A New Algorithm for Linear Programming

Dhananjay P. Mehendale
Department of Electronic Science, Sir Parashurambhau College, Tilak Road, Pune-411030, India
dhananjay.p.mehendale@gmail.com

***Abstract-* In this paper we propose a new algorithm for linear programming. This new algorithm is based on treating the objective function as a parameter. We transform the matrix of coefficients representing this system of equations in the reduced row echelon form containing only one variable, namely, the objective function itself, as a parameter whose optimal value is to be determined. We analyze this matrix and develop a clear method to find the optimal value for the objective function treated as a parameter. We see that the entire optimization process evolves through the proper analysis of the said matrix in the reduced row echelon form. It will be seen that the optimal value can be obtained 1) by solving certain subsystem of this system of equations through a proper justification for this act, or 2) By making appropriate and legal row transformations on this matrix in the reduced row echelon form so that all the entries in the submatrix of this matrix, obtained by collecting rows in which the coefficient of so called unknown parameter *d* whose optimal value is to be determined, become nonnegative and this new matrix must be equivalent to original matrix in the sense that the solution set of the matrix equation with original matrix and matrix equation with transformed matrix are same. We then proceed to show that this idea naturally extends to deal with nonlinear and integer programming problems. For nonlinear and integer programming problems we use the technique of Grobner bases since Grobner basis is an equivalent of reduced row echelon form for a system of nonlinear equations, and the methods of solving linear Diophantine equations respectively.**

Key Words: Objective function as a parameter, objective equation, objective plane, constraint planes, echelon form, Grobner bases, Diophantine equations

I. Introduction

There are two types of **linear programs** (linear programming problems):

1. Maximize: $C^T x$
Subject to: $Ax \le b$
$x \ge 0$
Or
2. Minimize: $C^T x$
Subject to: $Ax \ge b$
$x \ge 0$

where X is a column vector of size n×1 of unknowns. Where C is a column vector of size n×1 of profit (for maximization problem) or cost (for minimization problem) coefficients, and C^T is a row vector of size 1×n obtained by matrix transposition of C. Where A is a matrix of constraints coefficients of size m×n. Where b is a column vector of constants of size m×1 representing the boundaries of constraints.

By introducing the appropriate slack variables (for maximization problem) and surplus variables (for minimization problem), the above mentioned linear programs gets converted into **standard form** as:

Maximize: $C^T x$
Subject to: $Ax + s = b$ (1.1)
$x \ge 0, s \ge 0$

Where S is slack variable vector of size m×1.
This is a **maximization problem**.
Or

Minimize: $C^T x$
Subject to: $Ax - s = b$ (1.2)
$x \ge 0, s \ge 0$

Where S is surplus variable vector of size m×1.
This is a **minimization problem**.

In geometrical language, the constraints defined by the inequalities form a region bounded by a convex polyhedron, a region bounded by the **constraint planes** $Ax_i = b_i$, called **feasible region** and it is straightforward to check that there exists at least one vertex of this polyhedron at which the optimal solution for the problem is situated when the problem at hand is not unbounded or infeasible. There may be unique optimal solution and sometimes there may be infinitely many optimal solutions, e.g. when one of the constraint planes is parallel to the objective plane we may have a multitude of optimal solutions. An entire plane or an entire edge can constitute the optimal solution set.

We begin with some common notions and definitions that are prevalent in the literature. A variable x_i is called **basic variable** in a given equation if it appears with unit coefficient in that equation and with zero coefficients in all other equations. A variable which is not basic is called **nonbasic variable**. A sequence of elementary row operations that changes a given system of linear equations into an **equivalent system** (having the same solution set) and in which a given nonbasic variable can be made a basic variable is called a **pivot operation**. An equivalent system containing basic and nonbasic variables obtained by application of suitable elementary row operations is called **canonical system**. At times, the introduction of slack variables for obtaining standard form automatically produces a canonical system, containing at least one basic variable in each equation. Sometimes a sequence of pivot operations is needed to be performed to get a canonical system. The solution obtained from canonical system by setting the nonbasic variables to zero and solving for the basic variables is called **basic solution** and in addition when all the variables have nonnegative values the solution satisfying all the imposed constraints is called a **basic feasible solution**.

