Chin. Ann. of Math.
13B: 2(1992), 222—229.

A GENERAL DETERMINISTIC PIVOT METHOD FOR ORIENTED MATROID PROGRAMMING**

WANG ZHEMIN (王哲民)*

Abstract

A new pivot method for oriented matroid programming is given out. This method is deterministic by nature and is general in the sense that its flexible pivot selection rule allows a family of possible algorithms to be its special cases, including the so called criss-cross algorithm and the Edmonds-Fukuda algorithm as well. As an example of a special implementation of our general method, an extended version of the Edmonds-Fukuda algorithm is presented.

§ 1. Introduction

For oriented matroid programming, the so called criss-cross algorithm, which was discovered independently by Terlaky and the author[8,10,11], and the Edmonds-Fukuda algorithm[4] are both of deterministic type. The criss-cross algorithm requires no initial feasible circuit and uses no conformal eliminationoracle, whereas the Edmonds-Fukuda algorithm does. However, the latter gives an explicit way of showing how the iterations converge by a lexicographical ordering, while the former does not. Moreover, the pivot selection rule of the latter allows a certain flexibility, whereas that of the former is fixed. In this paper we relate these two algorithms and produce a general pivot method, which can be regarded somehow as a unified version of these two seemingly different algorithms. Also introduced in this paper, as an example of a special implementation of our general method, is an extended version of the Edmonds-Fukuda algorithm, which, unlike the original version, requires no initial feasible circuit. Methods based on recursive approach [1,2,5] do not in general give explicit ways of showing how the iterations converge, while ours does by means of providing an upper bounded binary digit, which increases monotonically not by only one unit in general) at each iteration.

Let E be a finite set, a signed set of E is defined as an ordered pair of subsets

Manuscript received May 25, 1990.

* Department of Statistics Operations Research, Fudan University Shanghai, 200433, China.

**Projects supported by the Natural Science Foundation of Fudan University.

(x', x'') of E, where $x' \cap x'' = \varnothing$, and $x' \cup x''$ is called the underlying set of (x', x''). Two signed sets (x', x'') and (y', y'') are called orthogonal if either the intersection of their underlying sets is empty or $(x' \cap y') \cup (x'' \cap y'')$ and $(x' \cap y'') \cup (x'' \cap y')$ are simultaneously nonempty. Let M and M^* be a pair of dual matroids with E as their ground set[12], M and M^* are called oriented matroids[3,4] if

(1) for every circuit C of M and M^*, it has been assigned to it two signed sets (C', C'') and (C'', C') with C as their underlying set;

(2) the signed sets assigned to the circuits of M are orthogonal to those assigned to the circuits of M^*.

Now, members of any such signed set are called the oriented parts of the circuit concerned.

One of the fund amental properties of oriented matroids is the so called Conformal Elimination Property[2,4,10], which can be stated as: Assume C_1 and C_2 are two distinct circuits of a given oriented matroid, (C_1^+, C_1^-) and (C_2^+, C_2^-) are two signed sets assigned to C_1 and C_2 respectively; if $A = (C_1^+ \cap C_2^-) \cup (C_1^- \cap C_2^+)$ is nonempty, then for any given nonempty subset A' of A and any given element f of $(C_1 \cup C_2)/A$, there exists an element f' of A' and a circuit C_3 of the given oriented matroid such that:

(1) $f \in C_3$ and $f' \overline{\in} C_3$;

and

(2) among the two signed sets assigned to C_3, there must be one, say, (C_3^+, C_3^-) satisfying:

$$(C_3^+ \cap C_1) \backslash A'' \subseteq C_1^+,$$
$$(C_3^- \cap C_1) \backslash A'' \subseteq C_1^-,$$
$$C_3^+ \cap (C_2 \backslash C_1) \subseteq C_2^+,$$
$$C_3^- \cap (C_2 \backslash C_1) \subseteq C_2^-,$$

where $A'' = A \backslash A'$.

When A' contains only one element, this property is briefly called the Elimination Property.

Let b and b^* be two distinct given elements of E; a circuit C of M is called feasible, if $b \in C$ and either $\{b^*\}$ or the empty set is an oriented part of C; a circuit D of M^* is called feasible, if $b^* \in D$ and either $\{b\}$ or the empty set is an oriented part of D. A feasible circuit C of M and a feasible circuit D of M^* are called complementary, if $(C \cap D) \backslash \{b, b^*\}$ is empty. A circuit $\tilde{C}$ of M is called infinitely-augmenting, if $b^* \in \tilde{C}$, $b \overline{\in} \tilde{C}$ and the empty set is an oriented part of $\tilde{C}$; a circuit $\tilde{D}$ M^* is called infinitely-augmenting, if $b \in \tilde{D}$, $b^* \overline{\in} \tilde{D}$ and the empty set is an oriented part of $\tilde{D}$.

