A Method for Finding *All* Solutions of a Linear Complementarity Problem

H. P. Williams

Operational Research Group,
Department of Management,
London School of Economics and Political Science

Working Paper LSEOR 07.96 ISBN: 978-0-85328-052-1

First published in Great Britain in 2007
by the Operational Research Group, Department of Management
London School of Economics and Political Science

Typeset, printed and bound by:
The London School of Economics and Political Science
Houghton Street
London WC2A 2AE

Working Paper No: LSEOR 07.96 ISBN: 978-0-85328-052-1

A Method of Finding *All* Solutions of a Linear Complementarity Problem

H.P Williams

Operational Research Group,
Department of Management
London School of Economics

Abstract

We define the Linear Complementarity Problem (LCP) and outline its applications including those to Linear Programming (LP), Quadratic Programming (QP), Two person Non-Zero Sum Games and Evolutionary Games. Then we briefly discuss previous methods of solution emphasising the problem of finding all solutions. A new algorithm is then presented, and illustrated by a numerical example, which finds all solutions.
It works by successive transformations of variables in order to eliminate the equations in the model.

Keywords: Linear Complementarity Problem, Linear Programming, Quadratic Programming, Non-Zero Sum Games, Evolutionary Games, Economic Equilibria.

1. Introduction

This Linear Complementarity Problem (LCP) can be written as the problem of finding solutions x and y to:

$$\boldsymbol{y} + \boldsymbol{M}\boldsymbol{x} = \boldsymbol{d} \tag{1}$$

$$\boldsymbol{x}, \boldsymbol{y} \geq \mathbf{0} \tag{2}$$

$$\boldsymbol{x}.\boldsymbol{y} = \mathbf{0} \tag{3}$$

where $\boldsymbol{M}$ is a square matrix.

(Throughout this paper we will assume all matrices and vectors are of compatible dimension) i.e we wish to solve a set of linear equations involving variables x, y which fall into two classes of equal size.

The variables must all be non-negative and the two classes containing $\boldsymbol{x}$ and $\boldsymbol{y}$ respectively orthogonal (satisfying (3)). This means that if an element of $\boldsymbol{x}$ is non-zero the corresponding entry of $\boldsymbol{y}$ must be zero.

Such problems arise in a number of contexts some of which we outline below.

The LCP has general applications in Engineering and Economics. For example if two surfaces are able to press against each other then a positive pressure implies a zero gap between them but a positive gap implies a zero pressure. In economics if a resource has a positive marginal value it is scarce and has zero slack (surplus) but if it has positive slack it is not scarce and has zero marginal value.

LCP problems and methods are also of interest since, for example, Linear Programming (LP) models, Quadratic Programming (QP) models and 2-person Non-Zero Sum Games can all be formulated as LCPs.

We outline how this can be done. Good references to the LCP are Murty and Ya (4) and Cottle (2).

<u>Linear Programming</u>

Any LP can be written in the form

$$\text{Maximise } \mathbf{c}'\mathbf{x} \tag{4}$$

$$\text{Subject to } \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \tag{5}$$

$$\boldsymbol{x} \geq \mathbf{0} \tag{6}$$

The dual LP model is

Minimise $\boldsymbol{b}' \boldsymbol{y}$ (7)

Subject to $\boldsymbol{A}' \ \boldsymbol{y} \geq \boldsymbol{c}$ (8)

$\boldsymbol{y} \geq \boldsymbol{0}$ (9)

A necessary and sufficient condition for optimality of both models is that the slack and surplus variables ($\boldsymbol{u}$ and $\boldsymbol{v}$) in (5) and (8) respectively are orthogonal to $\boldsymbol{y}$ and $\boldsymbol{x}$.

i.e $\boldsymbol{y}.\boldsymbol{u} + \boldsymbol{x}.\boldsymbol{v} = 0$ (10)

We can encompass all the conditions in (1), (2) and (3) by setting

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{I} \\ \boldsymbol{I} & -\boldsymbol{A}' \end{bmatrix} \qquad (11)$$

($\boldsymbol{x}$, $\boldsymbol{u}$) and ($\boldsymbol{v}$, $\boldsymbol{y}$) are the orthogonal pairs of vectors.

$$d = \begin{bmatrix} \ \end{bmatrix} \qquad (12)$$

Quadratic Programming

Any QP can be written in the form

Maximise $\boldsymbol{c}'\boldsymbol{x} + \frac{1}{2}\boldsymbol{x}'\boldsymbol{D}\boldsymbol{x}$ (13)

Subject to $\boldsymbol{A}\ \boldsymbol{x} \le \boldsymbol{b}$ (14)

$\boldsymbol{x} \ge 0$ (15)

A necessary and sufficient condition for a local optimum is that the Karush-Kuhn-Tucker conditions are satisfied (see (6)).

