A survey on pivot rules for linear programming

Report 91-99

T. Terlaky
S. Zhang

TU Delft

Technische Universiteit Delft
Delft University of Technology

Faculteit der Technische Wiskunde en Informatica
Faculty of Technical Mathematics and Informatics

ISSN 0922-5641

DELFT UNIVERSITY OF TECHNOLOGY

REPORT 91–99

A SURVEY ON PIVOT RULES FOR LINEAR PROGRAMMING

T. Terlaky and S. Zhang

ISSN 0922–5641

Reports of the Faculty of Technical Mathematics and Informatics no. 91–99

Delft, 1991

Tamás TERLAKY,
Faculty of Technical Mathematics and Informatics, Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands. e-mail: wioro11@dutiosa.tudelft.nl

Shuzhong ZHANG,
Department of Econometrics, University of Groningen, P.O. Box 800, 9700 AV Groningen, The Netherlands. e-mail: szhang@rugr86.rug.nl

The first author is on leave from the Eötvös University, Budapest, and partially supported by OTKA no. 2115.

and Informatics, Delft University of Technology, The Netherlands.

Abstract

The purpose of this paper is to survey the various pivot rules of the simplex method or its variants that have been developed in the last two decades, starting from the appearance of the minimal index rule of Bland. We are mainly concerned with the finiteness property of simplex type pivot rules. There are some other important topics in linear programming, e.g. complexity theory or implementations, that are not included in the scope of this paper. We do not discuss ellipsoid methods nor interior point methods. Well known classical results concerning the simplex method are also not particularly discussed in this survey, but the connection between the new methods and the classical ones are discussed if there is any.

In this paper we discuss three classes of recently developed pivot rules for linear programming. The first class (the largest one) of the pivot rules we discuss is the class of essentially combinatorial pivot rules. Namely these rules only use labeling and signs of the variables. These include minimal index type rules and recursive rules. The second class contains those pivot rules which can be considered in fact as variants or generalizations or specializations of Lemke's method, and so closely related to parametric programming. The last class of pivot rules discussed in this paper has the common feature that these rules all have close connections to certain interior point methods. Finally we mention some open problems for further study.

Key Words: *Linear programming, simplex method, pivot rules, cycling, recursion, minimal index rule, parametric programming.*

1 Introduction

In this paper, we consider the following linear programming problems in standard form:

$$(P) \qquad \min\{c^T x : Ax = b,\ x \geq 0\},$$

$$(D) \qquad \max\{b^T y : A^T y \leq c\},$$

where A is an $m \times n$ matrix, and b, c, x, y are m and n dimensional vectors respectively. It is well known that linear programming problem (P) is called the *primal* problem and (D) is called the *dual* problem.

The main purpose of this paper is to overview the pivot rules of the simplex method (or its variants) for solving linear programming problems either in the form (P) or in the form (D).

Linear programming has been one of the most turbulent area of applied mathematics in the last forty years. Among many recent approaches, Dantzig's (cf. [16]) simplex method (together with its implementations) still seems to be the most efficient algorithm for the great majority of practical problems (most of the practical problems are not "very large" when interior point methods became more efficient). In solving practical problems it turned out that in most cases the required number of pivot steps is a linear (at most quadratic) function of the number of variables. This observation about the practical efficiency of the simplex method [28] is theoretically justified by proving its polynomial behavior in the expected number of pivot steps [9], [2], [72] in average. On the other hand, for most of the simplex variants there are examples on which the number of pivot steps might be exponential. For such exponential examples see e.g. [3], [40], [49], [50], [56] and [61]. To find a *polynomial pivot rule* for linear programming, meaning that under which the number of pivot steps will always be bounded by a polynomial function of the number of variables, or to prove that such pivot rules do not exist seems to be very hard. Actually, to clarify if there exists such a polynomial time simplex algorithm together the so-called "Hirsch conjecture" [41] is the most challenging open problem in linear programming and in polyhedral theory.

One reason for the efficiency of the simplex method might be that the simplex method is very flexible, i.e. there are plenty of possibilities to select the pivot element during the procedure (even in case of nondegeneracy). Therefore many simplex variants have been developed.

In the recent years, however, most attention and research in linear programming have been devoted first to the ellipsoid method developed by Khachian [35] and later (even more intensively) to the interior point methods initiated by Karmarkar [34]. Recent papers concerning simplex pivot rules have not been receiving much attention even among researchers in the field of linear programming. Moreover, a pretty large part of the work was presented in terms of oriented matroid programming and frequently without explicitly specializing the new pivot rules for the linear programming case, and so remained unknown for researchers who are not working in that field. Some of the results were obtained as a side result (extreme case) of some interior point methods, and also remained unknown for the people working on the simplex method. Our aim here is to summarize the pivot rules that have appeared in the last two decades. Since most of the early classical results concerning simplex pivot rules are well known, we shall start our discussion from the middle seventies. The earlier results are mentioned only when it is necessary to relate them to some new algorithms, or to assist the understanding.

The paper is organized as follows. At the end of this section we will present some standard notations which we need to use throughout the paper. In Section 2 we are going to discuss simplex type algorithms that are related to or presented exclusively in oriented matroid programming. Due to their combinatorial character all of these methods are finite even in the case of degeneracy. This class includes Bland's [6], [7] minimal index rule (which is related to Murty's Bard type scheme and [48] Tucker's consistent labelling [73]), Bland's recursive rule [6], [7], the Edmonds–Fukuda rule [18] and its variants [14], [79], [80], [75], Jensen's general relaxed recursion [32], the finite criss–cross method [66], [67], [74], [10] and variants [19], [76] and finally Todd's rule [69], [70] which was presented for the linear complementarity problem of oriented matroids. All these methods, except for the criss–cross method and Jensen's recursion are simplex variants, i.e. they preserve the feasibility of the basis (the situation with Edmonds – Fukuda rule is not clear, it is a simplex method with Clausen's restriction, but in general its feasibility is still open). But, in contrast, in the case of oriented matroid programming only Todd's rule preserves feasibility of the basis and the finiteness at the same time and therefore yields a finite simplex algorithm.

Criss–cross methods gave a positive answer to the long standing question: if there exists a finite (may be infeasible) pivot method for linear programming which solves the LP problem in one phase. There were several attempts to relax the feasibility requirement in the simplex method. The first step was made by Dantzig [16] by presenting his parametric self dual algorithm, which can be interpreted as Lemke's [44] algorithm for the corresponding linear complementarity problem [45]. The first criss–cross type method was designed by Zionts [81]. The finite criss–cross algorithm was presented independently by Chang [10], Terlaky [66] and Wang [74]. Terlaky presented this algorithm for LP. In his unpublished report [10] Chang presented this algorithm for positive semidefinite linear complementarity problems, while Wang [74] presented for oriented matroid programming. Criss–cross methods, like the parametric self–dual method [16], can be initiated with any basic solution, and will finally find the optimal solution if it exists. Unfortunately, the finiteness of Zionts' criss–cross method is not clear in general, while Terlaky's criss–cross method is finite. But due to its purely combinatorial feature it is practically inefficient. Other combinatorial type algorithms have arisen recently. Jensen's [32] general recursive scheme is actually a general frame for finite pivot algorithms.

An interesting (and efficient) application of the minimal index rule and the criss–cross method is presented by Avis and Fukuda [4]. By reversing these rules an algorithm is derived to enumerate all the feasible bases (vertices of a polyhedra) and all the bases without extra storage requirement.

In Section 3, we present the second class of pivot rules. These pivot rules have a common feature, namely all of them can be interpreted as a parametric or iteratively reparametrized simplex algorithm. These algorithms may cycle for degenerate problems if no anti-cycling techniques are incorporated. The parametric programming procedure related to these algorithms is often referred as the shadow vertex algorithm [25], [9] and [49]. This variant of the simplex method is also exponential in the worst case [27]. But under some reasonable probabilistic assumptions its expected number of pivot steps is polynomial [9]. The algorithms reviewed in this section are the Monotonic Build-Up simplex algorithm (MBU) [1], the Exterior Point Simplex Algorithm (EPSA) [57], [58], [59] which can also be considered as a special case of MBU. The relations of these algorithms to Dantzig's [16] self–dual parametric simplex algorithm are also discussed. At the end of Section 3, we mention the Transition Node Pivoting (TNP) algorithm [26], [24].

In Section 4, we discuss the third group of pivot rules in this paper. In that group, the simplex pivot algorithms were derived from or inspired by interior point methods. We will include the pivot rule of Roos [60] which is related to Karmarkar's [34] potential function, Todd's algorithm [71] derived from a projective interior point method as the limit, and the dual-interior-primal-simplex method of Tamura *et al.* [65] as a generalization of Murty's [51], [52] gravitational method.

To conclude, some remarks and open problems will be mentioned at the end of the paper. There are rich research results concerning pivot rules for specially structured linear programming problems, like network linear programming, assignment problems, etc.. Due to their independent interest, we are not going to include these results in this survey.

