continue the search for a solution. Let with *D* we denote the set of data (*n* elements) and with *E* the range which contain all solutions [Edelsbrunner, 1987 :214]. Bellow is given the algorithm for LP problem in two dimension.

Algorithm 2.1 (Prune-and-search):

if the size of *D* is at most some constant **then**

Use a trivial procedure to solve the problem.

else

SEARCH: Iterate the following two steps some constant number of

times:

FIND_TEST: Find an appropriate test *t*.

BISECT: Decrease the range *E* which contains all solutions by

answering the test *t*.

PRUNE: Eliminate some subset of *D* which is irrelevant in *E*.

RECUR: Repeat the computation for the new sets *D* and *E*.

endif.

3. RESULTS

In this section we make the analysis for steps in Algorithm 2.1 and we give a solution of the numerical example with algorithm. To this end, we will look first **the selection problem:** Given a set $S = \{a_1, a_2, \ldots, a_n\}$ of n elements on which a linear ordering is defined, and an integer k, $1 \le k \le n$, find the k-th smallest element in the set. The concept of the prune-and-search paradigm in selection problem consist the following steps:

- $S = \{a_1, a_2, \ldots, a_n\}$ is a set of n elements
- With $p \in S$, the set S is partitioned into 3 subsets S_1, S_2, S_3:
  - $S_1=\{ a_i \mid a_i < p ,\ 1 \le i \le n\}$
  - $S_2=\{ a_i \mid a_i = p ,\ 1 \le i \le n \}$
  - $S_3=\{ a_i \mid a_i > p ,\ 1 \le i \le n \}$
- For partitioned subsets we have this three cases:
  - If $|S_1| \ge k$, then the k–th smallest element of S is in S_1, prune away S_2 and S_3.
  - Else, if $|S_1| + |S_2| \ge k$, then p is the k–th smallest element of S.
  - Else, the k–th smallest element of S is the $(k - |S_1| - |S_2|)$ –th smallest element in S_3, prune away S_1 and S_2.

Let see how the prune-and-search paradigm can be used for develop a linear-time algorithm for the two-dimensional linear programming problem [Megiddo,1984 : 119]. A general two-dimensional LP problem with inequality constraints is given as follows

$$\min c_1x_1 + c_2x_2 \qquad (3.1)$$

$$\text{Subject to} \qquad a_{i1}x_1 + a_{i2}x_2 + a_{i0} \le 0 \qquad i = 1,2,\ldots,n$$

The problem is solved if one can illustrate the feasible region satisfying the inequality constraints in the (x_1, x_2)–plane, who represent a convex polygon (see Figure 2.1). Here, instead of considering the problem in this general form, we restrict our attention to the following problem

$$\min y \qquad (3.2)$$

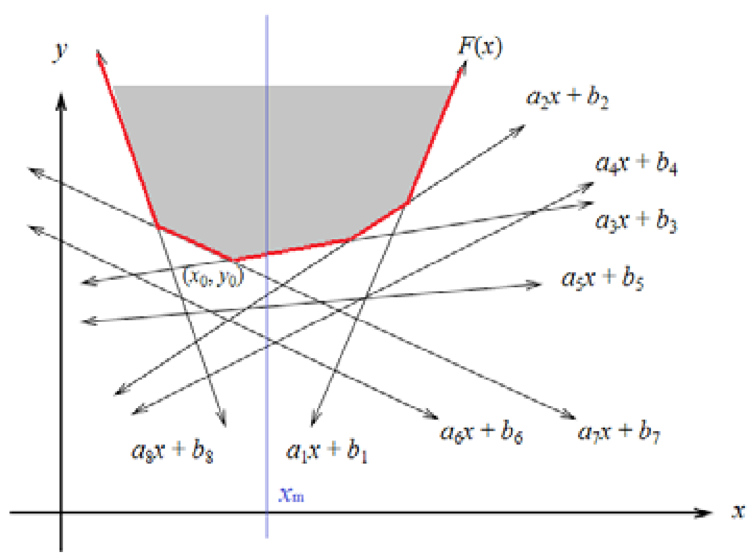
Subject to $\quad a_i x + y + b_i \leq 0 \qquad i = 1, 2, \ldots, n$

This is a special problem for linear-time algorithm for the general two-dimensional problem, which is simpler structure for better exhibit the essence of the prune-and-search technique. Figure 2.1 depicts this restricted problem for $n = 8$. Let the problem is defined as above, we define a function $f(x)$ by

$$f(x) = \max\left\{a_i x + b_i \mid i = 1,\, 2, \ldots, n\right\}.$$

Minimizing of $f(x)$ is equivalent to LP problem. The graph of $y = f(x)$ is drawn in red lines in Figure 2.1 an represent a convex function.

