**Noname manuscript No.**
(will be inserted by the editor)

# Combinatorial Relaxation Algorithm for Entire Sequence of Maximum Degree of Minors

**Kazuo Murota · Shun Sato**

Received: date / Accepted: date

**Abstract** This paper presents an efficient algorithm for computing the entire sequence of maximum degree of minors, whereas the previous one find them of a specified order $k$. The efficiency derives from good properties of valuated bimatroids such as concavity.

The present algorithm adopts "combinatorial relaxation" which is a general framework with practical efficiency and accuracy. Combinatorial relaxation type algorithm first solves the relaxation problem, weighted bipartite matching problem in our case, to obtain an estimate on optimal value. Then, it checks the estimate is correct or not. If it is incorrect, the algorithm modifies the problem to improve the estimate without affecting the optimal value.

**Keywords** Combinatorial relaxation · Degree of minor · Valuated bimatroid

## 1 Introduction

Finding the maximum degree of minors is fundamental problem in engineering. Smith–McMillan form at infinity and Kronecker form is important application of maximum degree of minors. Smith–McMillan form at infinity is a normal form of rational matrices and important in the control theory(Commault–Dion[2]). Kronecker form is a normal form of matrix pencils and it can be used to analyze DAEs(Gear[7]). They needs the entire sequence of maximum degree of minors. Therefore, we propose the efficient combinatorial relaxation

K. Murota
Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo, Tokyo 113-8656, Japan
E-mail: murota@mist.i.u-tokyo.ac.jp

S. Sato
Department of Mathematical Informatics, Graduate School of Information Science and Technology, University of Tokyo, Tokyo 113-8656, Japan.

type algorithm for finding the entire sequence of maximum degree of minors whereas previous one ([8,13]) find it of specified order $k$.

Murota[13,14] shows that "combinatorial relaxation" can be used to find the maximum degree of minors in rational matrices. Combinatorial relaxation algorithm is the framework due to Murota[12]. Generally, combinatorial relaxation type algorithm constructed as follows:

**Step 1:** Consider the combinatorial relaxation (e.g. matching on bipartite graph) and solve it.
**Step 2:** Test wether combinatorial relaxation is equal to optimal solution or not without knowing optimal solution itself.
**Step 3:** In the case of incorrectness, modify the problem and go back to Step 1.

Combinatorial relaxation algorithm has practical efficency like "graph theoretic approach(Lin[11])" and accuracy like numerical method.

This paper is constructed as follows: First, the preliminaries such as the definition of maximum degree of minors in section 2. Next, we propose the combinatorial relaxation algorithm for the entire sequence of maximum degree of minors in section 3. Then, we analyze time complexity of the proposed algorithm theoretically in section 5 and conclude this paper in section 6.

## 2 Preliminaries

### 2.1 Rational Matrices and Maximum Degree of Minors

Let $A(x)$ be a rational matrix over a field $F$, i.e., for all row $i$ and column $j$, $A_{ij}(x)$ is rational function in $x$ over $F$. Typically, $F$ is the real number field $\mathbb{R}$ or the rational number field $\mathbb{Q}$. We define the degree of rational function $f(x) = p(x)/q(x)$ by $\deg f = \deg p - \deg q$. If $f(x) = 0$, we denote $\deg f = -\infty$ by convention. The maximum degree of minors of order $k$ is defined as follows:

$$\delta_k(A) := \max\{\deg \det A[I,J] \mid I \subseteq \mathrm{Row}(A),\ J \subseteq \mathrm{Col}(A),\ |I| = |J| = k\}. \tag{1}$$

$\delta_0(A)$ is defined as $\delta_0(A) = 0$. $A[I,J]$ denotes the submatrix of $A(x)$ with row-set $I$ and column-set $J$.

### 2.2 Smith–McMillan Form at Infinity

We call a rational function $f(x)$ proper if $\deg f \leq 0$. A rational matrix is said to be a proper rational matrix if its all entries are proper. We call a square proper rational matrix $U(x)$ biproper if it is invertible and $U^{-1}(x)$ is proper. For a proper rational matrix $U(x)$, following two conditions are equivalent(see e.g. [16]):

– $U(x)$ is a biproper rational matrix;

– $\det U(x)$ is a nonzero constant.

Any rational matrix can be brought into the Smith–McMillan form at infinity(Verghese–Kailath[18]).

**Proposition 1** *Let $A(x)$ be a rational matrix and denote its rank by $r$. There exist biproper rational matrices $U(x)$ and $V(x)$ such that*

$$U(x)A(x)V(x) = \begin{pmatrix} \Gamma(x) & O \\ O & O \end{pmatrix},$$

*where*

$$\Gamma(x) = \mathrm{diag}(x^{t_1}, \ldots, x^{t_r}),$$

*and $t_k$ $(k = 1, \ldots, r)$ are nonincreasing integers, i.e., $t_1 \geq \cdots \geq t_r$. Moreover, $t_k$ satisfies following equation:*

$$t_k = \delta_k(A) - \delta_{k-1}(A) \quad (k = 1, \ldots, r).$$

The transformation such as: $U(x)A(x)V(x)$ is called biproper equivalence transformations if $U(x)$ and $V(x)$ are biproper rational matrices. It is important fact that $\delta_k(A)$'s are invariant under the biproper equivalence transformations.

### 2.3 Combinatorial Relaxation

Let $A(x)$ be a rational matrix with row set $R$ and the column set $C$. We define $G(A) = (R \cup C, E(A), c)$ be a bipartite graph assciated with $A(x)$ where arc set $E(A)$ and weight $c : E(A) \to \mathbb{Z}$ is defined as follows:

$$E(A) = \{(i, j) \mid i \in R,\ j \in C,\ A_{ij}(x) \neq 0\}, \tag{2}$$
$$c(i, j) = \deg A_{ij} \quad ((i, j) \in E(A)). \tag{3}$$

We define $\hat{\delta}_k(A)$ be a maximum weight matching of size $k$ in $G(A)$, i.e.,

$$\hat{\delta}_k(A) = \max\{c(M) \mid M : \text{a matching in } G(A),\ |M| = k\} \tag{4}$$

where $c(M) = \sum_{(i,j)\in M} c(i, j)$. If there are no matching of size $k$ in $G(A)$, we denote $\hat{\delta}_k(A) = -\infty$. $\hat{\delta}_k(A)$ can be computed efficiently by Hungarian method[10].