To unify the presentation, the following notations will be used in this paper. Matrices will be denoted by capital letters, vectors by lower case letters and their coordinates by the same letters with subscript (e.g. if $x \in R^n$, then $x = (x_1, \ldots, x_j, \ldots, x_n)$). Given a linear programming problem, a *basis* B is a maximal subset of indices $\{1, 2, ..., n\}$ such that the corresponding column vectors of matrix A (i.e. $\{a_i,\ i \in B\}$) are independent. We denote this submatrix by A_B. The *basic tableau* $T(B)$ corresponding to the basis B is as follows:

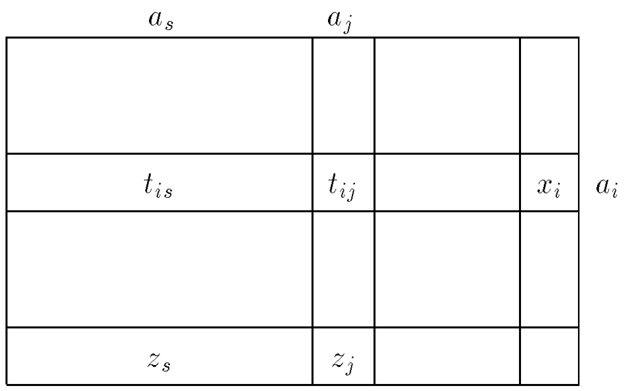

Figure 1.

Here t_{ij} denotes the coefficient of the basic vector a_i in the basic representation of the vector a_j. Note, that in our interpretation index pair ij of the tableau elements refer to basic vector a_i and nonbasic vector a_j, not to the geographical position of element t_{ij} in the tableau. x_i denotes the i-th coordinate of the *basic solution*. If there is no confusion we will also call x_i a *basic variable* for $i \in B$. We call $N := \{1, 2, ..., n\} \setminus B$ the *nonbasic* and x_j a *nonbasic variable* for $j \in N$. Denote $y^T = c_B^T A_B^{-1}$ the corresponding dual solution and z_j represents the j-th coordinate of the *reduced cost* ($z^T = (z_1, \ldots z_n) = c^T - y^T A$ is the reduced cost vector). We denote ${t^{(r)}}^T = (t_{r1}, \ldots t_{rn})$ the row in the tableau $T(B)$ corresponding to the basic index r.

Let us also define a vector $t_{(k)}$ for all $k \in N$ as follows:

$$t_{(k)} = (t_{(k)j})_{j=1}^n \quad \text{with} \quad t_{(k)j} = \begin{cases} t_{kj} & \text{if } j \in B, \\ -1 & \text{if } j = k, \\ 0 & \text{otherwise.} \end{cases}$$

A basis (tableau) is called *primal feasible* if $x_i \geq 0$ for all $i \in B$ and is called *dual feasible* if $z_j \geq 0$ for all $j \in N$. If a basic variable x_i leaves the basis and a nonbasic variable x_j enters, then this operation (transforming the tableau) is called *pivoting* on position (i, j) and element t_{ij} is referred as *pivot element.*

If all the basic variables are nonzero and all the nonbasic variables have nonzero reduced cost, then the basis is *nondegenerate* else the basis is *degenerate.* The basis is called *primal degenerate* if at least one basic variable is zero, and *dual degenerate* if at least one nonbasic variable has zero reduced cost.

2 Combinatorial and Finite Pivot rules

In this section we will review those recently developed pivot rules which are concerned only with the "infeasibility" of the basic variables or the nonbasic variables with negative reduced costs (dual infeasible). We call these type of pivot rules *combinatorial pivot rules.* In other words, combinatorial pivot rules take care of the signs of the elements in the last row and the last column of the tableau. As a counter example, the well known Dantzig's pivot rule always selects the nonbasic variable with the most negative reduced cost, and so Dantzig's rule concerns not only with the signs of variables but also their magnitudes. This means that Dantzig's rule is not combinatorial. Such rules will not be discussed in this section.

In fact, most combinatorial pivot rules are originally developed in the context of an abstracted form of linear programming, i.e. oriented matroid programming. The theory of oriented matroid programming was established by Bland [7], Bland and Las Vergnas [8], Folkman and Lawrence [17] and Fukuda [18]. In this section we will mainly focus on the presentation of these results in linear programming terminologies.

We start our discussion with the well known pivot rules of Bland [6]. Actually, in Bland's paper [6], two pivot methods are proposed. Those rules start from a feasible basis and keep the primal feasibility. Bland's first rule is known as the "minimal index rule" is as follows:

We start with a primal feasible tableau $T(B)$, and fix an arbitrary ordering (indices) of the variables.

Bland's Minimal Index Rule

Step 1 (Entering variable selection) Choose the nonbasic variable x_s whose index is minimal among the variables with negative reduced cost.

If there is no such variable, $T(B)$ is optimal, stop.

Step 2 (Leaving variable selection) Choose a variable, say x_r, to leave the basis such that after this pivot operation the basis remains feasible and index r is minimal among the possibilities.

If such a variable does not exist, the problem is unbounded.

Remark that the procedure to maintain the feasibility of the basis is normally called the *minimum-ratio test* in the simplex method.

Bland's second rule is less well known. In fact, Bland's second rule proposes an *inductive* (or *recursive*) procedure. The idea of that procedure is based on the elementary observation that a primal feasible basis would be optimal if the dual infeasible variables would be deleted. Bland's recursive rule can be described as follows.

We start with a primal feasible tableau $T(B)$.

Bland's Recursive Rule

Step 0 Fix all dual infeasible variable to zero level, therefore a subproblem is solved optimally.

Step 1 To get a subproblem with size one larger release one fixed variable (with negative reduced cost), preserving primal feasibility pivot this variable in, and fix it as a basis variable. The size of this subsidiary problem is again the same as we have solved previously. Solve this subsidiary problem recursively.

If it is unbounded, then the original problem is unbounded.

Step 2 If we have an optimal solution of the subproblem, release all fixed variables.

If there is no dual infeasible variable then the actual solution is optimal. Else repeat the procedure.

By the lemma provided below we shall see that for successive degenerate pivot steps, the minimal index rule works on the simplex tableau also in a recursive way. We will further show that a large class of combinatorial pivot rules are recursive for degenerate pivots and therefore finite. We note here that Murty's bard type scheme [48] and Tucker's consistent labelling algorithm [73] (for maximum flow problem) were proposed independently based on similar idea. First we will present the following important lemma for proving the finiteness of most anti-cycling pivot rules.

Lemma 1 *Let $T(B)$ and $T(B')$ be two different tableaus. For the tableau $T(B)$ and $T(B')$ consider the vectors $t_{(k)}$ for any $k \notin B$ and $t^{(r)}$ for any $r \in B'$.*
Then

$$
\begin{aligned}
(z-z')^T(x-x') &= 0,\\
(z-z')^T t_{(k)} &= 0,\\
(x-x')^T t^{(r)} &= 0,\\
t_{(k)}^T t^{(r)} &= 0.
\end{aligned}
$$

Proof:
Cf. [6], [7], [66] and [67].

□

Note, that if we extend vector x and z with the objective value, and vectors $t_{(k)}$ with the reduced cost z_k, vectors $t^{(r)}$ with x_r (denote the extended vectors by overline), then

$$\overline{x}^T\overline{z}' = 0, \quad \overline{t}_{(k)}^T\overline{z}' = 0, \quad \overline{x}^T\overline{t}'^{(r)} = 0, \quad \overline{t}_{(k)}^T\overline{t}'^{(r)} = 0.$$

Using Lemma 1 we are able to show the following fundamental theorem.

Theorem 1 *For any simplex pivot rule (in the primal form) satisfying one of the following two properties:*

1) *(*nondegeneracy*) either a pivot is nondegenerate; or*

2) *(*recursive*) any successive pivot steps are recursive,*

then the pivot rule is finite.

Remark. A recursive pivot rule (in the primal form) introduces a new variable entering the basis if all the previously introduced variables are in a feasible basis or have nonnegative reduced costs.

Proof:

We need only to prove that no cycling will occur in degenerate pivot steps. Suppose to the contrary, that cycling occurs. We delete the columns and the rows corresponding to those variables do not change their status from basic to nonbasic or the other way around. It is clear that the cycle remains in the reduced tableaus.

Now, let x_k denote the last involved nonbasic variable in the cycle that becomes basic, and the corresponding tableau in which x_k is entering the basis be T_1. Since the pivot steps are recursive, therefore we conclude that all the other nonbasic variables in T_1 have nonnegative reduced costs. By definition of a recursive procedure, after that pivot step we should first solve the subproblem of deleting the row and the column corresponding the variable x_k. If the subproblem is solved optimally, then with x_k a degenerate basic variable the problem is also solved optimally. This contradicts to the fact that a cycle exists. If the subproblem is unbounded whereas by adding the basic variable x_k it is not, this implies that in the last tableau the column of entering basis variable x_r has only one positive element in the cross of the row corresponding to x_k. Taking this column and by Lemma 1, the corresponding vector $\overline{t}_{(r)}$ should be orthogonal to the last row $\overline{z}'$ of the tableau T_1. However, for these two vectors each coordinate does not have the same sign. In particular, the coordinate corresponding to the variable x_k has opposite signs in $\overline{t}_{(k)}$ and in $\overline{z}'$. This again results in a contradiction (in this case $\overline{t}_{(k)}^T\overline{z}'$ is negative instead of zero) to Lemma 1. Therefore the theorem is proved.

$\square$

It is interesting to see that the following pivot rules satisfy the properties discussed in the above theorem.

1. Bland's minimal index rule;

2. The Last In First Out rule, i.e. select the nonbasic variable which has negative reduced cost and which has been nonbasic most recently, and select the variable to leave the basis according to the minimum-ratio test and if there are more than one candidates select one which has been basic most recently;

3. The Most Often Selected rule, i.e. select variables to enter and to leave the basis with the most often selected historical record.

These rules were discussed in Zhang [79]. Moreover, the so-called *well-preserved ordering* property discussed in Zhang's paper is essentially the same as the recursive property in Theorem 1.