Figure 2.1 A two dimensional LP problem



Let we choose the point x_m on x – axis. If $x_0 < x_m$ and the intersection of $a_3x + b_3$ and $a_2x + b_2$ is greater than x_m, then one of these two constraints is always smaller than the other for $x < x_m$. Thus, this constraint can be deleted.

It is similar for $x_0 > x_m$. Let $y_m = f(x_m) = \max_{1 \leq i \leq n}\{a_i x_m + b_i\}$ and suppose an x_m is known. How do we know whether $x_0 < x_m$ or $x_0 > x_m$? To get answer this question we must look two cases.

- **Case 1:** y_m is on only one constraint and let g denote the slope of this constraint.
  - If $g > 0$, then $x_0 < x_m$.
  - If $g < 0$, then $x_0 > x_m$.
- **Case 2:** y_m is the intersection of several constraints and $g_{\max} = \max_{1 \le i \le n}\{a_i \mid a_i x_m + b_i = f(x_m)\}$ is maximal slope and $g_{\min} = \min_{1 \le i \le n}\{a_i \mid a_i x_m + b_i = f(x_m)\}$ is minimal slop of constraints.
  - If $g_{\min} > 0$, $g_{\max} > 0$, then $x_0 < x_m$
  - If $g_{\min} < 0$, $g_{\max} < 0$, then $x_0 > x_m$
  - If $g_{\min} < 0$, $g_{\max} > 0$, then (x_m, y_m) is the optimal solution.

Now the question arises as to how to choose x_m? Arbitrarily must be grouped the n constraints into n / 2 pairs. For each pair, must be found their intersection. Among these n /2 intersections, must be choose the median of their x-coordinates as x_m.

The prune – and – search approach with Input: $S : a_i x + b_i$, i =1, 2, …, n (constraints) and Output: the value x_0 such that y is minimized at x_0 subject to the above constraints, consist the following steps.

- Step 1: If S contains no more than two constraints, solve this problem by a brute force method.
- Step 2: Divide S into n/2 pairs of constraints randomly. For each pair of constraints $a_i x + b_i$ and $a_j x + b_j$, find the intersection p_{ij} of them and denote its x-value as x_{ij}.
- Step 3: Among the x_{ij}'s, find the median x_m.
- Step 4: Determine $y_m = f(x_m) = \max_{1 \le i \le n}\{a_i x_m + b_i\}$

  $$g_{\min} = \min_{1 \le i \le n}\{a_i \mid a_i x_m + b_i = f(x_m)\}$$

  $$g_{\max} = \max_{1 \le i \le n}\{a_i \mid a_i x_m + b_i = f(x_m)\}$$
- Step 5:

Case 5a: If g_{min} and g_{max} are not of the same sign, y_m is the solution and exit.

Case 5b: otherwise, $x_0 < x_m$, if $g_{min} > 0$, and $x_0 > x_m$, if $g_{min} < 0$.

- Step 6:

Case 6a: If $x_0 < x_m$, for each pair of constraints whose x-coordinate intersection is larger than x_m, prune away the constraint which is always smaller than the other for $x \le x_m$.

Case 6b: If $x_0 > x_m$, do similarly.

Let S denote the set of remaining constraints. Go to Step 2.

There are totally $\lfloor n/2 \rfloor$ intersections. Thus, $\lfloor n/4 \rfloor$ constraints are pruned away for each iteration.

Time complexity: T(n) = T(3n/4)+O(n) = O(n) [Megiddo, 1983 :762]

The general two-variable linear programming problem solution is a procedure of finding piece - wise linear convex function of the x – axis [Preparata et al. 1985 :293]. The problem (3.1) can be transformed by setting $Y = c_1x_1 + c_2x_2$ and $X = x$ as follows

$$\min Y \tag{3.3}$$

Subject to $\alpha_i X + \beta_i Y + a_{i0} \le 0 \qquad i = 1, 2, ..., n$

where $\alpha_i = (a_{i1} - (c_1/c_2))a_{i2}$ and $\beta_i = a_{i2}/c_2$. Depending upon whether β_i is zero, negative or positive we partition the index set on three subsets I_0, I_-, I_+ respectively. All constraints whose index is in I_0 are vertical lines and determines the feasible intervals for X as follows

$$u_1 \le X \le u_2$$
$$\text{„} u_1 = \max\{-a_{i0}/\alpha_i \mid i \in I_0, \alpha_i < 0\}$$
$$u_2 = \min\{-a_{i0}/\alpha_i \mid i \in I_0, \alpha_i > 0\}$$

In the other hand letting $-(\alpha_i / \beta_i) \triangleq \delta_i$ and $-(a_{i0} / \beta_i) \triangleq \gamma_i$ all constraints on I+ are of the form

$$Y \le \delta_i X + \gamma_i \qquad i \in I_+$$

so that collectively define a piece – wise linear upward – convex function F+ (x) of the form

$$F_+(x) \triangleq \min_{i \in I_+} \left(\delta_i X + \gamma_i \right)$$

.