$\hat{\delta}_k(A)$ plays the role as a combinatorial relaxation of $\delta_k(A)$ because following inequality holds([13, Theorem 3]):

$$\delta_k(A) \leq \hat{\delta}_k(A). \tag{5}$$

The duality of linear programming problem plays an essential role for testing whether $\delta_k(A) = \hat{\delta}_k(A)$ or $\delta_k(A) < \hat{\delta}_k(A)$. The primal-dual pair of linear programming problems are defined as follows:

$$\begin{aligned}
\mathrm{PLP(A,k)}: \ \text{maximize} \quad & \sum_{e\in E} c_e\xi_e, \\
\text{subject to} \quad & \sum_{\partial e \ni i} \xi_e \le 1 \quad (i \in V), \\
& \sum_{e\in E} \xi_e = k, \\
& \xi_e \ge 0 \quad (e \in E);
\end{aligned} \tag{6}$$

$$\begin{aligned}
\mathrm{DLP(A,k)}: \ \text{minimize} \quad & \sum_{i\in R} p_i + \sum_{j\in C} q_j + kt \quad (\equiv \pi(p,q,t)), \\
\text{subject to} \quad & p_i + q_j + t \ge c_{ij} \quad ((i,j) \in E), \\
& p_i \ge 0 \quad (i \in R), \\
& q_j \ge 0 \quad (j \in C).
\end{aligned} \tag{7}$$

$\mathrm{PLP}(A,k)$ and $\mathrm{DLP}(A,k)$ has an integral optimal solution, and their optimal values are equall to $\hat{\delta}_k(A)$.

We define active rows $I^* \subseteq R$ and columns $J^* \subseteq C$ as follows:

$$I^* = I^*(p) = \{i \in R \mid p_i > 0\}, \tag{8}$$
$$J^* = J^*(q) = \{j \in C \mid q_j > 0\}. \tag{9}$$

Moreover, we define tight coefficient matrix $A^*$ as follows:

$$A^*_{ij} = \lim_{x\to\infty} x^{-p_i-q_j-t} A_{ij}(x). \tag{10}$$

Proposition 2 enables us to test whether $\delta_k(A) = \hat{\delta}_k(A)$ or not without knowing $\delta_k(A)$.

**Proposition 2 ([13] Theorem 4)** *Let $(p,q,t)$ be an optimal dual solution, $I^*$ and $J^*$ be the active rows and columns defined by (8) and (9), and $A^*$ be the tight coefficient matrix defined by (10). Then $\delta_k(A) = \hat{\delta}_k(A)$ if and only if the following four conditions are satisfied:*

(r1) rank $A^*[R,C] \ge k$,
(r2) rank $A^*[I^*,C] = |I^*|$,
(r3) rank $A^*[R,J^*] = |J^*|$,
(r4) rank $A^*[I^*,J^*] \ge |I^*| + |J^*| - k$.

### 2.4 Valuated Bimatroids

The concept of a valuated bimatroid is defined by Murota[15] as a variant of a valuated matroid (Dress–Wenzel[4,5]). And the valuation $w$ of valuated bimatroids represents the abstract property of degree of minors in rational matrices.

**Definition 1 (Valuated Bimatroid)** Let $R$ and $C$ be disjoint finite sets. And let $w : 2^R \times 2^C \to \mathbb{R} \cup \{-\infty\}$ be a map. We define $(R, C, w)$ to be a valuated bimatroid if and only if $w$ satisfies $w(\emptyset, \emptyset) \neq -\infty$, $w(I, J) = -\infty$ for $(I, J) \notin S$ where

$$S = \{(I, J) \mid |I| = |J|,\ I \subseteq R,\ J \subseteq C\},$$

and the exchange properties (E1) and (E2) below for any $(I, J) \in S$ and $(I', J') \in S$:

(E1) For any $i' \in I' \setminus I$, (a1) or (b1) holds:
(a1) $\exists j' \in J' \setminus J : w(I, J) + w(I', J') \leq w(I + i', J + j') + w(I' - i', J' - j')$,
(b1) $\exists i \in I \setminus I' : w(I, J) + w(I', J') \leq w(I - i + i', J) + w(I' - i' + i, J')$.
(E2) For any $j \in J \setminus J'$, (a2) or (b2) holds:
(a2) $\exists i \in I \setminus I' : w(I, J) + w(I', J') \leq w(I - i, J - j) + w(I' + i, J' + j)$,
(b2) $\exists j' \in J' \setminus J : w(I, J) + w(I', J') \leq w(I, J - j + j') + w(I', J' - j' + j)$.

We define $S_k \subseteq S$, $\delta_k \in \mathbb{Z}$, $M_k \subseteq S_k \times S_k$ as follows:

$$\begin{aligned} S_k &= \{(I, J) \mid |I| = |J| = k,\ I \subseteq R,\ J \subseteq C\}, \\ \delta_k &= \max\{w(I, J) \mid (I, J) \in S_k\}, \\ M_k &= \{(I, J) \in S_k \mid w(I, J) = \delta_k\}. \end{aligned}$$

**Proposition 3 (Murota[15] Theorem 1)** *The following inequality holds:*

$$\delta_{k-1} + \delta_{k+1} \leq 2\delta_k, \quad (k = 1, 2, \ldots, r - 1) \tag{11}$$

Since $w(I, J) := \deg \det A[I, J]$ and $w(I, J) := \max\{c(M) \mid M \subseteq E(A), \partial M = I \cup J\}$ define valuated bimatroids $(R, C, w)$, Proposition 3 means that $\{\delta_k(A)\}_{k=0}^r$ and $\{\hat{\delta}_k(A)\}_{k=0}^r$ have concavity.