Interestingly, in the combinatorial abstraction of linear programming — oriented matroid programming, *nondegenerate cycling* can take place. Examples of this type can be found in Fukuda [18], Jensen [32] and Ziegler [80]. Note that the minimal index pivot rule is recursive only for degenerate pivot steps. We may call this type of rules *partial recursive.* So the partial recursive rules may still cycle in the oriented matroid programming case. Attentions were paid to find finite pivot rules in that case. One of the early results in that respect is the the pivot rule suggested by Edmonds and Fukuda [18]. We state Edmonds–Fukuda rule here in terms of linear programming. Edmonds–Fukuda's rule is a recursive one (comparing to the partial recursive ones discussed in Theorem 1).

We start with a primal feasible tableau $T(B)$.

Edmonds–Fukuda Rule

Step 0 From $T(B)$ we construct a list $L := (l_1, ..., l_t)$ where $\{l_1, ..., l_t\}$ is the index set of nonbasic variables in T_B.

Step 1 (Entering variable selection) Choose the nonbasic variable x_{l_k} with negative reduced cost and whose index is the rightmost in L.

If there is no such variable, $T(B)$ is optimal, stop.

Step 2 (Leaving variable selection) Choose a variable, say x_r, to leave the basis such that after this pivot operation the basic variables in the set $\{x_i : i \in B \setminus \{l_1, ..., l_k\}\}$ remain primal feasible. (Partial minimum–ratio test)

If such a variable does not exist, the problem is unbounded.

Step 3 (Update) Pivot on (r, l_k) and let

$B := B \setminus \{r\} \cup \{l_k\}$,

$N := J \setminus B$,

$L := (l_1, ..., l_k, l'_{k+1}, ..., l'_s)$

where $\{l'_{k+1}, ..., l'_s\} := \{l_{k+1}, ..., l_t, r\} \cap N$, (here this subset can be reordered)

$t := s$, $l_j := l'_j$, $j = k+1, ..., t$.

Go to **Step 1**.

The finiteness proof of this pivot rule in linear programming case is similar to Theorem 1. We will leave it to the reader.

An interesting question arises here. Since the primal "feasibility" is only partially maintained in the above algorithm, is it enough to guarantee that all tableaus produced are feasible? The answer is negative for the oriented matroid programming case (see Jensen [32] and Ziegler [80]) and remains unknown for linear programming case. However, Clausen [14] proved that the algorithm will maintain feasibility in the linear programming case if a restriction (fixing the ordering) is added to the algorithm. Note that there are flexibilities in the Edmonds–Fukuda rule for updating the list L at Step 3. Due to Clausen's result, we will call the following restricted rule the modified Edmonds–Fukuda rule.

The Modified Edmonds–Fukuda Rule

Steps 0, 1, 2 The same as in the Edmonds–Fukuda rule.

Step 3' (Update) Pivot on (r, l_k) and let

$B := B \setminus \{r\} \cup \{l_k\}$,

$N := J \setminus B$,

$L := (l_1, ..., l_k, l'_{k+1}, ..., l'_s)$

where $(l'_{k+1}, ..., l'_{s-1}) := (l_{k+1}, ..., l_t) \cap N$ (remain the same order) and

$l'_s := r$,

$t := s$, $l_j := l'_j$, $j = k+1, ..., t$.

It is clear that the Modified Edmonds–Fukuda rule determines a unique pivot path if no degeneracy occurs. Actually, in the nondegenerate case it has the same pivot path as the Last In First Out rule, and in case of degeneracy it has more flexibility.

Concerning recursive type pivot rules, another interesting and extremely simple pivot rule is the so-called *finite criss–cross* rule developed independently by Terlaky [66], [67], Wang [74], and Chang [10]. The finite criss–cross rule is based on the following two observations. First, since one is only interested in finding the optimal tableau (or basis) of a linear program, it is not always necessary to maintain either primal or dual feasibilities as the primal or the dual simplex method does. Hopefully there can even be some "short-cut" if we allow primal and dual infeasibilities. Second, the recursive scheme can be carried out more thoroughly if we do not insist on maintaining the primal feasibility (in the primal simplex form) during the pivot iterations.

Originally a criss–cross method was proposed by Zionts [81] for linear programming. However, there is no guarantee of finiteness in Zionts' criss–cross method. The finite criss–cross method [66] was developed based on the minimal–index recursive technique. The method is surprisingly simple and requires neither primal nor dual feasible basis to start with. The method also remains finite for oriented matroid programming (Terlaky [67] and Wang [74]). Now we present the method as follows.

Finite Criss–cross Rule

Step 0 Let $T(B)$ be an arbitrary tableau (can be neither primal nor dual feasible);

Step 1 Let

$r := \min\{i :$ either x_i is primal infeasible or x_i has a negative reduced cost$\}$;

If there is nor r, then $T(B)$ is optimal, stop.

Step 2 If x_r is primal infeasible, let $s := \min\{l : t_{rl} < 0\}$.

Go to **Step 3.1** (if there is no s, the problem is infeasible, stop).

If x_r has a negative reduced cost, let $s := \min\{l : t_{lr} > 0\}$.

Go to **Step 3.2** (if there is no s, the problem is dual infeasible, stop);

Step 3.1 Pivot on (r, s). Go to **Step 1**.

Step 3.2 Pivot on (s, r). Go to **Step 1**.

The finiteness proof of this pivot rule (in linear programming case) is similar to the proof of Theorem 1. For details, one is referred to Terlaky [66].

Since the finite criss–cross method presented above, as well as Bland's minimal index rule, is completely combinatorial and so it may not be efficient in practice. An exponential example for the criss–cross pivot rule is given by Roos [61]. The construction of that example is closely related to the Avis-Chvátal example [3] for the minimal index pivot rule. A new and interesting finiteness proof for the criss–cross method is given by Fukuda and Matsui [19]. Fukuda and Matsui pointed out that there can be more flexibilities in the criss–cross method without loosing finiteness. We will come back to this point later.

We shall remark here that these combinatorial pivot rules can be used to solve linear complementarity problems as well. For papers discussing pivot rules for complementarity problem, among others, cf. Chang and Cottle [11] Fukuda and Terlaky [21], Terlaky [68] [22], Chang [10], Klafszky and Terlaky [38] [39], Fukuda and Namiki [20], Cottle [15] and den Hertog, Roos and Terlaky [30]. Particularly, a quadratic programming problem can be converted to a linear complementarity problem.

Since minimal index type methods need the indices of variables as an indication of priorities of subproblems to be solved recursively, it is profitable to find a "good" ordering of indices before hand. We can treat such an ordering procedure as *pre-conditioning*. Pan [55] observed that a facet of a polytope is likely to contain the optimal vertex if the angle between its tangent direction and the objective direction is small. Based on this observation, he suggested an ordering of variable indices according to the cosine of the constraints' tangent direction and the objective direction.

Now we consider again the criss–cross method. We notice that the criss–cross method is a *completely* recursive procedure, and so it can be linked to the Edmonds–Fukuda rule which is also completely recursive. This fact is observed by Fukuda and Matsui [19]. They remarked that in order to guarantee the finiteness in the criss–cross rule, only a partial ordering of variables is sufficient. At each pivot step we select a variable which is either primal infeasible or has a negative reduced cost according to this partial ordering (from low to high), and then we select the pivot element again according to this partial ordering (refer to the criss–cross rule presented above). Denote x_t to be the higher one according to the partial ordering among entering and leaving variables. In the next iteration, for those variables higher than x_t we keep the same ordering, and for those variables lower than x_t we let them equal and lower than x_t in the new partial ordering. This gives more flexibility in the criss–cross method and finiteness still can be proved. In fact Fukuda and Matsui [19] and Namiki and Matsui [54] explored more flexibility of the criss–cross method.

To ease the presentation of Jensen's rule the following convention is used. A tableau for an LP will be called below *terminal* if it shows either primal infeasibility, dual infeasibility or it is optimal. A framework of recursive pivot rules for oriented matroid programming was presented by Jensen [32] as follows.

Jensen's General Recursion

Step 0 Observe, that a small problem, with one or two variables is easy to solve.

Let $T(B)$ be an arbitrary tableau (can be neither primal nor dual feasible);

Step 1 Select a subproblem SLP (subtableau) for which the present tableau is terminal.

Step 2 Select a variable (say x_s) not included yet in the subproblem SLP and add it to this subproblem. Denote this new subproblem by SLPs (The variable might be in or out of the basis.)

If there is no such variable, the problem is solved, stop.

Step 3.1 If the present tableau is terminal for the new subproblem SLPs, then let SLP:=SLPs and Go to **Step 2**.

Step 3.2 If the tableau is not terminal for SLPs, then make a pivot, change the position of the new variable x_s and fix it in the new position.

Step 4 The subproblem SLP' obtained so (discarding again x_s) has the same number of variables as it was solved in Step 1. Call recursively this algorithm to solve this problem. The terminal tableau for SLP' is also terminal for SLPs. Let SLP:=SLPs and Go to **Step 2**.

The finiteness and validity of Jensen's general recursion can be proved the same way as the finiteness of the criss–cross method.

Jensen's work is very general. Many recursive type pivot rules can be viewed as special variants of Jensen's framework for linear programming. We should observe that the recursive order for the basic and nonbasic variables can be independent. For instance, it is pointed out at [79] that in primal simplex method for linear programming the minimal index rule for selecting entering basis variable together with the last-in-first-out rule for selecting leaving basis variable will guarantee no cycling.