We can encompass all these conditions by setting

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{A} & \boldsymbol{D} \\ \boldsymbol{I} & \boldsymbol{A}' \end{bmatrix} \quad (16)$$

where the variables and correspond to those in the LP case above.

2-Person Games

The payoff matrixes for the players in a 2 person Non Zero Sum game can be represented by matrices $\boldsymbol{A}$ and $\boldsymbol{B}$.

It can be shown (see Williams (12)) that we can associate the following polytopes with each of the players.

$$P_A: \quad \sum_j a_{ij} x_j - z_A \le 0 \qquad (17)$$

$$\sum_j x_j = 1 \qquad (18)$$

$$x_j \ge 0 \qquad (19)$$

$$P_B: \quad \sum_i b_{ij} y_i - z_B \le 0 \qquad (20)$$

$$\sum_i y_i = 1 \qquad (21)$$

$$y_i \ge 0 \qquad (22)$$

A major challenge is to find all equilibrium solutions for such games (see e.g (10)). Such solutions correspond to vertex solutions of P_A and P_B which are orthogonal. We can therefore represent it as an LCP by setting

$$\mathbf{M} = \begin{bmatrix} \mathbf{A} & \mathbf{-e} & \mathbf{I} & \mathbf{0} \\ \mathbf{e}' & \mathbf{0}' & \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{B}' & \mathbf{-e} \\ \mathbf{0}' & \mathbf{0}' & \mathbf{e}' & \mathbf{0}' \end{bmatrix} \qquad (23)$$

$\mathbf{e}$ represents a vector of 1s

If $\boldsymbol{u}$ and $\boldsymbol{v}$ are the slack variables in (16) and (20) respectively the orthogonal pairs of the variables are

$\boldsymbol{x}, z_{A,} \boldsymbol{u}, s$ and

$\boldsymbol{v}, t, \boldsymbol{y}, z_B$ respectively where s and t are artificial variables (constrained to be zero) appended to constraints (18) and (21) respectively.

$$\mathbf{d} = \begin{bmatrix} 1 \\ \\ 1 \end{bmatrix} \qquad (23)$$

(Although the above polyhedral description of equilibrium solutions is presented in (12) it was subsequently realised that it had previously been discovered (in a slightly different form) by Kuhn (5)).

Evolutionary Games

This concept is due to Maynard Smith (8). Here a population is considered as ‘playing a game against itself’. Population mixes which emerge through evolution are said to be Evolutionarily Stable. The possible mixes of types in such a population are said to be Evolutionarily Stable States (ESSs). If the game is modelled by a payoff matrix it can be shown (e.g Williams (13)) that ESS solutions are a subset of Equilibrium solutions.

If the payoff matrix is $\boldsymbol{A}$ (a square matrix) then the ‘opponents’ matrix is $\boldsymbol{A}$ (ie itself). Viewed as a non-zero sum game. $\boldsymbol{B}=\boldsymbol{A}$ This makes P_B redundant and we can set

$$\mathrm{M}=\begin{bmatrix} \mathbf{A} & \mathbf{-e} & \mathbf{I} & \mathbf{0} \\ \mathbf{e}' & \mathbf{0} & \mathbf{0}' & \mathbf{0} \end{bmatrix} \qquad (24)$$

The orthogonal pairs of variables are

$$(\boldsymbol{x}, z_A) \; and \; (\boldsymbol{u}, s)$$

$$\mathbf{d} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \qquad (25)$$

Although this makes finding Equilibrium solutions of such games easier (smaller) finding the subset of ESS solutions is much more difficult.

2. Methods of Solving LCPs

One of the earliest methods is due to Lemke (see Lemke and Howson (7)). It is analogous to the Simplex algorithm of LP. It starts with an orthogonal (feasible) solution by setting the x variables to be zero and allowing the y variables to be non-zero. Then, by a process of 'complementary pivoting' new variables are introduced only if their complementary variables can be removed.

Such methods (and their variants) will only find single solutions to LCPs. Although they can be used to find other solutions (by starting from a different initial solution) there is no guarantee that *all* solutions will be enumerated.

The guarantee of knowing that one has *all* the solutions is important in certain applications e.g Engineering and Game Theory.

A good survey of methods can be found in Ferris et al (3)

A number of specialist methods exist for finding Equilibrium solutions of Non-Zero Sum Games. For 2x2 Games these are described by, for example, Thomas (9). One such well known method described there is the swastika method. A general method is described by Winkels (14).