**Proposition 4 (Murota[15] Theorem 2)** *For any $(I_k, J_k) \in M_k$ with $k = 1, \ldots, r - 1$, there exists $(I_l, J_l) \in M_l$ $(0 \leq l \leq r, l \neq k)$ such that $(\emptyset =) I_0 \subseteq I_1 \subseteq \cdots \subseteq I_{k-1} \subseteq I_k \subseteq I_{k+1} \subseteq \cdots \subseteq I_r$ and $(\emptyset =) J_0 \subseteq J_1 \subseteq \cdots \subseteq J_{k-1} \subseteq J_k \subseteq J_{k+1} \subseteq \cdots \subseteq J_r$.*

## 3 Combinatorial Relaxation Algorithm

In this section, we propose a combinatorial relaxation algorithm for computing the entire sequence of maximum degree of minors $\{\delta_k(A)\}_{k=1}^r$ where $A(x)$ be a Laurent polynomial matrix and $r = \operatorname{rank} A(x)$. The proposed algorithm has 4 properties as follows:

- In the end of the algorithm, $\delta_k(A) = \hat{\delta}_k(A)$ is satisfied for all $k = 1, \ldots, r$;
- The number of the constant matrix which need to compute its rank for testing the tightness is only one, whereas previous one need 4 matrices;
- In the step of matrix modification, the size of matrix multiplication is reduced;
- During $\delta_k(A)$ is increasing(decreasing) at same rate, we can skip the computation.

The outline of the algorithm is summarized as follows.

**Outline of Algorithm for Computing** $\{\delta_k(A)\}_{k=1}^r$

**Step 0:** Compute $\delta_1(A)$ and set $k := 1$.
**Step 1:** Find an optimal solution $(p, q, t)$ of DLP$(A, k)$.
**Step 2:** Test whether $\delta_{k+1}(A) = \hat{\delta}_{k+1}(A)$ or not by using $(p, q, t)$. If equality holds, go to Step 4.
**Step 3:** Modify $A(x)$ to $A'(x)$, and go back to Step 1.
**Step 4:** Output optimal values, update $k$ and go back to Step 1.

We detail the algorithm description in 3.5.

For initialization, we find $(i, j) \in R \times C$ such that a maximizer of $\deg A_{ij}$. Since $\delta_1(A) = \hat{\delta}_1(A) = \deg A_{ij}$ holds, we define $I_1^* = \{i\}$, $J_1^* = \{j\}$, $M_1 = \{(i, j)\}$.

### 3.1 Construction of an Optimal Dual Solution

When we execute Step 1, following conditions hold:

- $\delta_l(A) = \hat{\delta}_l(A) \quad (l = 1, 2, \ldots, k)$;
- $\deg \det A[I_k^*, J_k^*] = \sum_{(i,j)\in M_k} \deg A_{ij} = \delta_k(A)$;
- $I_k^* = \partial^+ M_k$, $J_k^* = \partial^- M_k$.

In fact, we don't need to store $I_k^*$ and $J_k^*$ because they can be easily constructed from $M_k$. But they are helpful for describing and comprehending the proposed algorithm.

In this step, we introduce the method to construct an optimal dual solution. As stated in Iwata–Takamatsu[9], we can obtain it by solving shortest path problem on the auxiliary graph. Consider an auxiliary graph $G_{M_k} = (V, E, \gamma)$ associated with $M_k$ where

$$V = R \cup C \cup \{u^+\} \cup \{u^-\},$$
$$E = E(A) \cup M^\circ \cup W^+ \cup W^-.$$

Here, $u^+$ and $u^-$ are new vertices and

$$\begin{aligned} E(A) &= \{(i,j) \mid i \in R,\ j \in C,\ A_{ij}(x) \neq 0\}, \\ M^\circ &= \{(j,i) \mid (i,j) \in M_k\}, \\ W^+ &= \left\{(u^+, i) \mid i \in R \setminus I_k^*\right\}, \\ W^- &= \left\{(j, u^-) \mid j \in C \setminus J_k^*\right\} \cup \{(u^-, j) \mid j \in C\}. \end{aligned}$$

We define the arc length $\gamma : E \to \mathbb{Z}$ by

$$\gamma(i,j) = \begin{cases} -\deg A_{ij} & ((i,j) \in E(A)), \\ \deg A_{ji} & ((i,j) \in M^\circ), \\ 0 & ((i,j) \in W^+ \cup W^-). \end{cases}$$

Let $\varphi(i)$ be the length of a shortest path from $u^+$ to $i \in V$ with arc length $\gamma$ in $G_{M_k}$. If there is no path from $u^+$ to $i \in V$, we put $\varphi(i) = +\infty$. We define $(p,q,t)$ as follows:

$$\begin{aligned} p_i &= \varphi(i) && (i \in R), \\ q_j &= \varphi(u^-) - \varphi(j) && (j \in C), \\ t &= -\varphi(u^-). \end{aligned} \tag{12}$$

If there is a path from $u^+$ to $u^-$, $(p,q,t)$ is an optimal dual solution of $\mathrm{DLP}(A,k)$ as Lemma 1. In other case, i.e., there is no path from $u^+$ to $u^-$, $k = \operatorname{rank} A(x)$ holds ($k = \text{term-rank } A(x) \geq \operatorname{rank} A(x)$).

Lemma 1 is slightly different version of [9, Lemma 3].