Because of the far great practical importance of the linear programs and other similar problems in the operations research it is a most desired thing to have an algorithm which works in a **single step**, if not, in as few steps as possible. No method has been found which will yield an optimal solution to a linear program in a single step ([1], Page 19). We aim to propose an algorithm for linear programming which aims at fulfilling this requirement in a best possible and novel way.

II. A New Algorithm for Linear Programming

We start with the following equation:

$$C^T x = d \tag{2.1}$$

and call it **objective equation.** The (parametric) plane defined by this equation will be called **objective plane**. Thus, we have taken the objective function as a new **unknown parameter called** d and the problem of linear programming is to find the **optimal value** of this **unknown parameter**.

We discuss first the **maximization** problem. A similar approach for minimization problem will be discussed next.

Given a maximization problem**,** we first construct the combined system of equations containing the objective equation (2.1) and the equations defined by the constraints imposed by the problem under consideration, combined into a single matrix equation as follows:

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} d \\ b \end{bmatrix} \tag{2.2}$$

Let **E**= $\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix}$, and let **F** = $\begin{bmatrix} d \\ b \end{bmatrix}$

Let **[E, F]** denote the augmented matrix obtained by appending the column vector **F** to matrix **E** as a last column. We then find *R*, the **reduced row echelon form** ([2], pages 73-75) of the above augmented matrix **[E, F]**. Thus,

$$R = \text{rref}([\mathbf{E},\mathbf{F}]) \tag{2.3}$$

Note that the augmented matrix **[E, F]** as well as its reduced row echelon form *R* contains **only one variable**, namely, d and all other entries are **constants**. From *R* we can determine the solution set S for every fixed $d, S = \{\begin{bmatrix} x \\ s \end{bmatrix} /(fixed) d \in reals\}$. The subset of this solution set of vectors $\begin{bmatrix} x \\ s \end{bmatrix}$ which also satisfies the nonnegativity constraints is the set of all feasible solutions for that d**.** It is clear that this subset can be **empty** for a particular choice of d that is made. The maximization problem of linear programming is to determine the unique d which provides a feasible solution and has maximum value for d, i.e., to determine the unique d which provides an optimal solution. In the case of an **unbounded** linear program there is no upper (lower, in the case of minimization problem) limit for the value of d**,** while in the case of an **infeasible** linear program the set of feasible solutions is empty. The steps that will be executed to determine the optimal solution should also tell by implication when such optimal solution does not exist in the case of an unbounded or infeasible problem.

The general form of the **matrix *R*** representing the reduced row echelon form is

$$R = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & b_{11} & b_{12} & \cdots & b_{1m} & c_1 d + e_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_{21} & b_{22} & \cdots & b_{2m} & c_2 d + e_2 \\ a_{31} & a_{32} & \cdots & a_{3n} & b_{31} & b_{32} & \cdots & b_{3m} & c_3 d + e_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_{m1} & b_{m2} & \cdots & b_{mm} & c_m d + e_m \\ a_{(m+1)1} & a_{(m+1)2} & \cdots & a_{(m+1)n} & b_{(m+1)1} & b_{(m+1)2} & \cdots & b_{(m+1)n} & c_{(m+1)} d + e_{(m+1)} \end{bmatrix}$$