Similarly the constraints in I – defines piece-wise linear downward – convex functions F– (x) of the form

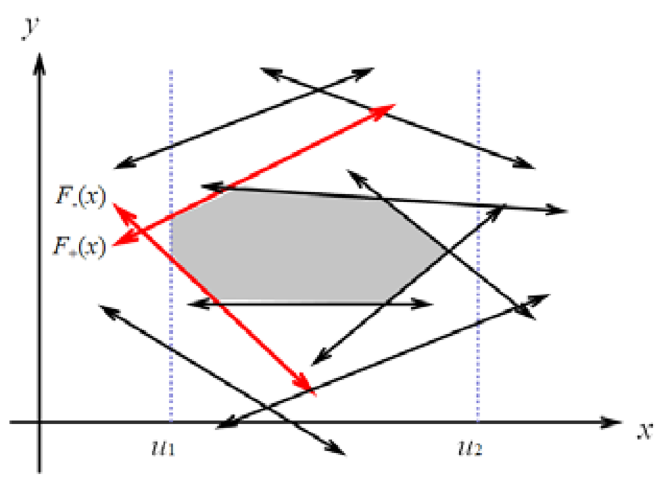
$$. \; F_-(x) \triangleq \max_{i \in I_-} \left(\delta_i X + \gamma_i \right)$$

In this way we obtain the transformed constraint $F\_(x) \le Y \le F_+(x)$, and since we have minimizing LP problem $F\_(x)$ is our objective function. The problem is

$$\begin{aligned} & \min F_-(x) \\ \text{Subject to} \quad & F_-(X) \le F_+(X) \,. \\ & u_1 \le X \le u_2 \end{aligned}$$

Let $H(x) = F\_(x) - F_+(x)$.

Figure 2. 2 Ilustration $F\_(x)$, $F_+(x)$, u_1 and u_2 in the reformulation of the LP problem



If we know $x_0 < x_m$, then $a_1x + b_1$ can be deleted because $a_1x + b_1 < a_2x + b_2$ for $x < x_m$.

Define:

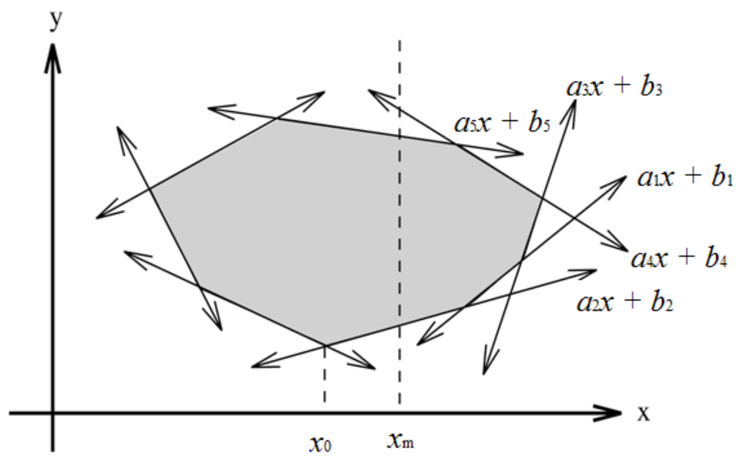
$g_{min} = \min \{a_i \mid i \in I_-, a_i x_m + b_i = F_-(x_m)\}$, minimal slope

$g_{max} = \max \{a_i \mid i \in I_-, a_i x_m + b_i = F_-(x_m)\}$, maximal slope

$h_{min} = \min \{a_i \mid i \in I_+, a_i x_m + b_i = F_+(x_m)\}$, minimal slope

$h_{max} = \max \{a_i \mid i \in I_+, a_i x_m + b_i = F_+(x_m)\}$, maximal slope

Figure 2. 2 Ilustration of possible cases for $F_-(x) > F_+(x)$



- **Case 1:** If $F(x_m) \le 0$, then x_m is feasible.
  - If $g_{min} > 0$, $g_{max} > 0$, then $x_0 < x_m$.
  - If $g_{min} < 0$, $g_{max} < 0$, then $x_0 > x_m$.
  - If $g_{min} < 0$, $g_{max} > 0$, then x_m is the optimum solution.
- **Case 2:** If $F(x_m) > 0$, x_m is infeasible.
  - If $g_{min} > h_{max}$, then $x_0 < x_m$.
  - If $g_{min} < h_{max}$, then $x_0 > x_m$.
  - If $g_{min} \le h_{max}$, and $g_{max} \ge h_{min}$, then no feasible solution exists.