**Lemma 1** *Let $M_k$ be a maximum weight matching with $|M_k| = k$ in $G(A)$. $(p,q,t)$ defined as the equation (12) with $G_{M_k}$ is an optimal dual solution of* $\mathrm{DLP}(A,k)$. *Furthermore, if a weight function is integer valued, then $(p,q)$ is an integral solution.*

In this step, we can adopt “ reweighting ” like Johnson's algorithm (see e.g. [3]) for all-pairs shortest path. And we can use the newest optimal dual variable for reweighting [6,17] in this case.

### 3.2 Test for Tightness

Testing for tightness, we adopt Theorem 1 instead of Proposition 2. Following lemma is essential for Theorem 1 and other theorems.

**Lemma 2** *Under the condition of Lemma 1, $(p,q,t)$ is an optimal dual solution of* $\mathrm{DLP}(A, k+1)$.

*Proof* Since $(p, q, t)$ is obviously a feasible solution of DLP$(A, k+1)$, we prove only optimality.

The objective function of DLP$(A, k+1)$ is expressed as:

$$\pi_{k+1}(p, q, t) = \sum_{i \in R} p_i + \sum_{j \in C} q_j + (k+1)t = \pi_k(p, q, t) + t.$$

On the other hand, since dual variable $t = -\varphi(u^-)$ is defined as the maximum length of augmenting path, the opimal value of DLP$(A, k+1)$ is $t$ more than the optimal value of DLP$(A, k)$. Therefore, $(p, q, t)$ is one of the optimal dual solution of DLP$(A, k+1)$. □

**Theorem 1** *Suppose that $\delta_k(A) = \hat{\delta}_k(A)$ holds and $M_k$ is a maximum weight matching of size $k$ in $G(A)$. Let $(p, q, t)$ be a dual variable defined as (12) from $M_k$ and $A^*$ be a tight coefficient matrix defined as (10). Then, following 2 conditions are equivallent:*

- $\delta_{k+1}(A) = \hat{\delta}_{k+1}(A) = \delta_k(A) + t$.
- rank $A^* \geq k+1$.

*Proof* Since $\delta_k(A) = \hat{\delta}_k(A)$ holds, following 4 conditions are satisfied by Proposition 2:

(r1) rank $A^*[R, C] \geq k$,
(r2) rank $A^*[I^*, C] = |I^*|$,
(r3) rank $A^*[R, J^*] = |J^*|$,
(r4) rank $A^*[I^*, J^*] \geq |I^*| + |J^*| - k$.

On the other hand, by Lemma 2 and Proposition 2, $\delta_{k+1}(A) = \hat{\delta}_{k+1}(A)$ holds if and only if following 4 conditions are all satisfied:

(r1)' rank $A^*[R, C] \geq k+1$,
(r2)' rank $A^*[I^*, C] = |I^*|$,
(r3)' rank $A^*[R, J^*] = |J^*|$,
(r4)' rank $A^*[I^*, J^*] \geq |I^*| + |J^*| - k - 1$.

Clearly, (r2)', (r3)' and (r4)' immediately derives from (r2), (r3) and (r4). Therefore, $\delta_{k+1}(A) = \hat{\delta}_{k+1}(A) \iff$ rank $A^* \geq k+1$. □

Since Theorem 1 holds, we can test whether $\delta_{k+1}(A) = \hat{\delta}_{k+1}(A)$ or not by calculation of rank $A^*$.

### 3.3 Matrix Modification

Theorem 1 enable us to consider only the case rank $A^* = k$ whereas existing algorithms cope with 4 cases. When we implement Step 3, following conditions are satisfied:

1. rank $A^* = k$,

2. rank $A^*[I_l^*, J_l^*] = l \quad (l = 1, 2, \ldots, k)$.

Therefore, there exists $m \times m$ constant matrix $U$ such that

(U1) $U[I_k^*, I_k^*]$ and $U[R \setminus I_k^*, R \setminus I_k^*]$ are identity matrices,
(U2) $U[I_k^*, R \setminus I_k^*] = O$,
(U3) term-rank $UA^* = k$.

Namely, we can interpret (U1), (U2) and (U3) as

$$UA^* = \begin{pmatrix} I & O \\ \tilde{U} & I \end{pmatrix} A^* = \begin{pmatrix} A^*[I_k^*, C] \\ O \end{pmatrix}$$

by appropriate permutation. Where $I$ denote an identity matrix of suitable order and $\tilde{U} = U[R \setminus I_k^*, I_k^*]$ is an constant matrix.

We can modify the matrix $A(x)$ by the constant matrix $U$ such that (U1), (U2) and (U3) are satisfied. Let $A'(x)$ be the modified matrix, then we define

$$A'(x) = \mathrm{diag}(x; p) \cdot U \cdot \mathrm{diag}(x; -p) \cdot A(x). \tag{13}$$

The right side of the equation (13) can be computed by partial matrix multiplication as follows:

$$A'(x) = \mathrm{diag}(x; p) \cdot \begin{pmatrix} I & O \\ \tilde{U} & I \end{pmatrix} \cdot \mathrm{diag}(x; -p) \cdot A(x) \tag{14}$$

$$= \begin{pmatrix} I & O \\ \tilde{U} \cdot \mathrm{diag}(x; -p_k^*) & I \end{pmatrix} \begin{pmatrix} A[I_k^*, C](x) \\ A[R \setminus I_k^*, C](x) \end{pmatrix} \tag{15}$$

where $p_k^* = (p_i \mid i \in I_k^*)$. Therefore,

$$A'[I_k^*, C] = A[I_k^*, C](x), \tag{16}$$

$$A'[R \setminus I_k^*, C] = \tilde{U} \cdot \mathrm{diag}(x : -p_k^*) \cdot A[I_k^*, C](x) + A[R \setminus I_k^*, C]. \tag{17}$$

We claim that this modification makes sense as stated in Theorem 2.