Among first $(n+m)$ columns of *R* certain first columns correspond to **basic variables** (columns that are unit vectors) and the remaining ones to **nonbasic variables** (columns that are not unit vectors). For solving a linear program we need to determine the **values** of nonbasic variables such that the value of d is optimal, from which we can determine the values of all the basic variables by substitution and the linear program is thus solved completely. Note that the rows of *R* actually represent equations with variables $x_i, i = 1,2,\cdots n$ and variables $s_j, j = 1,2,\cdots m$ on left side and expressions of type $c_k d + e_k, k = 1,2,\cdots(m+1)$ containing the variable d on the right side. The rows with a positive coefficient for the parameter d represent those equations in which the parameter d can be increased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. So, these equations with a positive coefficient for the parameter d are not implying any upper bound on the maximum possible value of parameter d however; these rows are useful in certain situations. The rows with a negative coefficient for the parameter d represent those equations in which the parameter d cannot be increased arbitrarily without violating the nonnegativity constraints on variables x_i, s_j. So, these equations with a negative coefficient for the parameter d are implying an upper bound on the maximum possible value of parameter d and so important ones in this respect. So, we now proceed to find out the **submatrix of** R **,** namely**,** R_N **,** made up of all columns of *R* and containing those rows j of *R* for which the coefficients c_j of the parameter d are negative. Let $c_{i_1}, c_{i_2}, \cdots, c_{i_k}$ are all and are only negative real numbers in the rows of R collected in R_N and the coefficients of d in all other rows of *R* are greater than or equal to zero. Our method for solving linear programming problem essentially consists of solving certain subsystem of equations represented by certain tows of R or R_N, or alternatively manipulating entries in the columns of R through certain suitable row transformations so that the transformed matrix still represents the same system of equations in essence and now the entries in the columns corresponding to all nonbasic variables in R_N are nonnegative.

Algorithm 2.1 (Maximization):

Step 1: Express the given problem in standard form:

Maximize: $C^T x$

Subject to: $Ax + s = b$

$x \geq 0, s \geq 0$

Step 2: Construct the augmented matrix [**E F**], where

$$\mathbf{E} = \begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix}, \text{ and } \mathbf{F} = \begin{bmatrix} d \\ b \end{bmatrix}$$

and obtain the reduced row echelon form:

R = rref ([**E, F**]).

Note:

(a) We call that variable 'basic variable' for which the corresponding column (column vector) of R representing coefficients for that variable is 'unit vector'.

(b) We call that variable 'nonbasic variable' for which the corresponding column (column vector) of R representing coefficients for that variable is '<u>not</u> unit vector'.

Step 3: If there is a row (or rows) of zeroes at the bottom of R in the first n columns and containing a nonzero constant in the last column then declare that the problem is **inconsistent** and stop.

Step 4: Else if the coefficients of d in the last column are all positive or if there exists a column of R corresponding to some nonbasic variable with all entries negative then declare that the problem at hand is **unbounded** and stop.

Step 5: Else if for any chosen value of d one observes that nonnegativity constraint for some variable gets violated by at least one of the variables then declare that the problem at hand is **infeasible** and stop.

Note:

(c) Infeasibility of a linear programming problem can also be decided by unbounded nature of its dual problem

Step 6: Else find the submatrix of R, say R_N made up of those rows of R for which the coefficient of d in the last column is negative.

Step 7: Solve $c_{i_r} d + e_{i_r} = 0$ for each such a term in the last column of R_N and find the value of $d = d_{i_r}$ for $r = 1, 2, \cdots, k$ and find $d_{\min} = \min\{d_{i_r}\}$ and $d_{\max} = \max\{d_{i_r}\}$.

Note:

(d) The optimal value of d_{optimal}, lies in between $d_{\min} = \min\{d_{i_r}\}$ and $d_{\max} = \max\{d_{i_r}\}$

Step 8: Check the columns of R_N corresponding to nonbasic variables. Find out the columns with all entries nonnegative and set these nonbasic variables to zero.

Step 9: If all these columns corresponding to nonbasic variables contain only nonnegative entries then (as per Step 7) set all nonbasic variables to zero. Substitute $d = d_{\min}$ in the last column of R. Determine the basic feasible solution (which will be the optimal solution for the problem) and stop.

Step 10: Finally, when there exist columns in R_N corresponding to some nonbasic variables containing positive entries in some rows and some negative entries in some other rows and the initial d values are such that they approach each other when the values of these nonbasic variables are increased from zero then we can take following two approaches:

(A) We form and solve a suitable subsystem of equations mainly containing these nonbasic variables and may sometime containing some basic variables that leads to maximal value for d and thus solve the problem completely. In such case we may require to check optimality by separately appending each basic variable.

(B) We carry out certain suitable row transformations on the matrix R so that the newly transformed matrix is equivalent to R and now all the entries in the columns of newly transformed R_N (as an effect of these transformations carried out on R) corresponding to nonbasic variables are nonnegative.

We can also deal with **minimization** problems on similar lines and develop similar algorithm for their solution.