Up to now, we have discussed recursive type combinatorial pivot rules for linear programming. To conclude this section, we will finally discuss Todd's finite and feasible pivot rule [70]. So far we can see that most finite pivot rules are of recursive type. These rules either do not keep feasibility or may cause *nondegenerate* cycling in oriented matroid programming. Therefore it is theoretically interesting to look for a pivot rule which is finite and keeps feasibility for oriented matroid programming. To the best knowledge of the authors, Todd's rule is the only pivot rule so far proved to be in that category. We notice here that there is a classical technique for preventing cycling in the simplex method, known as the lexicographic method. The lexicographic method has an advantage that only leaving basis selection is affected by the method and yet the finiteness is guaranteed. The lexicographic method can also be interpreted as a parametric perturbation method.

The equivalence of the self–dual parametric algorithm and Lemke's method applied to the linear complementarity problem defined by the primal and dual LP problem was proved by Lustig [45]. Todd's pivot rule is related to the lexicographic Lemke method (or the parametric perturbation method), hence using the equivalence mentioned above a simplex algorithm for LP can be derived. However, it is much more complicated to present this method in the linear programming context. The theoretical meaning exists in oriented matroid programming.

As Todd's rule is in fact a lexicographic Lemke method specialized to LP, and the perturbation is done first in the right hand side and then in the objective (wiht increasing order of the perturbation parameter), it gives finally a two phase simplex method. We do not present here the method completely, just the second phase – without proofs – to illustrate its behaviour. For a complete description we refer to [70] and [80].

Todd's Lexicographic Lemke Rule (II. Phase)

Suppose that we are given a feasible tableau $T(B)$.

For a given set of variable indices, say $\{1, 2, ..., m\}$, the tableau is called *lexico-feasible* if

$$(x_i, t_m^{(i)}, t_{m-1}^{(i)}, ..., t_1^{(i)})$$

is lexico-positive (the first nonzero element is positive) for every $i \in B$. It is obvious that if $B = \{1, 2, ..., m\}$ then the initial tableau is lexico-feasible.

In the following, at each iteration we always select the entering variable x_r such that $z_r < 0$ and

$$(\frac{t_{(r)m+1}}{z_r}, ..., \frac{t_{(r)n}}{z_r}) = lexico \min_{z_j<0}(\frac{t_{(j)m+1}}{z_j}, ..., \frac{t_{(j)n}}{z_j}),$$

where z is the vector of the reduced costs.

For selecting the leaving variable, let the index s of the leaving variable be such that

$$(\frac{x_s}{t_{rs}}, \frac{t_m^{(s)}}{t_{rs}}, ..., \frac{t_1^{(s)}}{t_{rs}}) = lexico \min_{t_{ri}>0}(\frac{x_i}{t_{ri}}, \frac{t_m^{(i)}}{t_{ri}}, ..., \frac{t_1^{(i)}}{t_{ri}}).$$

It can be seen that if $l, l+1, ..., n$ $(l \geq m+1)$ are not in the basis, then x_k with $k \geq l$ is selected to enter the basis only if $z_j \geq 0$ for $j \in \{1, 2, ..., m, m+1, ..., l-1\}$. Similar properties hold for leaving variables. This monotonicity guarantees no cycling.

The practical behavior of this method is similar to Bland's minimal index rule.To implement it is more complicated, but not much more expensive. A big drawback, as in general with lexicographic rules, is deciding when two entries are equal to within round off errors.

It is still interesting to give an easy and clear proof for the correctness of Todd's rule. Also, it is unknown if there are some other alternative simplex pivot rules which guarantee finiteness in the context of oriented matroid programming.

This concludes this section. In the following section, we will discuss some pivot rules especially for linear programming. They may not guarantee anti-cycling if there is no special treatment. Attentions are mainly paid to the numerical structures of the problem.

3 Pivot Rules Related to Parametric Programming

The algorithms discussed in this section are closely related to parametric programming, more precisely to the shadow vertex algorithm [9], [27], [50] and to Dantzig's self–dual parametric simplex algorithm [16].

As it was mentioned in the previous section that the self–dual parametric algorithm is equivalent to Lemke's method applied to the linear complementarity problem defined by the primal and dual LP problem. For details see Lustig [45]. It is also easy to see that the shadow vertex algorithm can be interpreted as a special case of the self–dual parametric simplex algorithm, therefore only the shadow vertex algorithm will be presented here. Using the above mentioned correspondence the reader can easily derive the connection between the new algorithms and the self–dual parametric algorithm (and the relation to Lemke's method).

Before presenting the algorithms, note that a variable will be called *driving variable* if it plays a special role in the pivot selection, namely the leaving (or entering) variable is subsequentially determined by using this column (or row).

The shadow vertex algorithm is mostly known by its geometrical interpretation. We present here an algebraic form, which was also discussed in Borgwardt [9].

Shadow Vertex Algorithm

Initialization: Let a dual feasible basis $\tilde{B}$ be given. Let $0 < \tilde{v} \in R^m$ be an arbitrary positive vector, and denote $v = A_{\tilde{B}}\tilde{v}$. Let us append this new vector to the current dual feasible basic tableau. (In the sequel vector v will play the role of the solution column, and vector $(-b)$ will play the role of the driving variable.)

Shadow Vertex Algorithm - The Pivot Rule

Step 1 Vector $(-b)$ is the driving variable.

Step 2 *Leaving variable selection*
Let $r = \text{argmin}\{-\frac{v_i}{x_i} \ : \ x_i < 0\}$ (x_r is the leaving variable).
If there is no r, then the problem is solved, stop.

Step 3 *Entering variable selection*
Let $s = \text{argmin}\{-\frac{z_j}{t_{rj}} \ : \ t_{rj} < 0, \ j \in N\}$.
If there is no s, then the problem is infeasible, stop.
Make a pivot on (r, s), x_r enters the basis. Go to **Step 2**.

There are two interesting properties of the shadow vertex algorithm. On one hand there are two ratio tests in every pivot steps — both the leaving and entering variables are selected by performing a ratio test. Dual feasibility is preserved during the algorithm – due to the second ratio test – while primal feasibility is lost for both b and v, but recovered in the last step. Primal feasibility (optimality) holds at each step for a linear combination of the two right hand sides. These properties are explored and extended in the Exterior Point Simplex Algorithm (EPSA) and are fully generalized in the Monotonic Build Up Simplex Algorithm (MBU). However, the original motivations of EPSA and MBU were independent from the shadow vertex algorithm. This correlation was found later.

Exterior Point Simplex Algorithm (EPSA)

EPSA was proposed by Paparrizos [57], [58]. By specializing this method Paparrizos obtained nice results for the assignment problem [59]. EPSA can be presented as follows.

Initialization: Let a dual feasible basis $\tilde{B}$ be given. Let $0 < \tilde{v} \in R^m$ be an arbitrary positive vector, and denote $v = A_{\tilde{B}}\tilde{v}$. Let us append this new vector $\tilde{v}$ to the current dual feasible basic tableau as the $(n+1)$-th column (for the original problem we have $a_{n+1} = v$). Let the corresponding reduced cost value be zero (for the original problem we have $z_{n+1} = c_{\tilde{B}}^T A_{\tilde{B}}^{-1} v = c_{\tilde{B}}^T \tilde{v}$).

EPSA - The Pivot Rule

Step 1 Make a pivot on $(r, n+1)$, where $r = \text{argmin}\{\frac{x_i}{t_{i,n+1}}\}$. (The new basis is dual feasible, and x_{n+1} is the only primal infeasible variable.)
The driving variable is $v = a_{n+1}$.

Step 2 *Entering variable selection*
Let $s = \text{argmin}\{-\frac{z_j}{t_{n+1,j}} \,:\, t_{n+1,j} < 0,\ j \in N\}$ (x_s is the entering variable).
If there is no s, then the problem is infeasible, stop.

Step 3 *Leaving variable selection*
Let $r = \text{argmin}\{\frac{x_i}{t_{is}} \,:\, t_{is} > 0,\ i \in B\}$. If there is no r, then let $r = n+1$.
Let $\theta_1 = \frac{x_r}{t_{rs}}$ and $\theta_2 = \frac{x_{n+1}}{t_{n+1,s}}$.

Step 3a If $\theta_1 < \theta_2$, then make a pivot on (r, s), x_r leaves the basis.
Go to **Step 2**.

Step 3b If $\theta_1 \geq \theta_2$, then make a pivot on $(n+1, s)$, v leaves the basis.
The new solution is optimal for the original problem, stop.

It is obvious that, after Step 1 we have an almost optimal basis except one primal infeasible variable.

EPSA and the shadow vertex algorithm have the common idea to introduce and auxiliary vector to force primal feasibility. The main difference is that an additional pivot is performed in EPSA. (A pivot in the $v-$row of the tableau.) Based on this observation it is easy to see that EPSA and the shadow vertex algorithm are closely related.

Anstreicher and Terlaky [1] presented the monotonic build-up simplex algorithm. It will be shown MBU can be interpreted as a common generalization of EPSA and the shadow vertex algorithm.

Monotonic Build Up Simplex Algorithm MBU

As in the primal simplex method, let us assume, that a primal feasible basic solution is available. The pivot rule is as follows.

MBU - The Pivot Rule

Step 0 Let $J^+ = \{j : z_j \geq 0\}$, $J^- = J - J^+ = \{j : z_j < 0\}$.