In principle it is always possible to find all solutions by complete enumeration ie taking each possible subset of variables, allowing these to be non-zero, setting their complements to zero, solving and seeing if the resultant solution is feasible.

Another approach is to formulate the orthogonality condition (3) using Integer Programming (IP) and solving the resultant model by an IP method (eg Branch-and-Bound).

Neither of these last two methods (even with refinements) have proved viable given the exponential growth in computation involved.

More recent methods are described by von Stengel (10).

An important class of methods due to Avis and Fukuda (1) simultaneously pivot through the vertices of P_A and P_B searching for orthogonal vertices.

3. The Algorithm

We consider the LCP given by (1), (2) and (3)

1. We 'homogenise' this by putting it in the form:

$$\boldsymbol{y} + \boldsymbol{Mx} - \boldsymbol{d}z = 0 \tag{26}$$

$$z = 1 \tag{27}$$

$$\boldsymbol{x}, \boldsymbol{y} \geq \boldsymbol{0} \tag{28}$$

$$\boldsymbol{x}.\boldsymbol{y} = 0 \tag{29}$$

2. We eliminate each homogenous constraint in turn, by a change of variables.

Consider a general homogenous constraint where the variables are $u_{i,} v_j$ and the co-efficients $p_{i,} q_j, i \in I, j \in J$

$$-\sum_i p_i u_i + \sum_j q_j v_j = 0 \tag{30}$$

$$u_{i,}\ \mathrm{v_j} \geq 0 \tag{31}$$

$$p_{i,}\ \mathrm{q_j} \geq 0 \tag{32}$$

We assume that it is not the case that I or J (but not both) are empty, otherwise the LCP would be infeasible (If both were empty the constraint could be removed immediately).

- $p_i u_i$ are negative quantities which must, in total, 'balance' with the positive quantities $q_j v_j$

We can regard these quantities as the sources and sinks respectively in the following flow diagram.

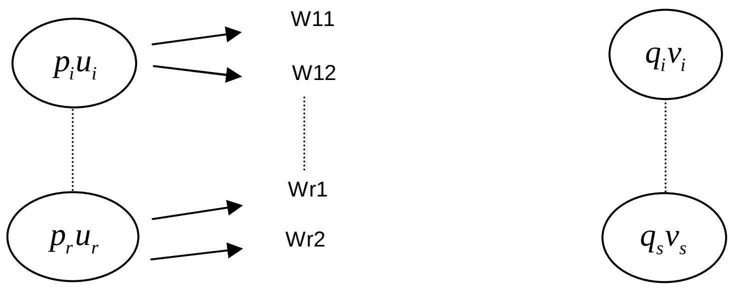

Figure 1

w_{ij} are the new variables which we substitute into the model.

$$u_i \text{ is replaced by } \frac{1}{p_i} \sum_j w_{ij} \tag{33}$$

$$v_j \text{ is replaced by } \frac{1}{q_j} \sum_i w_{ij} \tag{34}$$

Making these substitutions causes the first homogenous constraint to vanish.

3. We repeat this process eliminating each homogenous constraint (in any order).

4. Finally the system (26) (after, if necessary, scaling the variables) is reduced to a convexity constraint, resulting from (27), in some of the final set of (new) variables. Setting any of these variables to 1 corresponds to a basic solution of the system (26). The other variables (with zero entries in the convexity row) correspond to rays of the polytope of (26).

5. However we need to restrict ourselves to a subset of the solutions which are (a) extreme (vertices or extreme rays) and (b) satisfy the orthogonality condition (29). These properties allow us to restrict the new variables created at both the final and intermediate stages of the constraint eliminations.

The substitution of variables defined by figure 1 does not need to be carried out explicitly. It is implied by the matrix representing the elimination of a constraint from (26).

(26) can be written

$$\boldsymbol{A} \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{x} \\ z \end{bmatrix} = \begin{bmatrix} \bar{} \\ 1 \end{bmatrix} \tag{35}$$

where

$$A = \begin{bmatrix} I & M & \\ 0' & 0' & 1 \end{bmatrix} \qquad (36)$$

In order to eliminate a constraint from (26) we add together pairs of columns of the matrix , which have opposite sign in the constraint, in non negative multiples so as to create zeros. This procedure is best illustrated by a numerical example, given in the next section. The procedure is the dual of Fourier-Motzkin Elimination as described in Williams (11).