The most fundamental theorem of oriented matroid programming is the Du-

ality Theorem[2,4,8,9,10], which can be stated as: One and only one of the following statements holds:

(1) Either there is an infinitely-augmenting circuit $\tilde{C}$ of M or there is an infinitely-augmenting circuit $\tilde{D}$ of M^* (this statement is equivalent, because of orthoganality, to that: either there is no feasible circuit of M^* or there is no feasible circuit of M).

(2) There exist a feasible circuit C of M and a feasible circuit D of M^* such that C and D are complementory.

Oriented motroid programming provides algorithms for realizing constructively the Duality Theorem. Based on recursive approach, Bland first proposed such an algorithm[2], which was later on further improved by Jensen[5]. The Edmonds-Fukuda algorithm and the so called criss-cross algorithm are also algorithms of this kind; whereas theirs are based on deterministic approach. Although, generally speaking, their algorithms can be interpreted as refinements of the Jensen algorithm, theirs however employ no recursive procedure and, thereby, appear more straightforward and implemental. The Edmonds-Fukuda algorithm requires an initial feasible circuit and an oracle capable of realizing the Conformal Elimination Property, while the criss-cross algorithm requires none of them; unfortunately, the pivot selection rule of the latter allows no flexibility and gives no explicit way of showing how the iterations converge, while the former does. We relate this two seemingly different algorithms and present in the following section a new algorithm, which is virtually an improved version of the criss-cross algorithm in the light of the Edmonds-Fukuda algorithm.

§ 2. Algorithm

For a base B of M, let B^* denote the base $E\backslash B$ of M^*; for $e\in B^*$, let $C(e, B)$ denote the unique circuit of M that is contained in $B\cup\{e\}$, and let $C^+(e, B)$ denote the oriented part of $C(e, B)$ that contains e, $C^-(e, B)$ the other oriented part of $C(e, B)$; for $e^*\in B$, let $D(e^*, B^*)$ denote the unique circuit of M^* that is contained in $B^*\cup\{e^*\}$, and let $D^+(e^*, B^*)$ denote the oriented part of $D(e^*, B^*)$ that contains e^*, $D^-(e^*, B^*)$ the other oriented part of $D(e^*, B^*)$.

Without loss of generality, we assume that $\{b^*\}$ is not a circuit of M (otherwise, $\{b^*\}$ itself is an infinitely-augmenting circuit of M) and $\{b\}$ is not a circuit of M^* (otherwise, $\{b\}$ itself is an infinitely-augmenting circuit of M). On this assumption we can assume that the initial base B of M is such one that $b^*\in B$ and $b\in B^*$. We then assume that $E'=E\backslash\{b, b^*\}$ is nonempty (otherwise, $C(b, B)$ and $D(b^*, B^*)$ become now a pair of complementary feasible circuits of M and M^*).

This algorithm uses a working vector $\lambda=(\lambda_1,\lambda_2,\cdots,\lambda_{|E'|})$ for the purpose of giving the elements of E' a changeable ordering. Also used in this algorithm is a working subset L of E', which, together with λ, provides for an explicit way of showing how the iterations of this algorithm converge. Our algorithm is as follows:

Step 0: Set

$\lambda:=$ whichever permutation of the elements of E' that one chooses to take

$L:=\varnothing$.

Step 1: Let

$K_1=\{\lambda_k\,|\,\lambda_k\in D^-(b^*, B^*)\}$,

$K_2=\{\lambda_k\,|\,\lambda_k\in C^-(b, B)\}$;

and let

$K_1'=K_1\backslash L$,

$K_2'=K_2\backslash L$.

If $K_1'\cup K_2'=\varnothing$, set

$C:=C(b, B)$,

$D:=D(b^*, B^*)$;

now, C and D are a pair of complementary feasible circuits of M and M^*, stop.

Otherwise, let

$$\tilde{k}=\min\{k\,|\,\lambda_k\in K_1'\cup K_2'\};$$

if $\lambda_{\tilde{k}}\in K_1'$, go to Step 2; if $\lambda_{\tilde{k}}\in K_2'$, go to Step 3.

Step 2: Let

$$I=\{\lambda_i\,|\,\lambda_i\in C^-(\lambda_{\tilde{k}}, B)\},$$
$$I'=I\backslash\{\lambda_i\,|\,i>\tilde{k} \text{ and } \lambda_i\in L\}.$$

If $I'=\varnothing$, set

$$\tilde{C}:=C(\lambda_{\tilde{k}}, B);$$

now, $\tilde{C}$ is an infinitely-augmenting circuit of M, stop.