**Theorem 2** *A matrix $A'(x)$ defined as (13) has following 4 properties:*

1. $\delta_l(A') = \delta_l(A) \quad (l = 1, \ldots, r)$;
2. $\deg \det A'[I_k^*, J_k^*] = \delta_k(A')$;
3. $\hat{\delta}_l(A') = \delta_l(A') \quad (l = 1, \ldots, k)$;
4. $\hat{\delta}_{k+1}(A') < \hat{\delta}_{k+1}(A)$.

*Proof*

1. Since $U(x) := \mathrm{diag}(x; p) \cdot U \cdot \mathrm{diag}(x : -p)$ is a biproper rational matrix, the equation are satisfied.
2. $A'[I_k^*, J_k^*](x) = A[I_k^*, J_k^*](x)$ holds from (16). Therefore,

$$\deg \det A'[I_k^*, J_k^*] = \deg \det A[I_k^*, J_k^*] = \delta_k(A) = \delta_k(A').$$

3. By Proposition 4 and the property 2 of this theorem, $\hat{\delta}_l(A[I_k^*, J_k^*]) = \hat{\delta}_l(A)$ and $\hat{\delta}_l(A'[I_k^*, J_k^*]) = \hat{\delta}_l(A')$ hold for all $l \leq k$. From these equations and $A[I_k^*, J_k^*] = A'[I_k^*, J_k^*]$, we obtain

$$\hat{\delta}_l(A') = \hat{\delta}_l(A'[I_k^*, J_k^*]) = \hat{\delta}_l(A[I_k^*, J_k^*]) = \hat{\delta}_l(A) = \delta_l(A) = \delta_l(A').$$

4. First, we prove that an optimal dual solution $(p, q, t)$ of $\mathrm{DLP}(A, k+1)$ is feasible in $\mathrm{DLP}(A', k+1)$. We define a rational matrix $F(x)$ as follows:

$$F(x) := x^{-t}\mathrm{diag}(x : -p) \cdot A'(x) \cdot \mathrm{diag}(x : -q).$$

Using (13), we obtain

$$\begin{aligned} F(x) &= x^{-t}U \cdot \mathrm{diag}(x; -p) \cdot A(x) \cdot \mathrm{diag}(x; -q) \\ &= U \cdot (A^* + A^\infty(x)), \end{aligned}$$

where $A^\infty(x)$ is an strictly proper rational matrix, i.e., all entries of $A^\infty(x)$ have negative degrees. From the equation, $\deg F_{ij} \leq 0$ holds. Then, using $\deg F_{ij} = \deg A'_{ij} - p_i - q_j - t$, we obtain $p_i + q_j + t \leq \deg A'_{ij}$ for all $i \in R$ and $j \in C$. Threfore, $(p, q, t)$ is a feasible solution of $\mathrm{DLP}(A', k+1)$.

Next, we prove that $(p, q, t)$ is not optimal. It is sufficient that there exists another feasible solution $(p', q', t')$ such that $\pi_k(p', q', t') < \pi_k(p, q, t)$. We define $(p', q', t')$ as follows:

$$\begin{aligned} p'_i &= \begin{cases} p_i + 1 & (i \in I_k^*), \\ p_i & (i \notin I_k^*), \end{cases} \\ q'_j &= q_j \quad (j \in C) \\ t' &= t - 1. \end{aligned}$$

Clearly, $p'_i \geq 0$ and $q'_j \geq 0$ hold for all $i \in R$ and $j \in C$. We confirm that $p'_i + q'_j + t \geq \deg A'_{ij}$ holds for all $i \in R$ and $j \in C$. If $i \in I_k^*$, using feasibility of $(p, q, t)$ in $\mathrm{DLP}(A', k)$, we obtain

$$p'_i + p'_j + t = p_i + 1 + q_j + t - 1 = p_i + q_j + t \geq \deg A'_{ij}.$$

In other case, i.e., $i \notin I_k^*$, $p_i + q_j + t > \deg A'_{ij}$ is satisfied, then

$$p'_i + p'_j + t = p_i + q_j + t - 1 \geq \deg A'_{ij}$$

holds. Furthermore, by definition of $(p', q', t')$, $\pi_k(p', q', t')$ is one less than $\pi_k(p, q, t)$.

□

### 3.4 Outputs and Updates

In this step, we output $\delta_l(A)$ $(l = k+1, \ldots, r^*)$ and update $k$ to $r^*$, where $r^* := \operatorname{rank} A^*$.

**Lemma 3** *If $k < r^*$ holds, then following equation is satisfied:*

$$\delta_{r^*}(A) = \delta_k(A) + (r^* - k)t. \tag{18}$$

*Proof* First, we prove that $\delta_{r^*}(A) \geq \delta_k(A) + (r^* - k)t$ holds. Since $\operatorname{rank} A^* = r^*$, there exists $I \subseteq R$, $J \subseteq C$ such that $|I| = |J| = r^*$ and $A^*[I, J]$ is nonsingular. Then, we obtain

$$\begin{aligned}
\delta_{r^*} &\geq \deg \det A[I, J] \\
&= \deg \left( x^{r^* t} \cdot \det \operatorname{diag}(x; p_I) \cdot \det(A^*[I, J] + A^{\infty}[I, J](x)) \cdot \det \operatorname{diag}(x; p_J) \right) \\
&= \deg \det(A^*[I, J] + A^{\infty}[I, J](x)) + \delta_k(A) + (r^* - k)t \\
&\geq \delta_k(A) + (r^* - k)t.
\end{aligned}$$

On the other hand, $\delta_{r^*} \leq \delta_k(A) + (r^* - k)t$ is easily proved by concavity. Therefore, the equality holds. □

Theorem 3 allows us to output optimal values.