At each stage in the process, after the constraints have been eliminated, the elimination to date can be represented as the postmultiplication of the matrix A by a matrix T consisting of zeros and positive entries. Each column of T represents the non negative multiples in which the columns of A must be added to eliminate the rows to date.

Although we are implicitly transforming to new variables after each elimination we need not *explicitly* state the new variables.

It is shown in (11) that, after r rows have been eliminated, any column of T **,** consisting of more than r+1 non zero entries, is redundant and can be removed . This is efficiently implemented by, after every two extra constraints are eliminated, removing columns of the current matrix T which depend on more than 3 of the columns of the matrix before last.

The columns of the final transformation matrix T represent extreme solutions (rays and vertices) of the polytope defined by (26))

However, since we want solutions to an LCP we can also remove non self-orthogonal columns of T i.e columns whose entries corresponding to y are not orthogonal to those corresponding to x . The columns of any intermediate transformation matrix T which are non self-orthogonal will give rise to non self-orthogonal columns in the final transformation matrix and may, therefore be removed at intermediate stages.

Although we may still obtain a ‘combinatorial explosion’ in the number of columns of T (since an LCP can have an exponential number of basic solutions) we avoid an unnecessary explosion.

The details of the procedure are best illustrated by a numerical example in the next section.

4. A Numerical Example

$$\boldsymbol{d} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Writing in homogenised form we have:

$$\boldsymbol{A}_1 \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

where $\boldsymbol{A}_1 = \begin{bmatrix} 1 & 0 & 0 & 2 & 3 & -1 & -1 \\ 0 & 1 & 0 & 1 & -2 & 0 & -1 \\ 0 & 0 & 1 & 1 & 1 & 1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

The elimination of the first constraint is effected by successively carrying out elementary column operations A_1

To do this pairs of columns of opposite sign are added in non-negative multiples so as to eliminate the first coefficient in the resultant column.

In order to carry of this transformation we postmultiply A_1 by

$$\boldsymbol{T}_1 = \left[\begin{array}{cccccccc} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 2 & 3 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{array}\right]$$

We have partitioned $\boldsymbol{T}_1$ in order to separate multipliers corresponding to $\boldsymbol{y}, \boldsymbol{x}$ and z

Carrying out the post multiplication transforms the co-efficient matrix A to:

$$\boldsymbol{A_2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & -2 & -1 & -1 & -5 \\ 0 & 1 & 1 & 3 & 4 & -1 & -1 & -2 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix}$$

Similarly in order to eliminate the second row of the above matrix we postmultiply by:

$$\boldsymbol{T_2} = \begin{bmatrix} 0 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 5 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Simultaneously we can update the transformation matrix to create $\boldsymbol{T_2}' = \boldsymbol{T_1 T_2}$
This gives

$$\boldsymbol{T_2}' = \left[\begin{array}{cccccccccc} 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 5 & 0 \\ 1 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 2 & 0 & 1 & 1 & 2 & 0 & 5 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 3 & 7 & 0 & 2 & 0 & 2 & 0 & 2 \\ \hline 0 & 0 & 0 & 0 & 1 & 1 & 2 & 2 & 3 & 3 \end{array}\right]$$

It is shown in (11) that we can remove all columns from $\boldsymbol{T}_2'$, which contain more than 3 non-zero entries, as being non-extreme. In general after r rows of the co-efficient matrix are eliminated we can discard any column of the transformation matrix which has more than $r+1$ non zero entries.

More generally we can discard any column where the index set of non-zero entries is a superset of the corresponding set for another column. If two columns have identical such sets both can be discarded.

In our example this enables us to remove columns 6 and 10 from $\boldsymbol{T_2}'$, (and the corresponding columns from $\boldsymbol{A_3}$).

While this helps to reduce the explosive growth in columns of the transformation matrix it may still grow exponentially. It can be shown that the columns of the final transformation matrix correspond to all vertices and extreme rays of the polytope represented by the original system.

However we need only concern ourselves with a subset of these columns. The columns of the final transformation matrix represent *all* solutions to the LCP so long as they are *self-orthogonal* i.e the component corresponding to the y variables is orthogonal to those corresponding the x variables. In order for this to be the case the columns of the intermediate transformation matrixes, from which the final columns arise, must also be orthogonal.