Example 2.1: We now consider a problem for which the simplex iterations are **exponential** function of the size of the problem. A problem belonging to the class described by Klee and Minty containing n variables requires $2^n - 1$ simplex steps. We see that the new method doesn't require any special effort

Maximize: $100x_1 + 10x_2 + x_3$

Subject to: $x_1 \leq 1$

$20x_1 + x_2 \leq 100$

$200x_1 + 20x_2 + x_3 \leq 10000$

$x_1, x_2, x_3 \geq 0$

Solution: The following is the matrix R:

$$R=\begin{bmatrix}1 & 0 & 0 & 0 & 1/10 & -1/100 & -90+(1/100)d\\ 0 & 1 & 0 & 0 & -1 & 1/5 & 1900-(1/5)d\\ 0 & 0 & 1 & 0 & 0 & -1 & 2d-10000\\ 0 & 0 & 0 & 1 & -1/10 & 1/100 & 91-(1/100)d\end{bmatrix}$$

So clearly,

$$R_N=\begin{bmatrix}0 & 1 & 0 & 0 & -1 & 1/5 & 1900-(1/5)d\\ 0 & 0 & 0 & 1 & -1/10 & 1/100 & 91-(1/100)d\end{bmatrix}$$

The first four columns of R, R_N correspond to basic variables, while the next two columns correspond to nonbasic variables. The last column corresponds to entries of type $c_{i_r}d+e_{i_r}$. The columns for variables s_1, s_3 of R_N contain all positive entries so we set s_3 = 0. The column for variable s_2 contains entries with negative signs, so for these rows the value of parameter d is unbounded, since by assigning any large (positive) value to this nonbasic variable we can increase the value of parameter d in these rows to any high value without violating the nonnegativity constraints. So, as mentioned above, we need to append a row from R with positive sign for d and construct the system $Pz=Q$. So, by appending first row of R and unit vector for s_1 we have

$$P=\begin{bmatrix}0 & 1/10 & -1/100\\ 0 & -1 & 1/5\\ 1 & -1/10 & 1/100\end{bmatrix},\ z=\begin{bmatrix}s_1\\ s_2\\ d\end{bmatrix},\text{ and}$$

$$Q=\begin{bmatrix}-90\\ 1900\\ 91\end{bmatrix}$$

The solution for this system yields s_1 = 1, s_2 = 100, d = 10000. The substitution of these values and the already fixed value of s_3, namely, $s_3=0$, we get the complete solution as follows:

x_1 = 0, x_2 = 0, x_3 = 10000, s_1 = 1, s_2 = 100, s_3 = 0, and the maximum value of d = 10000.

Alternative Solution: Let R_k represents *k*-th row of R. With the aim of achieving the nonnegativity of entries in the columns corresponding to nonbasic variables in R_N we perform following suitable row transformations on R, namely, $R_2 \rightarrow 10R_1 + R_2$, $R_4 \rightarrow R_1 + R_4$. With these transformations on R we get the new (transformed) R_N as follows:

$$R_N=\begin{bmatrix}0 & 1 & 0 & 0 & 0 & 1/10 & (1000-(1/10)d)\\ 1 & 0 & 0 & 1 & 0 & 0 & 1\end{bmatrix}$$

Now, using nonnegativity of 2nd and 6th column from first row we have x_1 = 0, s_3 = 0, and so, d = 10000. Further using these values in newly transformed R we can find the complete solution and check that it turns out same as we obtained above by solving subsystem of equations.

III. A New Algorithm for Nonlinear Programming

We now show that we can deal with **nonlinear programming problems** using the similar technique. Here the methods developed by **Bruno Buchberger** which transformed the abstract notion of **Grobner basis** into a fundamental tool in computational algebra will be utilized. The technique of Grobner bases is essentially a **version** of reduced row echelon form (used above to handle the linear programs made up of linear equations) for **higher degree** equations [3].