**Theorem 3** *Suppose that $\delta_k(A) = \hat{\delta}_k(A)$ holds and $(p, q, t)$ is an optimal solution of* DLP$(A, k)$ *and* DLP$(A, k+1)$. *Then, following equality holds:*

$$\delta_l(A) = \hat{\delta}_l(A) = \delta_k(A) + (l - k)t \quad (l = k+1, \ldots, r^*). \tag{19}$$

*Moreover, $\delta_{r^*+1}(A) < \delta_k(A) + (r^* + 1 - k)t$ holds.*

*Proof* First, we prove (19). Since Theorem 1, Lemma 3 and Proposition 3 hold, following equality is satisfied:

$$\delta_l(A) = \delta_k(A) + (l - k)q \quad (l = k+1, \ldots, r^*). \tag{20}$$

On the other hand, $\hat{\delta}_l(A) \leq \hat{\delta}_k(A) + (l - k)t$ holds because of concavity. Using this inequality and $\delta_l(A) \leq \hat{\delta}_l(A)$, we obtain

$$\hat{\delta}_l(A) = \delta_k(A) + (l - k)t \quad (l = k+1, \ldots, r^*).$$

Therefore, the equation (19) holds.

Next, We prove that $\delta_{r^*+1}(A) < \delta_k(A) + (r^* + 1 - k)t$ holds. If $(p, q, t)$ is not an optimal dual solution of DLP$(A, r^* + 1)$, the inequality is satisfied. On the other case, i.e., $(p, q, t)$ is an optimal solution of DLP$(A, r^* + 1)$, we can adopt Theorem 1. Since $\operatorname{rank} A^* = r^*$, we obtain

$$\delta_{r^*+1}(A) < \hat{\delta}_{r^*+1}(A) = \delta_k(A) + (r^* + 1 - k)t.$$

□

After we output $\{\delta_l(A)\}_{l=k+1}^{r^*}$, we must construct $M_{r^*}$ such that

$$\sum_{(i,j)\in M_{r^*}} \deg A_{ij} = \deg \det A[\partial^+ M_{r^*}, \partial^- M_{r^*}] = \delta_{r^*}(A)$$

holds. First, we construct $I^*_{r^*} = \partial^+ M_{r^*}$ and $J^*_{r^*} = \partial^- M_{r^*}$. We can do that with calculation of rank $A^*$ as follows:

0. $M' := M_k$, $R' := \mathrm{Row}(A^*)$.
1. Repeat following steps for $k$ times.
   (a) Choose $(i,j) \in M'$.
   (b) Conduct row elimination for all rows in $R' \setminus \{i\}$ by a row $i$.
   (c) $M' := M' \setminus \{(i,j)\}$, $R' := R' \setminus \{i\}$.
2. Execute Gaussian elimination for $A^*[R', C]$.
3. Define $I^*_{r^*}$ and $J^*_{r^*}$ as follows:

$$I^*_{r^*} = \{i \in R \mid \exists j,\ A^*_{ij} \neq 0\},$$
$$J^*_{r^*} = \{j \in C \mid \exists i \in I^*_{r^*},\ j = \min\{j' \mid A^*_{ij} \neq 0\}\}.$$

Then, $M_{r^*}$ is an optimal solution of weighted bipartite matching problem on $G = (I^*_{r^*} \cup J^*_{r^*}, E, \gamma)$ where

$$E = \{(i,j) \mid i \in I^*_{r^*}, j \in J^*_{r^*}, A_{ij}(x) \neq 0\},$$
$$\gamma(i,j) = \deg A_{ij}(x) \quad ((i,j) \in E).$$

Using augmenting path, we can construct $M_{r^*}$ efficiently.

### 3.5 Algorithm Description

*Step 0: Initialization*

Let a row $i^*$ and a column $j^*$ be a maximizer of $\deg A_{ij}$, then initialize $I^*_1$, $J^*_1$, $M_1$ as:

$$I^*_1 = \{i^*\}, \quad J^*_1 = \{j^*\}, \quad M_1 = \{(i^*, j^*)\}.$$

Output $\deg A_{i^*j^*}$ as $\delta_1(A)$, and set $k = 1$, then go to Step 1.

*Step 1: Construction of an Optimal Dual Solution*

1. Construct an optimal dual solution from $M_k$ by solving " shortest path " on directed graph $G_{M_k}$. If there is no path, set rank $A = k$ and halt.
2. If $t < (k+1)d_{\min} - \delta_k(A)$, set rank $A = k$ and halt.

*Step 2: Test for Tightness*

1. Make $A^*$ from the optimal dual solution.
2. Compute rank $A^*$ by Gaussian elimination.
3. If rank $A^* = k$, go to Step 3. In other case, go to Step 4.

*Step 3: Matrix Modification*

1. Solve $\tilde{U} \cdot A^*[I_k, J_k] + A^*[R \setminus I_k, J_k] = 0$ to obtain $\tilde{U}$.
2. Construct $A'(x)$as follows($p_k^* := (p_i \mid i \in I_k^*)$):

$$A'[I_k^*, C](x) = A[I_k^*, C](x),$$

$$A'[R \setminus I_k^*, C](x) = \tilde{U} \cdot \mathrm{diag}(x; -p_k^*) \cdot A[I_k^*, C](x) + A[R \setminus I_k^*, C](x).$$

3. Go to Step 1.

*Step 4: Outputs and Updates*

1. Output as follows:

$$\delta_l(A) = \delta_k(A) + (l-k)t \quad (l = k+1, \ldots, r^*)$$

2. If $r^* = \min(m, n)$, halt.
3. Construct $I_{r^*}$, $J_{r^*}$.
4. Compute $M_{r^*}$ by solving weighted matching.
5. Set $k = r^*$ and go to Step 1.

## 4 Example

In this section, We implement the calculation along the proposed algorithm.

*Example 1* Let $A(x)$ be a matrix pencil as follows:

$$A(x) = \begin{pmatrix} x+1 & x+3 & x+2 \\ x+2 & x+6 & x+4 \\ x+1 & x+3 & x+1 \\ 2 & 1 & 3 \end{pmatrix}. \tag{21}$$

Then, the set of rows $R = \{r_1, r_2, r_3, r_4\}$ and the set of columns $C = \{c_1, c_2, c_3\}$ are defined (The rows run along up to down, and the columns do left to right).