Step 1 Let $k \in J^-$ be an arbitrary index. (Variable x_k will referred as the driving variable.)
If $J^- = \emptyset$, then an optimal solution is found, stop.

Step 2 *Leaving variable selection*
Let $r = \text{argmin}\{\frac{x_i}{t_{ik}} \; : \; t_{ik} > 0\}$. ($x_r$ is the leaving variable.)
If there is no r, then the problem is unbounded, stop.

Step 3 *Entering variable selection*
Let $s = \text{argmin}\{-\frac{z_j}{t_{rj}} \; : \; t_{rj} < 0, \; j \in J^+\}$. If there is no s, then let $s = k$.
Let $\theta_1 = -\frac{z_s}{t_{rs}}$ and $\theta_2 = -\frac{z_k}{t_{rk}}$.

Step 3a If $\theta_1 < \theta_2$, then make a pivot on (r, s), x_r enters the basis. Let $J^+ = \{j : z_j \geq 0\}$. The driving variable is not changed. Go to **Step 2**.

Step 3b If $\theta_1 \geq \theta_2$, then make a pivot on (r, k), x_k enters the basis. Let $J^+ = \{j : z_j \geq 0\}$. Go to **Step 1**.

Note that at Step 3b, the set of dual feasible indices J^+ is strictly increasing and it does not decrease (but may increase) at Step 3a. That is why this algorithm is called *monotonic build-up* simplex method.

The pivot selection rule can be illustrated by the following Figure 2. In this scheme pivots are made in position $(-)$ at step 3a and on position $[+]$ at step 3b of MBU.

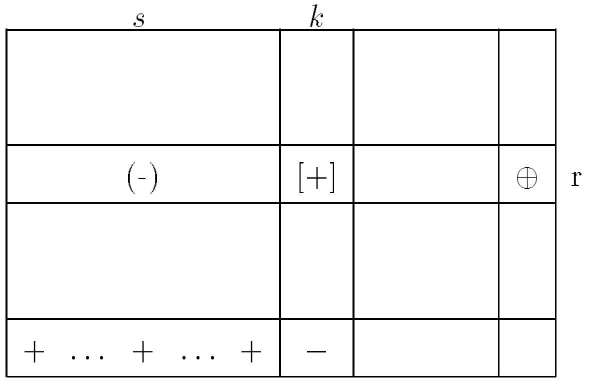

Figure 2.

Note that during certain pivot sequence while the driving variable is fixed, the basis may become infeasible. Interestingly as it was shown in [1], the ratio test at Step 2 is automatically restricted to the feasible part of the basis. At Step 3b, when the driving variable enters the basis, the primal feasibility of the whole basis is restored. Therefore one always has a primal feasible basis at Step 1. The validity of these statements and the correctness of the algorithm are proved in [1].

We further note that the algorithm has a strict monotone build-up feature. Obviously, J^+ is monotonically extending at Step 3b and at the same time the primal feasibility is restored. Interestingly, J^+ can be extended even if the driving variable is kept unchanged, and the dual feasibility of these variables is preserved in the subsequential pivots at Step 3a.

While the driving variable does not change, one may use dual lexicographic method in the inner cycle to make MBU finite. Of course the minimal index rule can also be applied, but to prove the finiteness in this case needs some extra effort.

This build-up structure resembles slightly to recursive rules, especially to the Edmonds–Fukuda [18] rule. But it can be easily seen that the two pivot rules are different. The relation between MBU and EPSA (and also the shadow vertex algorithm, the self–dual parametric simplex method and to Lemke's method) is more direct. In fact MBU can be considered as a generalization of EPSA. This claim becomes clearer if we look at the dual version of MBU.

Having a dual feasible basis, the pivot rule of the dual MBU is as follows.

Dual MBU

Step 0 Let $J^+ = \{i : x_i \geq 0\}$, $J^- = J - J^+ = \{i : x_i < 0\}$.

Step 1 Let $k \in J^-$ be an arbitrary index. (Variable x_k will be referred as the driving variable.)
If $J^- = \emptyset$, then an optimal solution is found, stop.

Step 2 *Entering variable selection*
Let $s = \text{argmin}\{-\frac{z_j}{t_{kj}} \ : \ t_{kj} < 0\}$. ($x_s$ is the entering variable.)
If there is no s, then the problem is unbounded, stop.

Step 3 *Leaving variable selection*
Let $r = \text{argmin}\{\frac{x_i}{t_{is}} \ : \ t_{is} > 0, \ i \in J^+\}$. If there is no r, then let $r = k$.
Let $\theta_1 = \frac{x_r}{t_{rs}}$ and $\theta_2 = \frac{x_k}{t_{ks}}$.

Step 3a If $\theta_1 < \theta_2$, then make a pivot on (r, s), x_r leaves the basis. Let $J^+ = \{i : x_i \geq 0\}$. The driving variable is not changed. Go to **Step 2**.

Step 3b If $\theta_1 \geq \theta_2$, then make a pivot on (k, s), x_k leaves the basis. Let $J^+ = \{i : x_i \geq 0\}$. Go to **Step 1**.

We have seen that MBU goes through basic solutions, that are neither primal nor dual feasible, but some of the solutions are primal feasible. The algorithm solves the original problem by subsequently solving its subproblems. Having a primal feasible solution at Step 1 we identify the actual subproblem that is solved (collect the dual feasible variables). Then by choosing the driving variable, the size of the subproblem to be solved is increased. This subproblem is solved by relaxing primal feasibility, but during the procedure, the size of the subproblem might be increased at Step 3a if new variables became dual feasible. So the size of the subproblem dynamically grows. MBU is based on the repeated application of this idea.

Now it is clear why it is called *monotonic build up* and why it is a generalization of EPSA. But it is still not clear why it is a *simplex* algorithm since the feasibility may be lost here, whereas it is preserved in simplex algorithms.

Actually it can be shown that MBU is equivalent to a specific primal simplex variant. Namely, a *primal feasible basic solution* can be associated to every basic solution that occurs in the procedure. More precisely, if we would make a pivot on position (r, k) at Step 3, then the resulting basic tableau (basis) would be primal feasible.

Another interesting feature of MBU (as a generalization of the *shadow vertex idea* – optimization on a plane) is that it can be interpreted geometrically as an *active set method.* Let us consider the dual variant of MBU. A subproblem (while a driving variable is fixed) can be interpreted as to maximize the dual objective on the intersection of the hyperplane defined by the driving variable and the optimal cone. After that, a new driving variable (and so a new hyperplane) is selected, and the procedure repeats.

EPSA, MBU and Finite Combinatorial Rules

We discussed some monotonic build-up schemes for LP in the second section. In fact any recursive type algorithm ([32], [66], [18]) can be interpreted as an algorithm that solves subproblems with monotonically increasing size. Edmonds–Fukuda rule [18] is the best known among the real simplex variants (with some restrictions). In contrast to MBU, the modified Edmonds–Fukuda rule preserves primal feasibility (see [14]) in solving the subproblems.

The criss–cross methods [81], [66], [67], [74] are exterior 'simplex-like' methods, but they do not need any feasible basis to start with. Furthermore, in criss–cross methods if primal or dual feasibility occurs, it is only by accident. In Zionts' [81] criss–cross method, the feasibility is preserved once attained, but in Terlaky's criss–cross method it is neither preserved nor recovered. Primal and dual feasibilities are guaranteed only at the last pivot step in which the optimal solution is found. Although these methods visits vertices which are neither primal nor dual feasible, they are essentially different from MBU. One of the main differences is that there is no simplex interpretation for the criss–cross methods.

Via the shadow vertex algorithm and the EPSA, it is clear that MBU has a strong background in parametric programming. In fact it can be interpreted as a repeated reparametrization of the right hand side with the actual driving variable (vector).

The last problem that we want to address in this section is the finiteness property and exponential examples. For exponential example for the shadow vertex algorithm see Goldfarb [27]. Paparrizos [58] presents an exponential example for EPSA. Since both are special cases of MBU, hence MBU is also exponential in the worst case.

Transition Node Pivoting (TNP)

To conclude this section we mention the so called Transition Node Pivot (TNP) rule suggested by Geue [26] and studied further by Gal and Geue [24]. The TNP rule applies only when degeneracy occurs. The rule is activated at the step before degeneracy occurs.

Assume that we are facing the situation such that the current basis is nondegenerate with an entering variable k, and after this pivot the next basis becomes primal degenerate. The set of indices for which the minimum ratio is attained will be denoted by $J_{\min}$.

TNP Pivot Rule

Step 0 The entering variable k is given, where the next basis is degenerate. Let t be another nonbasic variable index (t is referred as a *transition column*).
Let $r = \text{argmax}\{\frac{t_{it}}{t_{ik}} \; : \; i \in J_{\min}\}$ (x_r is the leaving variable).

Step 1 Make a pivot on (r, k).

If the basis is nondegenerate continue with any simplex pivot rule.

Step 2 *Entering variable selection*
Choose any dual infeasible variable k to enter the basis.

If there is no entering variable, then the problem is solved, stop.

Step 3 *Leaving variable selection*
Make the ratio test.

If $|J_{\min}| = 1$ then denote the element in $J_{\min}$ by r. Go to **Step 1**.

Step 3a Let $r = \text{argmin}\{\frac{t_{it}}{t_{ik}} \; : \; i \in J_{\min}\}$ (x_r is the leaving variable). Go to **Step 1**.

Unfortunately the TNP rule is not finite in general. To guarantee the finiteness it is necessary to introduce an additional "perturbation" column. Then the reduced cost of this extra column strictly increases, and hence cycling becomes impossible.