The prune – and – search approach for general two variable LP problem with Input: $I_-: y \ge a_i x + b_i$, $i = 1, 2, \ldots, n_1$ $I_+: y \le a_i x + b_i$, $i = n_1+1, n_1+2, \ldots, n$, $a \le x \le b$ (constraints) and Output: the value x_0 such that y is minimized at x_0 subject to the above constraints, consist the following steps.

- Step 1: Arrange the constraints in I_1 and I_2 into arbitrary disjoint pairs respectively. For each pair, if $a_i x + b_i$ is parallel to $a_j x + b_j$, eliminate $a_i x + b_i$ if $b_i < b_j$ for $i, j \in I_-$ or $b_i > b_j$ for $i, j \in I_+$. Otherwise, find the intersection p_{ij} of $y = a_i x + b_i$ and $y = a_j x + b_j$. Let the x-coordinate of p_{ij} be x_{ij}.

- Step 2: Find the median x_m of x_{ij}'s (at most $\lfloor n/2 \rfloor$ points).

- Step 3:

    - If x_m is optimal, report this and exit.
    - f no feasible solution exists, report this and exit.
    - Otherwise, determine whether the optimum solution lies to the left, or right, of x_m.

- Step 4: Discard at least 1/4 of the constraints. Go to Step 1.

The general approach given on this paper by Megiddo is applied for minimum enclosing circle of n point set.

4. CONCLUSION & DISCUSSION

Linear programming as a central problem in the discrete-algorithm study plays a very important role in solving numerous combinatorial optimization problems. Because of its various applications in many areas, the problem of linear programming is gaining great attention in the field of computational geometry. Linear programming can also be viewed as computational geometry problems in which the feasible region is the cross-section of the half-spaces determined by their constraints. For these problems, the target function is minimized or maximized in the convex polyhedron field. There are several known problems in computational geometry such as the smallest circle, extreme point, farthest point that are closely connected from the n points in the plane. These problems are considered as problems of linear programming with n variables and in the end dimension $\mathbb{R}^2$ consists of finding a point P which is a convex combination of other n points from $\mathbb{R}^2$. Another problem of computational geometry that is serious for O (n) is the problem of finding the smallest circle enclosing n given points in the plane. In the end we can conclude that the linear programming give good basis for further investigation in the low dimensional space treated in computational geometry.

5. REFERENCES:

1. Boyd Stephen., Vandenberghe Lieven., *Convex Optimization*, Cambridge University Press, 2004
2. Danny Z. Chen , Jinhui Xu Two-variable linear programming in parallel, Computational Geometry 21 (2002) 155–165
3. F.P. Preparata, M.I. Shamos, Computational Geometry: An Introduction, Springer, Berlin, 1985.
4. H. Edelsbrunner, Algorithms in Combinatorial Geometry, Springer, New York, 1987
5. Hiroshi Imai, Computational Geometry and Linear Programming, Algorithms & Architectures: Proceedings of the Second NEC Research Symposium, (1991), 9 – 28
6. M.E. Dyer, A parallel algorithm for linear programming in fixed dimension, in: Proc. of 11th Annual Symp. on Computational Geometry, 1995, pp. 345–349.
7. Mark de Berg , Otfried Cheong, Marc van Kreveld, Mark Overmars, Computational Geometry Algorithms and Applications, Springer-Verlag Berlin Heidelberg, 2008
8. Nimrod Megiddo, Linear programming in linear time when the dimension is fixed, J. ACM 31 (1) (1984) 114–127.
9. Nimrod Megiddo, Linear programming in linear time when the dimension is fixed, J. ACM 31 (1) (1984) 114–127.
10. Nimrod Megiddo, Linear time algorithms for linear programming in R^3 and related problems, SIAM Journal of Computation 12 (4) (1983) 759–776.
11. R. Seidel, Small-dimensional linear programming and convex hulls made easy, Discrete Computational Geometry. 6 (1991) 423–434.