*Step 0: Initialization*

From the equation (21), one of the maximizer of $\deg A_{ij}(A)$ is $(r_1, c_1)$. Therefore, $I_1^* = \{r_1\}$, $J_1^* = \{c_1\}$, $M_1 = \{(r_1, c_1)\}$ and $\delta_1(A) = 1$. Set $k = 1$ and go to step 1.

*Step 1: Construction of Optimal Dual Solutions*

As in section 3.1, the auxiliary graph $G_{M_1}$ is defined as Fig. 1.

In $G_{M_1}$, $\varphi(i)$, the length of the shortest path from $u^+$ to $i \in V$, is computed as follows:

$$\varphi(r_1) = \varphi(r_2) = \varphi(r_3) = \varphi(r_4) = 0,$$
$$\varphi(c_1) = \varphi(c_2) = \varphi(c_3) = \varphi(u^-) = -1.$$

Therefore, an optimal dual solution is

$$p = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}, \quad q = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}, \quad t = 1. \tag{22}$$

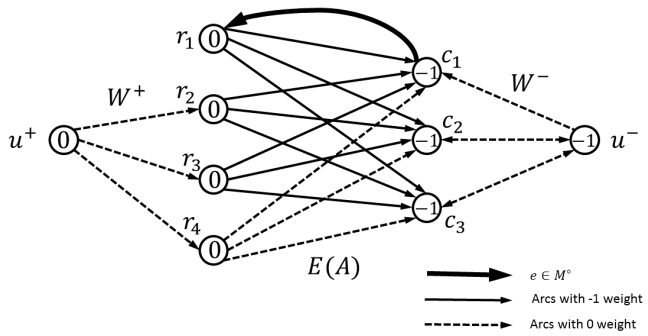


**Fig. 1** Auxiliary graph $G_{M_1}$

*Step 2: Test for Tightness*
With the optimal dual solution in equation (22), $A^*$ is defined as follows:

$$A^* = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}. \tag{23}$$

Since rank $A^* = 1 = k$, go to Step 3.

*Step 3: Matrix Modification*
The $3 \times 1$ matrix $\tilde{U}$ satisfies following equation:

$$\tilde{U} \cdot 1 + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = O.$$

Therefore, $\tilde{U} = \begin{pmatrix} -1 & -1 & 0 \end{pmatrix}^\top$. Then, rational matrix $A(x)$ is modified as follows:

$$\begin{aligned} A'(x) &= \begin{pmatrix} A[I_1^*, C] \\ \tilde{U} \cdot x^{p_{r_1}} \cdot A[I_1^*, C](x) + A[R \setminus I_1^*, C](x) \end{pmatrix} \\ &= \begin{pmatrix} x+1 & x+3 & x+2 \\ 1 & 3 & 2 \\ 0 & 0 & -1 \\ 2 & 1 & 3 \end{pmatrix}. \end{aligned}$$

Next, we implement same step (Step 1 and 2) to confirm that the modification makes sense.

*Step 1 and 2*
In the same way, we can obtain the optimal dual solution:

$$p = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix}, \quad q = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}, \quad t = 0. \tag{24}$$

Then, $A^*$ is modified as follows:

$$A^* = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 3 & 2 \\ 0 & 0 & -1 \\ 2 & 1 & 3 \end{pmatrix}. \tag{25}$$

To compute rank $A^*$, we execute the Gaussian Elimination as stated in section 3.2:

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 3 & 2 \\ 0 & 0 & -1 \\ 2 & 1 & 3 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 2 & 1 \\ 0 & 0 & -1 \\ 0 & -1 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 3 \\ 0 & 0 & -1 \\ 0 & -1 & 1 \end{pmatrix} \to \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & -1 & 1 \end{pmatrix}. \tag{26}$$

Since rank $A^* = 3$, go to Step 4.

*Step 4: Outputs and Updates*
From the equation (26), $I_3^* = \{r_1, r_3, r_4\}$ and $J_3^* = C$. Solving the weighted bipartite matching problem over $G = (I_3^* \cup J_3^*, E_3, c)$ (Fig. 2) where $E_3 \subseteq E$ is defined as $E_3 = E(A) \cap (I_3^* \times J_3^*)$, we obtain $M_3 = \{(r_1, c_1), (r_3, c_3), (r_4, c_2)\}$. As a result, $\delta_2(A)$ and $\delta_3(A)$ is calculated as follows:

$$\delta_2(A) = \delta_1(A) + 1 \cdot 0 = 1,$$
$$\delta_2(A) = \delta_1(A) + 2 \cdot 0 = 1.$$

Since $k = \text{rank } A^* = 3$ and $|C| = 3$, the algorithm has finished.

## 5 Complexity Analysis

### 5.1 Worst Case Analysis

Fist of all, we show the definition of parameters in Table 1 and time complexity of each operation in Table 2. Time complexity of proposed algorithm is $O(dnr(n + dmr))$ as Theorem 4

**Theorem 4** *Let $A(x)$ be an $m \times n$ Laurent polynomial matrix and each parameter is defined as Table 1. Then, proposed algorithm runs in $O(dnr(n + dmr))$ time.*

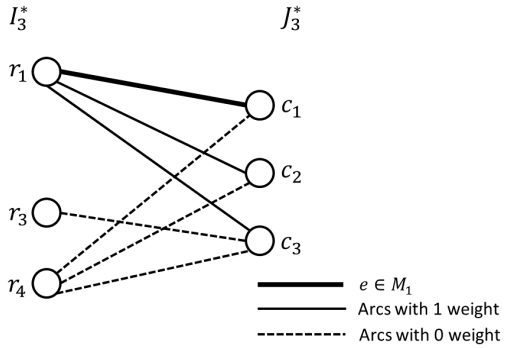


**Fig. 2** $G = (I_3^* \cup J_3^*, E_3, c)$

**Table 1** The definition of parameters

| symbol | | definition |
|---|---|---|
| $m$ | : | The number of rows |
| $n$ | : | The number of columns |
| $r$ | : | rank $A$ |
| $d_{\max}$ | : | $\max \deg A_{ij}$ |
| $d_{\min}$ | : | $\min \mathrm{ord} A_{ij}$ |
| $d$ | : | $d_{\max} - d_{\min}$ |