4 Pivot Rules Related to Interior Point Methods

In this section we will discuss some recent pivot rules for the simplex method which are in one way or another related to interior point methods. Since Karmarkar [34] first proposed an interior method which solves linear programming problems in polynomial time and is reported to perform superior to the simplex method for large scale problems, there have been a large number of research papers devoted to that topic. Interior point methods allow the iterative points to go inside the polytope, it is believed to be able to use more global information and therefore to avoid "myopiness" of the simplex method. Although it is still unclear by using interior point type methodology we may succeed in finding a polynomial simplex algorithm, we think it is definitely a new field deserves further investigation. In this section mainly three pivot rules will be discussed. They are: the minimal volume rule of Roos [60], the dual interior primal simplex method of Tamura, Takehara, Fukuda, Fujishige and Kojima [65], and Todd's rule [71] obtained as a limit of a projective method.

Minimal Volume Simplex Pivot Rule

We start our discussion of this section with the *minimal volume simplex* pivot rule suggested in an unpublished paper of Roos [60]. In Roos' paper, the primal problem (P) is considered. As in Karmarkar's approach, he assumes that the optimal value is known to be zero (this assumption was usual in projective methods). Moreover, all basic solutions are assumed to be nondegenerate. For a given basis B, let A_B be the submatrix of A and c_B be the subvector of c corresponding to the index set (basis) B. Now, consider a cone $\mathcal{C}_B$ in R^m associated with the basis B as follows:

$$\mathcal{C}_B := \{y \in R^m : A_B^T y \leq c_B\}.$$

Let the rows of A_B^{-1} be $y_1, y_2, ..., y_m$, then it can be shown that

$$\mathcal{C}_B = \{y_B - \sum_{i=1}^{m} \lambda_i y_i : \lambda_i \geq 0, i = 1, 2, ..., m\},$$

where $y_B^T := c_B^T A_B^{-1}$.

Moreover, y_i, $i = 1, 2, ..., m$, are rays of the cone $\mathcal{C}_B$. By the definition of the tableau, it is obvious that $t_{ij} = y_i^T a_j$, $i = 1, 2, ..., m$, $j = 1, 2, ..., n$.

The problem $\max\{y^T b : y \in \mathcal{C}_B\}$ is bounded from above if and only if $y_i^T b \geq 0$ for $i = 1, 2, ..., m$, and this corresponds to the feasibility of the basis B. Suppose the feasible basis B is not optimal. Remember that the optimal value was assumed to be zero, this means that $y_B^T b > 0$. Therefore the intersection of the cone $\mathcal{C}_B$ and the half space $\{y : y^T b \geq 0\}$ forms a nontrivial simplex. We denote this simplex by S_B. It is shown in Roos [60] that the volume of S_B can be computed as

$$vol(S_B) = \frac{1}{n!} \frac{(c^T x)^n}{\prod x} \frac{1}{\mid \det A_B \mid},$$

where Πx denotes the product of positive entries of x. The minimal volume pivot rule of Roos seeks for the next feasible basis with the minimal volume of such simplex. Note here that the simplex corresponding to the optimal basis has zero volume.

To calculate the volume of this simplex does not need much effort. If the pivot element is denoted by t_{rs}, then it follows that $\mid \det A_{B'} \mid = t_{rs} \mid \det A_B \mid$ with B' the next feasible basis. Hence Roos' pivot rule looks for entering variable index r with $z_r < 0$ and the feasible pivot element t_{rs} such that

$$\frac{1}{t_{rs}} \frac{(c^T x')^n}{\prod x'}$$

is minimal. Where x' denotes the basic feasible solution corresponding to the next feasible basis B'.

Observe that this formula strongly resembles Karmarkar's potential function. However, it is still unclear at this stage how to cope with the degenerate case. Also, more properties of this method need to be investigated.

Concerning Roos' unpublished paper [60] we note that he also presented another pivot rule, the so called *maximum distance* pivot rule. Here the variable is selected to enter the basis for which the corresponding hyperplane (in the dual formulation) is the farthest from the present vertex.

Dual Interior Primal Simplex Method (DIPS)

In the next we will introduce the so called Dual Interior Primal Simplex (DIPS) method developed by Tamura, Takehara, Fukuda, Fujishige and Kojima [65]. This method is also referred sometimes as *Stationary Ball Method.* DIPS is a modification of the gravitational method of Chang and Murty [53] and Murty [51]. The gravitational method can be explained as follows.

We consider the dual problem (D) : $\max\{b^T y : A^T y \leq c\}$. Suppose the interior of $\{y : A^T y \leq c\}$ is nonempty, then we can place a ball with certain radius r inside that region. Now, suppose the objective direction b is vertical, the ball will then fall down by the gravitational force until it touches some boundary of that region and rolls down while minimizes its potential energy. If we then reduce the radius of the ball subsequentially to zero, it will fall and roll down until it reaches the "bottom" of the feasible region (optimal point).

If an initial ball in a stationary position is known, and the radius of the ball is reduced discretely to zero, it can be shown under nondegeneracy assumption, that the trajectory of the centers of the ball is a piece wise linear locus. The gravitational method of Chang and Murty is to follow this central trajectory until certain precision requirements are satisfied. Their method is finite.

The DIPS method is based on a similar idea. An essential observation (Chang and Murty [12], Murty [53], and Tamura *et al.* [65]) is that a subset of constraints touched by a ball in its stationary position (after falling down) forms a feasible basis for the primal problem (P). Moreover, two adjacent break points in the the central trajectory results in two different but adjacent bases of the primal problem. For detailed proofs, one refers to Tamura *et al.* [65]. This trajectory is related to the "inverse barrier path" and is discussed by Den Hertog, Roos and Terlaky [31].

To derive a pivot rule some additional assumptions were needed. It the original paper it was assumed, that $||a_j|| = 1$ for all j and the LP problem is dual nondegenerate. Note, that the first assumption is not restrictive, it can be easily obtained by scaling, and also the resulted pivot rule can be presented without this assumption as follows.

Let y^i, $i = 1, 2, ...$, denote the break point of the piecewise linear central trajectory, B_i denote the feasible basis corresponding to the dual constraints reached by the ball with center y^i and with the maximum radius r_i of that position. Furthermore, let x^i denote the basic feasible solution corresponding to the basis B_i, $i = 1, 2, ...$, it is shown by Tamura *et al.* that

$$c^T x^{i+1} \leq c^T x^i,\ i = 1, 2, ...$$

$$b^T y^{i+1} \leq b^T y^i,\ i = 1, 2, ...$$

and

$$\lim_{i\to\infty} c^T x^i = \lim_{i\to\infty} b^T y^i.$$

This explains the name of the method (Dual Interior Primal Simplex).

More specifically,[1] the rule for selecting entering basis variable in the primal simplex version can be stated explicitly as follows.

For the tableau $T(B_i)$, if there is a nonbasic variable k such that $a_k^T y^i - c_k = -\|a_k\| r_i$ (i.e., the hyperplane corresponding to k touches the ball) then we set the entering variable r to the minimal index among such nonbasic variables; otherwise select

$$r = \text{argmin}_{c_k - a_k^T z < 0} \{ \frac{(a_k^T y^i - c_k + \|a_k\| r_i) + (c_k - a_k^T z)}{a_k^T y^i - c_k + \|a_k\| r_i} \}$$

where $z^T := c_{B_i}^T A_{B_i}^{-1}$, a_k denotes the kth column of A, c_k the kth coordinate of c (therefore $c_k - a_k^T z$ is the reduced cost of x_k^i). The leaving variable is selected by the usual simplex ratio test.

The DIPS method is reported to have similar numerical behavior as the maximum decrease pivot rule, whereas the complexity to perform each pivot is much less. Also, relations between the DIPS method and the parametric programming technique were discussed in [65]. This conclude the DIPS method.

Todd's Dantzig–Wolfe Like Pivot Rule

The last pivot rule of this section is the one proposed by Todd [71] which is closely related to the pivot rules presented by Klotz [42]. These rules are based on the reduced cost scaling as it was earlier used by Harris [29] in the DEVEX code and also studied by Kalan [33]. Reduced cost scaling with various scaling methods and their effect on the numerical performance of the simplex algorithm is extensively studied by Klotz [42]. Todd's pivot rule is based on a variant of Karmarkar's interior point method (cf. Todd [71]). The method is described as follows.

[1]This formulation is informed to the authors by Professor Tamura in private communications

We consider the linear programming in the following form

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax = 0 \\ & g^T x = 1 \\ & x \geq 0. \end{aligned}$$

Let x^k be an interior point in the feasible region of the above problem. Let $X = X_k := diag(x^k)$ denote the diagonal matrix with diagonal entries the coordinates of x^k. Define the oblique projection matrix

$$P = P_k := I - X^2 A^T (AX^2 A^T)^{-1} A.$$

It is easy to see that $AP = 0$ and $P^2 = P$.

Let the oblique projection of c and g be given as

$$\tilde{c} := P^T c \text{ and } \tilde{g} := P^T g.$$

The relaxed problem is defined by

$$\begin{aligned} \text{(RP)} \quad \min \quad & \tilde{c}^T x \\ \text{s.t.} \quad & \tilde{g}^T x = 1 \\ & x \geq 0. \end{aligned}$$

Note that (RP) is a continuous knapsack problem and it can be solved easily. If (RP) has an optimal solution w, let $\tilde{w} := Pw$. If (RP) is unbounded with an unbounded ray direction t, let $\tilde{t} := Pt$. In the first case, let $x^{k+1} := (1 - \lambda_k)x^k + \lambda_k \tilde{w}$, $\lambda_k \in (0, 1]$. In the second case, let $x^{k+1} := x^k + \mu_k \tilde{t}$, $\mu_k > 0$ With proper choice of λ_k or μ_k, and using Karmarkar's potential function, Todd proved ([71], 1990) that this method has the same convergence property as Karmarkar's method.