Otherwise, go, according to what one likes, to Step 2.1 or Step 2.2.

Step 2.1: Choose, if possible, whichever λ_i one likes such that $i<\tilde{k}$ and $\lambda_i\in C(\lambda_{\tilde{k}}, B)$; let $\tilde{j}$ denote $\tilde{k}$, go to Step 4.

Step 2.2: Let $\lambda_{\tilde{i}}$ be such a coordinate of λ that $\tilde{i}=\min\{i\,|\,\lambda_i\in I'\}$; let $\tilde{j}$ denote $\tilde{k}$, go to Step 4.

Step 3: Let

$$J=\{\lambda_j\,|\,\lambda_j\in D^-(\lambda_{\tilde{k}}, B^*)\},$$
$$J'=J\backslash\{\lambda_j\,|\,j>\tilde{k} \text{ and } \lambda_j\in L\}.$$

If $J'=\varnothing$, set

$\tilde{D}:=D(\lambda_{\tilde{k}}, B^*)$;

now, $\tilde{D}$ is an infinitely-augmenting circuit of M^*, stop.

Otherwise, go, according to what one likes, to Step 3.1 or Step 3.2.

Step 3.1: Choose, if possible, whichever λ_j that one likes such that $\tilde{j}<\tilde{k}$ and $\lambda_j \in D(\lambda_{\tilde{k}}, B^*)$; let $\tilde{i}$ denote $\tilde{k}$, go to Step 4.

Step 3.2: Let $\lambda_{\tilde{j}}$ be such a coordinate of λ that $\tilde{j}=\min\{j \mid \lambda_j \in J'\}$; let $\tilde{i}$ denote $\tilde{k}$, go to Step 4.

Step 4:

Step 4.1: Let $t=\max\{\tilde{i}, \tilde{j}\}$, and set $L:=\{\lambda_t\} \cup \{\lambda_u \mid u>t$ and $\lambda_u \in L\}$ (now, if $t=\tilde{k}$, we say that λ_t enters into L "actively"; if $t\neq\tilde{k}$ i. e., $t>\tilde{k}$, we say that λ_t enters into L "passively"),

$B:=(B\setminus\{\lambda_i\}) \cup \{\lambda_j\}$.

Step 4.2: In this step, the positions of the elements of E'/L in λ can be rearranged in any way on condition that their relative positions with respect to the positions of the e lements of L in λ remain unchanged.

Go to Step 1.

The following lemma ensures that this algorithm is finite.

Lemma 1. *Cycling of bases never occurs to this algorithm.*

Proof From the rule for resetting L, it canbe seen that λ_t, if once having entered into L, can no longer go out until some other λ_u with $u>t$ has entered; hence, it is not difficuit to see that cycling of bases is impossible for this algorithm.

The finiteness of our algorithm also follows from the following lemma.

Lemma 2. *The binary digit* $\sum_{\lambda_i \in L} 2^i$ *is less than* $2^{|E'|+1}$ *and it increases monotonically at each iteration of* L.

Proof This lemma is a direct outcome of the rule for resetting L.

Lemma 2 not only assures us of the finiteness of this algorithm but also provides an explicit way of showing how the iterations of this algorithm converge.

To prove the validity of this algorithm, we need the following lemma, which is actually the essence of this paper.

Lemma 3. *The following three statements must hold:*

(1) *If, in Step* 1, $K_1 \cup K_2 \neq \emptyset$, *then* $K_1 \cup K_2' \neq \emptyset$ *and*

$$\min\{k \mid \lambda_k \in K_1 \cup K_2\}=\min\{k \mid \lambda_k \in K_1' \cup K_2'\}.$$

(2) *If, in Step* 2, $I \neq \emptyset$, *then* $I' \neq \emptyset$ *and*

$$\min\{i \mid \lambda_i \in I\}=\min\{i \mid \lambda_i \in I'\}.$$

(3) *If, in Step* 3, $J \neq \emptyset$, *then* $J' \neq \emptyset$ *and*

$$\min\{j \mid \lambda_j \in J\}=\min\{j \mid \lambda_j \in J'\}.$$

From lemma 3 we know that $K_1' \cup K_2'=\emptyset$, $I'=\emptyset$, and $J'=\emptyset$ are respectively equivalent to $K_1 \cup K_2=\emptyset$, $I=\emptyset$, and $J=\emptyset$; hence, it is easy to see that our

algorithm is valid.