**Table 2** Time complexity of each operation

| Step | Operation | Time Complexity | Method |
|---|---|---|---|
| Step 0 | Initialization | $O(mn)$ | Max Search |
| Step 1 | Optimal Dual Solution | $O(dn^2r)$ | Single Source Shortest Path |
| Step 2 | Calculation of Rank | $O(dmnr^2)$ | Gaussian Elimination |
| Step 3 | Construction of $\tilde{U}$ | $O(dmr^3)$ | LU Decomposition |
| | Matrix Modification | $O(d^2mnr^2)$ | Matrix Multiplication |
| Step 4 | Construction of $M_{r^*}$ | $O(r^3)$ | Single Source Shortest Path |

*Proof* We can prove the theorem by confirming that Table 2 is correct.

In Step 0, we search the maximizer of $\deg A_{ij}$. Therefore, it can be done in $O(mn)$ time.

In Step 1, we construct the optimal dual solution by solving the single source shortest path problem. Since we can reweight the arc length to non-negative, it can be done in square time of the number of vertices($O(n+m)$). And Step 1 executes $O(dr)$ times in worst case. Because the value of $t$ decreases at least one each time and the difference between the optimal $q$ of $\mathrm{DLP}(A,1)$ and $\mathrm{DLP}(A,r)$ is $rd$ at most. Therefore, Step 1 runs in $O(dn^2r)$ time.

Step 2 runs in $O(mnr) \times O(dr)$ time. Because Gaussian elimination can be done in $O(mnr)$ time and it executes $O(dr)$ times such as Step 1.

In Step 3, We can construct $\tilde{U}$ by LU decomposition in $O(mr^2)$ time each execution. Therefore, it can be done in $O(dmr^3)$ time. In the phase of matrix modification, we execute multiplication between $(m-s)\times s$ polynomial matrix and $s\times n$ polynomial matrix if $k=s$ holds. Furthermore, we can execute multipication of polynomials with degree $d$ in $O(d\log d\log\log d)$(Cantor–Kaltofen[1]). Then, it can be done in $O(d^2mnr^2)$ if we ignore $O(\log d)$.

We solve the single source shortest path problem $(r-1)$ times in the whole of Step 4. Moreover, the arc length of auxiliary graph can be reweight to non-negative. Therefore, Step 4 runs in $O(r^3)$. □

## 6 Conclusion

The proposed algorithm runs in $O(nr(n+mr))$ if we assume $d$ is constant. It is more efficient than previous algorithm(Iwata–Murota–Sakuto[8]) runs in $O(mn^2r)$.

**Acknowledgements** [MEMO]:あとでかく

## References

1. Cantor, D. G., Kaltofen, E.: On fast multiplication of polynomials over arbitrary algebras. Acta Informatica. **28**, 693–701 (1991)
2. Commault, C., Dion, J.M.: Structure at infinity of linear multivariable systems: A geometric approach. IEEE Trans. Automat. Control. **AC-27**, 693–696 (1982)
3. Cormen, T. H., Leiserson, C. E., Rivest, R. L., Stein, C.: Introduction to Algorithms. The MIT Press. Cambridge, second edition (2001)
4. Dress, A. W. M., Wenzel, W.: Valuated matroid: A new look at the greedy algorithm. Appl. Math. Lett. **3**(2), 33–35 (1990)
5. Dress, A. W. M., Wenzel, W.: Valuated matroids. Adv. Math. **93**, 214–250 (1992)
6. Edmonds, J., Karp, R. M.: Theoretical improvements in algorithmic efficiency for network flow problems. Journal of ACM. **19**, 248–264 (1972)
7. Gear, C. W.: Differential algebraic equations, indeces, and integral algebraic equations. SIAM J. Number. Anal. **27**, 1527–1534 (1990)
8. Iwata, S., Murota, K., Sakuta, I.: Primal-dual combinatorial relaxation algorithm for the maximum degree of subdeterminants. SIAM J. Sci. Comput. **17**, 993–1012 (1996)
9. Iwata, S., Takamatsu, M.: Computing the maximum degree of minors in mixed polynomial matrices via combinatorial relaxation. Algorithmica **66**, 346–368 (2013)
10. Kuhn, H.: The hungarian method for the assignment problem. Naval Research Logistics Quarterly **2**, 83–97 (1955)
11. Lin, C. T.: Structural controllability. IEEE Trans. Automat. Control **AC-19**, 201–208 (1974)
12. Murota, K.: Computing Puiseux-series solutions to determinantal equations via combinatorial relaxation. SIAM J. Comput. **19**, 1132–1161 (1990)
13. Murota, K.: Combinatorial relaxation algorithm for the maximum degree of subdeterminants: Computing Smith–Mcmillan form at infinity and structural indices in Kronecker form. Appl. Algebra Eng. Commun. Comput. **6**, 251–273 (1995)
14. Murota, K.: Computing the degree of determinants via combinatorial relaxation. SIAM J. Comput. **24**, 765–796 (1995)
15. Murota, K.: Finding optimal minors of valuated bimatroids. Appl. Math. Lett. **8**, 37–42 (1995)
16. Murota, K.: Matrices and Matroids for Systems Analysis. Springer, Berlin (2000)

17. Tomizawa, N.: On some techniques useful for solution of transportation network problems. Networks **1**, 173–194 (1971)
18. Verghese, G.C., Kailath, T.: Rational matrix structure. IEEE Trans. Autom. Control **AC-26**, 443–439 (1981)