Now, if x^k is a nondegenerate basic feasible solution with basis B, a similar procedure can be applied. This results in the following pivot rule.

Let P solve $A_B^T P = c_B$ and σ solve $A_B^T \sigma = X_B^{-1} e$ where X_B is the diagonal matrix with diagonal entries the values of the basic variables. For each $j \notin B$, compute the reduced cost $z_j := c_j - P^T a_j$ and $\tilde{g}_j := \sigma^T a_j$.

If there is an r with $z_r < 0$ and $\tilde{g}_r = 0$, select it as the entering basis variable index.
Otherwise, select $r = arg \max\{\frac{z_j}{\tilde{g}_j} : z_j < 0, \tilde{g}_j < 0\}$.
Else, select $r = \text{argmin}\{\frac{z_j}{\tilde{g}_j} : z_j < 0, \tilde{g}_j > 0\}$.
If none of these applies, the current basis is optimal.
The leaving variable is selected by the usual minimum ratio test.

Reduced cost scaling is used in Todd's pivot rule. More analyses can be found in Todd's paper [71]. We note here that in case of degeneracy additional rules need to be adapted to prevent cycling.

This concludes this section. There are many interesting results related to pivot rules, as well as many open questions. In the next section, we will conclude the paper with some remarks.

5 Concluding Remarks

Although interior point methods have been most intensively studied in the last years, several pivot rules were developed for linear programming. We have surveyed three classes of pivot rules – combinatorial, parametric type and those derived from interior point methods.

Concerning combinatorial pivot rules remark, that there are a great number of pivot rules developed for special classes of network optimization. Most of them were never studied or generalized for general linear programming. To discuss these rules is out of the scope of the present paper. We also note that this is interesting to find a simple finiteness proof for Todd's simplex pivot rule for oriented matroid programming, or to find some other pivot rules belonging to this category.

Interestingly, only simplex methods based on parametric programming – the shadow vertex algorithm and Lemke's method – are proved to be polynomial in the expected number of pivot steps [9], [72]. It is still unknown if the other rules have this property as well. To the best knowledge of the authors only one attempt was made to examine the average number of steps of combinatorial pivot rules [20].

In the extensive literature of interior point methods one can find some papers discussing the relation between interior point methods and the simplex methods. Among them, there are papers by Chiu and Ye [13] and by Stone and Tovey [64]. In the last paper the simplex method and interior point methods were presented as iteratively reweighted least squares methods.

Sehti and Thompson [63] proposed the Pivot and Probe method where the simplex method was combined with build–up structure. This algorithm appeared before the theory of interior point methods became the central area for research in linear programming. The pivot and probe method is not polynomial, and was not seriously tested numerically.

Another recent trend to improve the efficiency of LP algorithms is to combine interior point algorithms with the simplex method. In this approach some interior point method is first applied to generate a sufficiently good interior solution, then the algorithm switches to the simplex method to find an optimal basis. This type of methods are discussed in [43], [46], [47], [5] and [77].

Towards degeneracy in linear programming and some other related problems, there is a relatively new approach called *theory of degeneracy graph* studied by Gal *et al.* [23].

In that theory, a degenerate vertex is associated with a graph. Each node of the graph represents a basis corresponding to the vertex. An edge between two nodes on the graph implies that a pivot step can be performed to move from one basis to the other. To study this graph will certainly give insight about the structure of the degenerate vertex.

To conclude the paper we note that the hardest and long standing open problems in the theory of linear programming are still concerned with pivot methods. Those include the d–step conjecture [41] and the question if there exist a polynomial pivot rule or not. For the last problem Zadeh's rule [78] might be a candidate. At least it is still not proved to be exponential in the worst case.

References

[1] K.M. Anstreicher and T. Terlaky, (1990) A Monotonic Build Up Simplex Algorithm, Report 91–82, Faculty of Technical Mathematics and Informatics, Delft University of Technology, The Netherlands.

[2] I. Adler and N. Megiddo, (1985) A Simplex Algorithm Whose Average Number of Steps is Bounded Between Two Quadratic Functions of the Smaller Dimension, *Journal of the Association of Computing Machinery,* 32, 891–895.

[3] D. Avis and V. Chvatal, (1978) Notes on Bland's Rule, *Mathematical Programming Study,* 8, 24–34.

[4] D. Avis and K. Fukuda, (1991) A Pivoting Algorithm for Convex Hulls and Vertex Enumeration of Arrangements and Polyhedra, *Research Report B-237,* Department of Information Sciences, Tokyo Institute of Technology, Tokyo, Japan.

[5] R.E. Bixby, J.W. Gregory, I.J. Lustig, R.E. Marsten and D.F. Shanno, (1991) Very Large–Scale Linear Programming: A Case Study in Combining Interior Point and Simplex Methods. *Report,* Department of Mathematical Sciences, Rice University, Houston, Texas.

[6] R.G. Bland, (1977) New Finite Pivoting Rules for the Simplex Method, *Mathematics of Operations Research,* 2, 103–107.

[7] R.G. Bland, (1977) A Combinatorial Abstraction of Linear Programming, *Journal of Combinatorial Theory Ser. B.* 23, 33–57.

[8] R.G. Bland and M. Las Vergnas, (1978) Orientability of Matroids, *Journal of Combinatorial Theory, Ser. B.* 24, 94–123.

[9] K.H. Borgwardt, (1987) *The Simplex Method: A Probabilistic Analysis.* Algorithms and Combinatorics, Vol 1. Springer Verlag.

[10] Y.Y. Chang, (1979) Least Index Resolution of Degeneracy in Linear Complementarity Problems, *Technical Report 79–14.* Department of Operations Research, Stanford University, Stanford, California.

[11] Y.Y. Chang and R.W. Cottle, (1980) Least Index Resolution of Degeneracy in Quadratic Programming, *Mathematical Programming,* 18, 127–137.

[12] S.Y. Chang and K.G. Murty, (1989) The Steepest Descent Gravitational Method for Linear Programming, *Discrete and Applied Mathematics,* 25, 211–239.

[13] S.S. Chiu and Y. Ye, (1985) Simplex Method and Karmarkar's Algorithm: A Unifying Structure, *Technical Report.* Engineering Economic Systems Department, Stanford University, Stanford, California.

[14] J. Clausen, (1987) A Note on Edmonds–Fukuda Pivoting Rule for the Simplex Method, *European Journal of Operations Research*, 29, 378–383.

[15] R.W. Cottle, (1989) The Principal Pivoting Method Revisited, *Technical Report SOL 89–3.* Stanford University, Stanford, California.

[16] G.B. Dantzig, (1963) *Liner Programming and Extensions.* Princeton University Press, Princeton N.J.

[17] J. Folkman and J. Lawrence, (1978) Oriented Matroids, *Journal of Combinatorial Theory, Ser. B.* 25, 199–236.

[18] K. Fukuda, (1982) Oriented Matroid Programming. *Ph.D. Thesis,* Waterloo University, Waterloo, Ontario, Canada.

[19] K. Fukuda and T. Matsui, (1991) On the Finiteness of the Criss–Cross Method, *European Journal of Operations Research* 52, 119–124.

[20] K. Fukuda and M. Namiki, (1991) Two Extremal Behavior of the Criss–Cross Method for Linear Complementarity Problems, *Research Report B-241,* Department of Information Sciences, Tokyo Institute of Technology, Tokyo, Japan..

[21] K. Fukuda and T. Terlaky, (1989) A General Algorithmic Framework for Quadratic Programming and a Generalization of Edmonds–Fukuda Rule as a Finite Version of Van de Panne–Whinston Method, *Mathematical Programming*, (to appear).

[22] K. Fukuda and T. Terlaky, (1990) Linear Complementarity and Oriented Matroids, *Journal of the Operational Research Society of Japan,* (to appear).

[23] T. Gal, (1989) Degeneracy Graphs – Theory and Application: A State-of-the-Art Survey, *Report No. 142*, Fern Universität Hagen, Germany.

[24] T. Gal and F. Geue, (1990) The use of the TNP–Rule to Solve Various Degeneracy Problems, *Report No. 149a*, Fern Universität Hagen, Germany.

[25] S. Gass and Th. Saaty, (1955) The Computational Algorithm for the Parametric Objective Function. *Naval Research Logistics Quarterly.* 2, 39–45.

[26] F. Geue, (1989) Eine Neue Pivotauswahlregel und die Durch sie Induzierten Teilgraphen des Positiven Entartungsgraphen. *Report No. 141.* Fern Universität Hagen, Germany.

[27] D. Goldfarb, (1983) Worst Case Complexity of the Shadow Vertex Simplex Algorithm. *Report,* Columbia University, Department of Industrial Engineering and Operations Research.

[28] M. Haimovitz, (1983) The Simplex Method is Very Good! – on the Expected Number of Pivot Steps and Related Properties of Random Linear Programs, *Report,* Columbia University, New York.

[29] P.M.J. Harris, (1973) Pivot Selection Methods for the Devex LP Code, *Mathematical Programming,* 5, 1–28.