A typical nonlinear program can be stated as follows:

Maximize: $f(x)$

Subject to: $h_j(x)=0, j=1,2,\cdots,m$

$g_j(x)\le 0, j=m+1,m+2,\cdots,p$

$x_k\ge 0, k=1,2,\cdots,n$

Given a nonlinear optimization problem we first construct the following nonlinear system of equations:

$$f(x)-d=0 \quad (3.1)$$

$$h_j(x)=0, j=1,2,\cdots,m \quad (3.2)$$

$$g_j(x)+s_j=0, j=m+1,m+2,\cdots,p \quad (3.3)$$

where d is the unknown parameter whose optimal value is to be determined subject to nonnegativity conditions on problem variables and slack variables. For this to achieve we first transform the system of equations into an equivalent system of equations bearing the same solution set such that the system is easier to solve. We have seen so far that the effective way to deal with linear programs is to obtain the reduced row echelon form for the combined system of equations incorporating objective equation and constraint equations. We will see that for the nonlinear case the effective way to deal with is to obtain the equivalent of reduced row echelon form, namely, the **Grobner basis representation** for this system of equations (3.1)-(3.3). We then set up the equations obtained by equating the partial derivatives of d with respect to problem variables x_i and slack variables s_i to zero and utilize the standard theory and methods used in **calculus**. We demonstrate the essence of this method by solving an example:

Example 3.1: Maximize: $-8x_1^2-16x_2^2+24x_1+56x_2$

Subject to: $x_1 + x_2 \le 4$

$$2x_1 + x_2 \le 5$$

$$-x_1 + 4x_2 \ge 2$$

$$x_1, x_2 \ge 0$$

Solution: We build the following system of equations:

$$-8x_1^2 - 16x_2^2 + 24x_1 + 56x_2 - d = 0$$

$$x_1 + x_2 + s_1 - 4 = 0$$

$$2x_1 + x_2 + s_2 - 5 = 0$$

$$-x_1 + 4x_2 - s_3 - 2 = 0$$

We now transform the above system of equations and obtain its Grobner basis representation as follows:

$$504 - 9d + 8s_2 - 16s_2^2 + 56s_3 - 8s_3^2 = 0 \qquad (3.1.1)$$

$$9 - 9s_1 + 5s_2 - s_3 = 0 \qquad (3.1.2)$$

$$-9 + s_2 - 2s_3 + 9x_2 = 0 \qquad (3.1.3)$$

$$-18 + 4s_2 + s_3 + 9x_1 = 0 \qquad (3.1.4)$$

from first equation (3.2.1), in order to maximize d, we determine the values of s_2, s_3 as follows:

If we set $\dfrac{\partial d}{\partial s_2} = 0$ we get the value of s_2 that maximizes d, namely, $s_2 = \dfrac{1}{4}$. Similarly, if we set $\dfrac{\partial d}{\partial s_3} = 0$ we get the value of s_3 that maximizes d, namely, $s_3 = \dfrac{7}{2}$. Putting these values of s_2, s_3 in the first and second equation we get respectively the maximum value of $d = 67$ and the value of $s_1 = \dfrac{3}{4}$. Using further these values in the third and fourth equation we get $x_1 = 1.5, x_2 = 1.75$.

Note: It is instructive to notice that the optimal solution that we have obtained is **<u>not on the boundary</u>** of feasible region (as one always gets in the case of a linear programming problem) but in the interior of the feasible region.

IV. A New Algorithm for Integer Programming

We now proceed to deal with **integer programming problems** using the similar technique. The essential difference in this case is that we obtain integer solutions by treating the **system** of equations as a set of **Diophantine equations**

We begin (as done previously for linear programming problems) with the following equation:

$$C^T x = d \qquad (4.1)$$

where d is an **unknown parameter**, and call it **objective equation.** The (parametric) plane defined by this equation will be called **objective plane**. Let C^T be a row vector of size 1×n and made up of integer components $c_1, c_2, \cdots, c_n$, not all zero. It is clear that the objective equation will have integer solutions if and only if gcd (greatest common divisor) of $c_1, c_2, \cdots, c_n$ divides d. We now proceed to form the following Diophantine system of equations as a single matrix equation:

We discuss the **maximization** problem. A similar approach for minimization problem can be developed.