The proof of Lemma 3 is given in the following.

Proof Suppose that, in the process of iterations, Lemma 3 first fails in (2), then a contradiction can be derived as follows: Let $\tilde{\tilde{i}}=\min\{i\,|\,\lambda_i\in I\}$, then it must be now that $\tilde{\tilde{i}}>\tilde{k}$ and $\lambda_{\tilde{i}}\in L$. Assume that, right before $\lambda_{\tilde{i}}$ last enters into L, the base of M^* is B_1^*; if $\lambda_{\tilde{i}}$ enters into L "actively", let D' denote $D(b^*, B_1^*)$; if $\lambda_{\tilde{i}}$ enters into L "passively", let D' denote $D(\lambda_{\tilde{k}_1}, B^*)$, where $\tilde{k}_1$is the "k" in Step 1 referring to the base B_1 of M. Now, it can be verified that the signed set assigned to $C(b, B)$ would not be orthogonal to that assigned to D'; this is, of course,contradictory.

Suppose that, in the process of iterations, Lemma 3 first fails in (3), then a contradiction can be derived in the same way as what we have done above.

Suppose that, in the process of iterations, Lemma 3 first fails in (1), then a contradiction can be derived as follows: Let $\tilde{\tilde{k}}=\min\{k\,|\,\lambda_k\in K_1\cup K_2\}$, then it must be now that $\lambda_{\tilde{k}}\in L$. We assume, without loss ofgenerality, that $\lambda_k\in K_2$. Right befo forc $\lambda_{\tilde{i}}$ last enters into L, let the then base of M^* be B_1^*. If $\lambda_{\tilde{i}}$ enters into L "passively", it can be verified that the signed set assigned to $C(b, B)$ would not be orthogonal to that assigned to $D(\lambda_{\tilde{k}_1}, B_1^*)$, where $\tilde{k}_1$ is the k in Step 1 referring to the base B_1 of M; this is of course contradictory. If $\lambda_{\tilde{i}}$ enters into L "actively", let C_1, C_2, C_1^+, C_2^-, f and f' of the Elimination Property be respectively $C(b, B)$, $C(b, B_1)$, $C^+(b, B)$, $C^+(b, B_1)$, $\lambda_{\tilde{i}}$ and b. Now, on the one hand because the signed set assigned to the resultant C_3 of the EliminationProperty should be orthogonal to that assigned to $D(b^*, B_1)$, it can be verified that b^* and $\lambda_{\tilde{i}}$ must locate in the same oriented part of C_3; on the other hand, because the signed set assigned to C_3 should also be orthogonal to that assigned to $D(b^*, B)$, it can be verified that b^* and $\lambda_{\tilde{i}}$ must not locate in the same oriented part of C_3; this is of course a contradiction, hence, this lemma is proved.

We should acknowledge here that the above proof is enlightened somehow by the argumens that Bland had used to establish his famous Rule I and II in [1].

§ 3. Examples

Two examples of special implementations of our general method are given in the following, one of them results in producing a revised version of the criss-cross algorithm, and the other an extended version of the Edmonds-Fukuda algorithm, which, unlike the original version, requires no initial feasible circuit.

In our first example, we keep λ unchanged throughout, and we always adopt Step 2.2 in Step 2 and Step 3.2 in Step 3; then, our algorithm becomes specifically the criss-cross algorithm now. Although in its original version there happens no

λ, L, K_1', K_2', I' and J', but K_1,K_2 I, and J only, this difference is however nonessential in the light of Lemma 3; and, what is more, this revised version gives an insight into seeing how the iterations converge.

In our second example, three additional specifications for our general algorithm are added as follows:

(1) In Step 0, set λ specifically in such a way that its coordinates, if being elements of B, always precede those being elements of B^*.

(2) In Step 4.2, λ is reset specifically in such a way that, among all such coordinates of λ that precede λ_t, those, if being elements of B, always precede the others.

(3) In Step 2, always let Step 2.1 be adopted (this is possible in this case because, owing to (1) and (2), for any $\lambda_i \in I'$, i must be less than $\tilde{k}$), and let $\lambda_{\tilde{i}}$ be chosen specifically in such a way; if there is an element of I' that does not belong to $C(b, B)$, then this element is chosen as $\lambda_{\tilde{i}}$; otherwise, let C_1, C_2, C_1^+, C_2^-, A',and f of the Conformal Elimination Property be respectively $C(b, B)$, $C(\lambda_{\tilde{k}}, B)$, $C^+(b, B)$, $C^-(\lambda_{\tilde{k}}, B)$, I' and b; and let the resultant f' of the Conformal Elimination Property be chosen as $\lambda_{\tilde{i}}$ (because of the first conclusion of Lemma 3, I' is now a subset of $C_1^+ \supset C_2^-$, hence, I' is qualified to be chosen as A').