[30] D. Den Hertog, C. Roos and T. Terlaky, (1990) The Linear Complementarity Problem, Sufficient Matrices and the Criss–Cross Method, *Report,* Faculty of Technical Mathematics and Informatics, Delft University of Technology, The Netherlands.

[31] D. Den Hertog, C. Roos and T. Terlaky, (1991) The Inverse Barrier Method for Linear Programming, *Report No. 91–27,* Faculty of Technical Mathematics and Informatics, Delft University of Technology, The Netherlands.

[32] D. Jensen, (1985) Coloring and Duality: Combinatorial Augmentation Methods. *Ph.D. Thesis,* School of OR and IE, Cornell University, Ithaca, NY.

[33] J.E. Kalan, (1976) Machine Inspired Enhancements of the Simplex Algorithm, *Technical Report CS75001-R,* Computer Science Department, Virginia Polytechnical University, Blacksburg, Virginia.

[34] N. Karmarkar, (1984) A New Polynomial–Time Algorithm for Linear Programming, *Combinatorica* 4, 373–395.

[35] L.G. Khachian, (1980) Polynomial Algorithms in Linear Programming, *Zhurnal Vichislitelnoj Matematiki i Matematischeskoi Fiziki* 20, 51–68 (in Russian), *USSR Computational Mathematics and Mathematical Physics* 20, 53–72 (English translation).

[36] E. Klafszky and T. Terlaky (1987) Remarks on the Feasibility Problems of Oriented Matroids, *Annales Universitatis Scientiarum Budapestiensis de Rolando Eötvös Nominatae, Sectio Computatorica,* Tom. VII, 155–157.

[37] E. Klafszky and T. Terlaky, (1989) Variants of the Hungarian Method for Solving Linear programming problems, *Math. Oper. und Stat. ser. Optimization*, 20, 1, 79–91.

[38] E. Klafszky and T. Terlaky, (1989) Some Generalizations of the Criss–Cross Method for Quadratic Programming, *Math. Oper. und Stat. ser. Optimization,* (to appear).

[39] E. Klafszky and T. Terlaky, (1989) Some Generalizations of the Criss–Cross Method for the Linear Complementarity Problem of Oriented Matroids, *Combinatorica,* 9, (2), 189–198.

[40] V. Klee and G.J. Minty, (1972) How Good is the Simplex Algorithm?, *in: O. Shisha, (ed.) Inequalities–III,* Academic Press, New York, 159–175.

[41] V. Klee and P. Kleinschmidt, (1987) The d–step Conjecture and its Relatives, *Mathematics of Operations Research*, Vol. 12, No. 4, 718–755.

[42] E. Klotz (1988) Dynamic Pricing Criteria in Linear Programming, *Technical Report SOL. No. 88–15,* Systems Optimization Laboratory, Stanford University, Stanford, California.

[43] K.O. Kortanek and J. Zhu, (1988) New Purification Algorithms for Linear Programming, *Naval Research Logistics Quarterly* 35, 571–583.

[44] C.E. Lemke, (1965) Bimatrix Equilibrium Points and Mathematical Programming, *Management Science,* 11, 681–689.

[45] I. Lustig, (1987) The Equivalence of Dantzig's Self–Dual Parametric Algorithm for Linear Programs to Lemke's Algorithm for Linear Complementarity Problems Applied to Linear Programming, *Technical Report,* SOL 87–4. Department of Operations Research, Stanford University, Stanford, California.

[46] N. Megiddo, (1988) On Finding Primal– and Dual–Optimal Bases. RJ 6328 (61997), IBM Almaden Research Center, San Jose, California.

[47] N. Megiddo, (1988) Switching from a Primal–Dual Newton Algorithm to a Primal–Dual (Interior) Simplex–Algorithm. RJ 6327 (61996), IBM Almaden Research Center, San Jose, California.

[48] K.G. Murty, (1974) A Note on a Bard Type Scheme for Solving the Complementarity Problem. *Opsearch* 11, (2–3), 123–130.

[49] K.G. Murty, (1976) *Linear and Combinatorial Programming,* Krieger Publishing Company, Malabar, Florida, USA.

[50] K.G. Murty, (1980) Computational Complexity of Parametric Linear Programming, *Mathematical Programming* 19, 213–219.

[51] K.G. Murty, (1986) The Gravitational Method of Linear Programming, *Technical Report No. 86–19.* Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109–2117, USA.

[52] K.G. Murty, (1988) *Linear Complementarity, Linear and Nonlinear Programming.* Sigma Series in Applied Mathematics, Vol. 3, Heldermann Verlag, Berlin.

[53] K.G. Murty, (1985) A New Interior Variant of the Gradient projection Method for Linear Programming, *Technical Report No. 85–18*, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109–2117, USA.

[54] M. Namiki and T. Matsui, (1990) Some Modifications of the Criss–Cross Method, *Research Report,* Department of Information Sciences, Tokyo Institute of Technology, Tokyo, Japan..

[55] P.–Q. Pan, (1988) Practical Finite Pivoting Rules for the Simplex Method, *OR Spektrum*, 12, 219–225.

[56] K. Paparrizos, (1989) Pivoting Rules Directing the Simplex Method Through All Feasible Vertices of Klee–Minty Examples, *Opsearch,* 26, 2, 77–95.

[57] K. Paparrizos, (1989) An Exterior Point Simplex Algorithm, *Lecture at the EURO–TIMS conference,* Belgrade, Jugoslavia.

[58] K. Paparrizos, (1991) A Simplex Algorithm with a New Monotonicity Property, *Preprint,* 1991.

[59] K. Paparrizos, (1991) An Infeasible (Exterior Point) Simplex Algorithm for Assignment Problems, *Mathematical Programming*, 51, 45–54.

[60] C. Roos, (1986) A Pivoting Rule for the Simplex Method which is Related to Karmarkar's Potential Function, *Manuscript*, Faculty of Technical Mathematics and Informatics, Delft University of Technology, The Netherlands.

[61] C. Roos, (1990) An Exponential Example for Terlaky's Pivoting Rule for the Criss–Cross Simplex Method, *Mathematical Programming*, 46, 78–94.

[62] A. Schrijver, (1987) *Theory of Linear and Integer Programming,* John Wiley & Sons.

[63] A. Sehti and G. Thompson, (1984) The Pivot and Probe Algorithm for Solving a Linear Problem, *Mathematical Programming,* 29, 219–233.

[64] R.E. Stone and C.A. Tovey, (1991) The Simplex and Projective Scaling Algorithms as Iteratively Reweighted Least Squares Methods, *SIAM REVIEW,* 33, No. 2, 220–237.

[65] A. Tamura, H. Takehara, K. Fukuda, S. Fujishige and M. Kojima, (1988) A Dual Interior Primal Simplex Method for Linear Programming, *Journal of the Operations Research Society of Japan,* Vol. 31, No. 3, 413–430.

[66] T. Terlaky, (1987) A Finite Criss–Cross Method for Oriented Matroids. *Journal of Combinatorial Theory (B)*, Vol. 42, No. 3, 319–327.

[67] T. Terlaky, (1985) A Convergent Criss–Cross Method. *Math. Oper. und Stat. ser. Optimization*, 16, 5, 683–690.

[68] T. Terlaky, (1987) A New Algorithm for Quadratic Programming, *European Journal of Operations Research,* 32, 294–301.

[69] M.J. Todd, (1984) Complementarity in Oriented Matroids, *SIAM Journal on Algebraic and Discrete Mathematics,* 5, 467–485.

[70] M.J. Todd, (1985) Linear and Quadratic Programming in Oriented Matroids, *Journal of Combinatorial Theory Ser. B.* 39, 105–133.

[71] M.J. Todd, (1990) A Dantzig–Wolfe Like Variant of Karmarkar's Interior–Point Linear Programming Algorithm, *Operations Research,* 38, 1006–1018.

[72] M.J. Todd, (1986) Polynomial Expected Behavior of a Pivoting Algorithm for Linear Complementarity and Linear Programming Problems, *Mathematical Programming,* 35, 173–192.

[73] A. Tucker, (1977) A Note on Convergence of the Ford–Fulkerson Flow Algorithm, *Mathematics of Operations Research*, Vol. 2, No. 2, 143–144.

[74] Zh. Wang, (1987) A Conformal Elimination Free Algorithm for Oriented Matroid Programming, *Chinese Annals of Mathematics*, 8 B 1.

[75] Zh. Wang, (1989) A Modified Version of the Edmonds–Fukuda Algorithm for LP problems in the General Form, *Asia–Pacific Journal of Operations Research*, (to appear).

[76] Zh. Wang, (1990) A General Deterministic Pivot Method for Oriented Matroid Programming, *Chinese Annals of Mathematics*, (to appear).

[77] Y. Ye and J.A. Kaliski, (1991) Further Results on Build–Up Approaches for Linear Programming, *Lecture at the ORSA/TIMS Meeting*, Anaheim, California.

[78] N. Zadeh, (1980) What is the Worst Case Behavior of the Simplex Algorithm? *Technical Report No. 27,* Department of Operations Research, Stanford University, Stanford, California.

[79] S. Zhang, (1991) On Anti–Cycling Pivoting Rules for the Simplex Method, *Operations Research Letters*, 10, 189–192.

[80] G.M. Ziegler, (1990) Linear Programming in Oriented Matroids, *Technical Report, No. 195,* Institute für Mathematik, Universität Augsburg, Germany.

[81] S. Zionts, (1969) The Criss–cross Method for Solving Linear Programming Problems, *Management Science,* 15, 7, 426–445.