Given a maximization problem**,** we first construct the combined system of equations containing the objective equation and the equations defined by the constraints imposed by the problem under consideration, combined into a single matrix equation, viz.,

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} \\ A_{(m\times n)} & I_{(m\times m)} \end{bmatrix} \begin{bmatrix} x \\ s \end{bmatrix} = \begin{bmatrix} d \\ b \end{bmatrix} \qquad (4.2)$$

and obtain LP-relaxation solution. This LP-relaxation solution provides the upper bound that must be satisfied by the optimal integer solution. Then we proceed to solve the system as a system as Diophantine system of equations as follows: In order to solve this system as Diophantine system of equations we use the standard technique given in ([4], pages 212-224). First by appending new variables $u_1, u_2, \cdots, u_{(m+n)}$ and carrying out appropriate **row and column transformations** discussed in ([4], pages 217, 221) we obtain the **parametric solutions** for the system. Thus, we start with the following **table**:

$$\begin{bmatrix} C^T_{(1\times n)} & 0_{(1\times m)} & d \\ A_{(m\times n)} & I_{(m\times m)} & b \\ & & \\ \multicolumn{2}{c}{I_{((m+n)\times(m+n))}} & \end{bmatrix} \qquad (4.3)$$

and transform the system of equations into an equivalent system that is diagonal. Thus, we have the following **parametric solution**:

$u_k = d$ (for some k)

$u_{i_r} = h_{i_r}$ (where h_{i_r} are constants for $r = 1$ to n , $i_r \ne k$), and

$$x_i = \sum_{r=1}^{n} \alpha_{ij_r} u_{j_r} + \delta_i \text{ (where } \alpha_{ij_r}, \delta_i \text{ are constants.)}$$

$$s_i = \sum_{r=1}^{n} \beta_{ij_r} u_{j_r} + \eta_i \text{ (where } \beta_{ij_r}, \eta_i \text{ are constants.)}$$

We setup procedure to analyze this parametric solution and determine the optimal integral solution.

Example 4.1: Maximize: $-x_1 + 10x_2$

Subject to: $-x_1 + 5x_2 \leq 25$

$2x_1 + x_2 \leq 24$

$x_1, x_2 \geq 0$, and integers.

Solution: We first find the LP-relaxation optimal value, which is **58.636** for this problem. And the complete LP-relaxation optimal solution is

(x_1, x_2, s_1, s_2) = (8.6364, 6.7273, 0, 0)

Thus, the upper limit for optimal value for integer program, say $d_{opt.}$, can be 58. Starting with the table (4.5) mentioned above and carrying out the appropriate row-column transformations we get the following parametric solution:

$$u_1 = -d \qquad (4.1.1)$$

$$u_3 = 25 \qquad (4.1.2)$$

$$u_4 = 24 \qquad (4.1.3)$$

$$x_1 = -d + 10u_2 \qquad (4.1.4)$$

$$x_2 = u_2 \qquad (4.1.5)$$

$$s_1 = -d + 5u_2 + 25 \qquad (4.1.6)$$

$$s_2 = 2d - 21u_2 + 24 \qquad (4.1.7)$$

Using the upper limit on the optimal value, namely, $d = d_{opt.} = 58$ in the last equation above we see that the maximum value that u_2 can take (to maintain nonnegativity of s_2) is **6**. The forth and sixth equation given above for x_1 and s_1 respectively contains d with a negative coefficient ($=-1$). so substituting s_1= 0 and u_2 = 6 in equation (4.1.6) we get the desired optimal solution

$$d = 55, s_1 = 0, s_2 = 2, x_1 = 5, x_2 = u_2 = 6$$

Acknowledgements

I am thankful to Dr. M. R. Modak and Dr. P. S. Joag for useful discussions.

References

[1] Hadley G., *Linear Programming*, Narosa Publishing House, New Delhi 110 017, Eighth Reprint, 1997.

[2] Goldberg Jack L., *Matrix Theory with Applications*, McGraw-Hill, International Editions, 1992.

[3] William W. Adams, Philippe Loustaunau, *An Introduction to Grobner Bases, Graduate Studies in Mathematics*, Volume 3, American mathematical Society.

[4] Niven I., Zuckerman H., Montgomery H., *An Introduction to the Theory of Numbers*, John Wiley & Sons, Pvt. Ltd., 1991.