In the example this enables us also to discard columns 3 and 9 of $\boldsymbol{T_2}'$ (and the corresponding columns of A_3) giving

$$\boldsymbol{A_3}' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 10 & -1 & -1 & 2 \\ 0 & 0 & 0 & 1 & 2 & 2 \end{bmatrix}$$

$$\boldsymbol{T_2}'' = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 2 & 0 & 1 & 2 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 7 & 0 & 0 & 2 \\ \hline 0 & 0 & 0 & 1 & 2 & 2 \end{bmatrix}$$

We can now eliminate the third row of $\boldsymbol{A_3}'$ by postmultiplying by the following transformation matrix.

$$\boldsymbol{T}_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 10 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 10 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The updated co-efficient and transformation matrices are:

$$\boldsymbol{A_4} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 10 & 4 & 2 & 2 & 20 & 6 \end{bmatrix}$$

$$\boldsymbol{T_3}' = \begin{bmatrix} 1 & 2 & 10 & 2 & 0 & 1 & 0 & 0 \\ 1 & 1 & 10 & 2 & 1 & 1 & 10 & 2 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 2 & 2 & 1 & 1 & 12 & 4 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 7 & 2 & 0 & 1 & 7 & 2 \\ \hline 1 & 1 & 10 & 4 & 2 & 2 & 20 & 6 \end{bmatrix}$$

Columns 3,4,6 and 7 of $\boldsymbol{T_3}'$ (and $\boldsymbol{A_4}$) can be discarded as depending on more than 4 of the original columns. The remaining columns are all self-orthogonal giving:

$$\boldsymbol{A_4} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 2 & 6 \end{bmatrix}$$

$$\boldsymbol{T_3}'' = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 1 & 1 & 1 & 2 \\ 1 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 4 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 \\ \hline 1 & 1 & 2 & 6 \end{bmatrix}$$

In order to satisfy the condition z=1 we scale the 3rd and 4th columns by 2 and 6 respectively to give *all* the solutions to the original LCP.

$$y_1 = y_2 = y_3 = 1,\ x_1 = x_2 = x_3 = 0$$

$$y_1 = 2. y_2 = 1, y_3 = 0,\ x_1 = x_2 = 0, x_3 = 1$$

$$y_1 = 0,\ y_2 = y_3 = \frac{1}{2},\ x_1 = \frac{1}{2}, x_2 = x_3 = 0$$

$$y_1 = 0,\ y_2 = \frac{1}{3}, y_3 = 0,\ x_1 = \frac{2}{3}, x_2 = 0, x_3 = \frac{1}{3}$$

Should convex combinations of self-orthogonal vertices, together with non-negative combinations of extreme ray solutions, be self orthogonal than they will also give solutions to the LCP.

References

1. D.Avis and Fukuda K, A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra, *Disc. Comp. Geometry* 8 (1992) 295-313

2. Richard Cottle, *The Linear Complementarity Problem*, Academic Press, Boston, Mass. USA 1992

3. M.C Ferris, Mangasarian, O.L and Pong J.S (Eds) *Complementarity : Applications, Algorithms and Extensions* Vol.50 Applied Optimization, Dordrecht, The Netherlands, Kluwer, 2001

4. Katta G. Murty and Feng-Tien Ya, *Linear Complementarity Linear and NonLinear Programming*, Internet Edition, University of Michigan, Ann Arbor, USA, 1999

5. H.W Kuhn, An Algorithm for Equilibrium Points in Bi Matrix Games, *Proceedings of the National Academy of Sciences,* USA , 47 (1961) 1657-1662

6. H.W Kuhn and A.W Tucker, Non Linear Programming. J Neyman (Ed), *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, Berkeley, CA USA 1951.

7. C.E Lemke and J.T Howson, Equilibrium Points of Bimatrix Games, SIAM *Journal of Applied Mathematics* 12 (1964) 413-423

8. J. Maynard Smith, *Evolution and the Theory of Games*, Cambridge, UK, 1982.

9. L.C Thomas, *Games Theory and Applications*, Ellis Horwood, Chichester, UK, 1984

10. B Von Stengel, Equilibrium Computation for Two-Player Games in Strategic and Extensive Form, in N Nisan et al (Eds) *Algorithmic Game Theory*, Cambridge University Press, Cambridge 2007

11. H.P Williams, Fourier's Method of Linear Programming and its Dual, *The American Mathematical Monthly*, 93 (1986) 681-695

12. H.P Williams, A method of Finding All Equilibrium Solutions of a 2-Person Matrix Game, Working Paper OR22, Faculty of Mathematical Studies, University of Southampton, UK (1989)

13. H.P Williams, Evolution, Games Theory and Polyhedra, *Journal of Mathematical Biology*, 25 (1987), 393-409

14. H.M Winkels, An Algorithm to Determine All Equilibrium Points of a Bimatrix Game, Moeschlin and Pallescke (Eds) *Game Theory and Related Topics*, North Holland, Amsterdam, Holland 1979.