Now, with referring to the specifications listed above, the following properties concerning this specified algorithm can be readily verified.

(i) If $e \in L$, then $e \in B$.

(ii) In Step 3, Step 3.1 can never be reached (because in this case for $\lambda_j \in J'$, j must be greater than $\tilde{k}$).

(iii) Step 3.2 can onlybe reached at the beginning stage of operation, hence, once Step 2 is reached, Step 3 can never be come across again.

(iv) After Step 2 has been come across, every ensuing $C(b, B)$ always L' satisfies $C(b, B)/L \subseteq C^+(b, B)$.

(It should be noted here that when the matroids under consideration are, specifically linear ones, (iv) can be further strengthenedi Let L' denote sush a subset of L that its elements are those that enter into L before Step 2 is come across, then after Step 2 has been reached, every ensuing $C(b, B)$ satisfies: $C(b, B) \backslash L' \subseteq C^+(b, B)$. Now, if the initial $C(b, B)$ is assumed to be feasible, then every ensuing $C(b, B)$ must also be feasible. This argument has been proved by Clausen in [3]. In fact the general conclusion made in the strengthened (iv) can be proved by means of the same approach that Clausen had used in [3], i.e., making full use of the fact that once Step 2 is reached the value of the "objective function" with respect to $C(b, B)$ must be monotonical.)

In case that th initial $C(b, B)$ is assumed to be feasible, the above specified

algorithm becomes, in particular, the Edmonds–Fukuda algorithm. As feasibility of the initial $C(b, B)$ is not made a precondition for this specified algorithm, hence, it can be regarded as an extended version of the Edmonds–Fukuda algorithm.

The oriented matroid rendering of the linear complementarity programming (OM–LCP) was first discussed by Todd[9]. He proposed an algorithm for OM–LCP that its prototype is an algorithm given by Lemke[7]. However, the Todd algorithm has to be carried out in an enlarged matroid structure of the original one. Later Klafszky and Terlaky made modification of the criss-cross algorithm for oriented matroid programming and developed accordingly an algorithm for OM–LCP[6]. Their algorithm is still criss-cross like and can be carried out within the scope of the given matroid structure without having to make it enlarged. However, it still suffers the same disadvantages as its prototype, i.e., its pivot selection rule is fixed and it gives no explicit way of showing how the iterations converge. With proper technicality modification, our general deterministic scheme for oriented matroid programming can also be applied to OM–LCP, and a general deterministic pivot method for OM–LCP can be established accordingly. We will have this discussed separately in another paper that is about to come in the near future.

References

[1] Bland, R. G., New finite pivoting rules for the simplex method, *Mathematics of Operations Research*, **2** (1977), 103—107.

[2] Bland, R. G., A combinatorial abstraction of linear programming, *Journal of Combinatorial Theory, Ser. B.*, **23** (1977), 33—57.

[3] Clausen, J., A note on the Edmonds-Fukuda pivoting rule for simplex algorithm, *European J. of Oper. Res.* 29 (1987), 378—383.

[4] Fukuda, K., Oriented Matroid programming, Ph. D. Thesis, University of Waterloo, Waterloo, CANADA, 1982.

[5] Jensen, D., Colouring and duality: combinatorial augmentation methods Ph. D. Thesis, Cornell University, 1985.

[6] Klafszky, E. & Terlaky, T., Some generalization of the criss-cross method for the linear complementarity problem of oriented matroids, *Combinatorica*, **9**: 2 (1989), 189—198.

[7] Lemke, C. E., Bimatix equilibrium points and mathematical programming, *Management Science*, **11** (1965), 681—689.

[8] Terlaky, T., A finite criss-cross method for oriented matroids, *Journal of Combinatorial Theory Ser. B.*, **42** (1987), 319—327.

[9] Todd, M. J., Linear and quadratic programming in oriented matroids, *Journal of Combinatorial Theory Ser. B.*, **39** (1985), 105—133.

[10] Wang Zhemin, A finite conformal-elimination free algorithm for oriented matroid programming, *Chinese Annals of Maths.* **8B**: 1 (1987), 120—125.

[11] Wang Zhemin, A minimum-ratio-test free approach to linear programming, *Chin. Ann. of Math.*, **10B**: 4 (1989), 472—479.

[12] Welsh, D. J. A., Matroid Theory Academic Press, London, 1976.