A STRONGLY POLYNOMIAL TIME ALGORITHM FOR AN LP PROBLEM WITH A PRE-LEONTIEF COEFFICIENT MATRIX

Akira Nakayama
Fukushima University

Tsutomu Anazawa
Kurume University

Yudai Iwaki
Alps Alpine Co., Ltd.

(Received March 24, 2020; Revised August 3, 2021)

Abstract In 1991, Adler and Cosares proposed a strongly polynomial time algorithm for an LP problem with a pre-Leontief coefficient matrix and pointed out that the algorithm can be efficiently applied to a generalized transshipment problem. In their generalized transshipment problem, a given demand is satisfied at each vertex except for a distinguished one while we impose the demand condition on all the vertices. Their approach is as follows: By using Veinott’s matrix partition theorem, they partitioned the coefficient matrix into four submatrices including a Leontief submatrix, and these partitioned matrices were utilized in their algorithm. We suggest that the theorem needs more refinement. In order to clarify the suggestion, we refined the theorem to a new one by incorporating trivialities/nontrivialities of the rows and columns of a matrix whose notions were introduced by Veinott. With the help of the refined theorem, we have developed a new strongly polynomial time flow-based algorithm for a broader class of problems including their problem. In the paper by Adler and Cosares, we can not see any algorithm for finding how to divide the columns of the coefficient matrix into two sets when we partition the matrix. Given a coefficient matrix partitioned, our complexity is the same as theirs. Our main contribution is the following two: 1) The developed algorithm can also determine the feasibility of the generalized transshipment problem, and our complexity is much smaller than theirs; 2) We showed an efficient algorithm for partitioning the given coefficient matrix into such four submatrices by introducing the trivialities/nontrivialities explained above.

Keywords: Linear programming, network flow, strongly polynomial time algorithm, pre-Leontief and Leontief matrices

1. Introduction

The aim of this paper is to propose a strongly polynomial time algorithm for a linear programming problem LP with a kind of pre-Leontief coefficient matrix, which includes a generalized minimum cost flow problem in an uncapacitated network, abbreviated to GMCU, as a special case. Here, a pre-Leontief matrix, introduced by Veinott [24], is a matrix such that each column has at most one positive component. Problem GMCU, formulated below, is one of generalized flow problems in a network with gains or losses where a unit of flow starting the initial vertex of an edge enters into the terminal vertex with units of flow multiplied by a rate of a gain/loss.

The generalized flow problems have been studied for over sixty years. For the applications refer to [2] and [9], while one of the surveys is Shigeno [19]. These problems belong to a special class of LP. So, they can be solved by applying general efficient linear programming methods: the ellipsoid (Khachiyan [12]), interior point methods (Karmarkar [11], Vaidya [22]). They are roughly grouped into two classes: the generalized maximum flow and minimum cost flow problems where the former class is a special case of the latter one. On the former class, see the following remarkable literature, for example, Onaga [16], Truemper [21], Goldfarb et al. [7], and Goldfarb and Jin [6]. In addition to, a quite recent result
was proposed by Oliver and Vegh [15] which speeded up Vegh’s first strongly polynomial time algorithm [23]. We focus on describing the brief history of the latter class. Jewell [10] contributed to one of the earliest algorithms. Adler and Cosares [1] designed a strongly polynomial time algorithm in an uncapacitated network. Goldberg, Plotkin, and Tardos [5] derived the first combinatorial polynomial time algorithm, after which the following efficient algorithms follow: Tardos and Wayne [20], Goldfarb, Jin, and Lin [7], Radzic [17, 18], Fleischer and Wayne [4]. An important open problem to date is to determine whether there exists a strongly polynomial time algorithm for the latter grouped class. See also Oldham [14] for approximation algorithms.

Let us outline our discussion. We deal with the following pair of LP problems $\mathbf{P}$ and $\mathbf{D}$: the former is to minimize $\mathbf{c}^\top \mathbf{x}$ subject to $\mathbf{A} \mathbf{x} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, and the latter is to maximize $\mathbf{b}^\top \mathbf{y}$ subject to $\mathbf{A}^\top \mathbf{y} \leq \mathbf{c}$, where $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{A} \in \mathbb{R}^{m \times n}$ for some integers $m$ and $n$. Hereafter, $\mathbf{P}$ and $\mathbf{D}$ are often represented by parametric forms $\mathbf{P} (\mathbf{A}, \mathbf{b}, \mathbf{c})$ and $\mathbf{D} (\mathbf{A}, \mathbf{c}, \mathbf{b})$, respectively. GMCU, a special case of $\mathbf{P}$, has the form: minimize $\sum_{k=1}^{n} c_k x_k$ subject to flow conservation constraint ($\sum_{k: \partial_G^-(e_k) = j} c_k x_k - \sum_{k: \partial_G^+(e_k) = j} x_k = b_j$) for each $j \in V$ and nonnegativity constraint ($x_k \geq 0$) for each $k = 1, \ldots, n$, given a digraph (i.e., directed graph) $G := (V, E)$ with a vertex set $V := \{1, \ldots, n\}$ and an edge set $E := \{e_k : k = 1, \ldots, n\}$, a supply/demand $b_j \in \mathbb{R}$ for each vertex $i \in V$, and a pair of cost $c_k \in \mathbb{R}$ and gain $\gamma_k \in \mathbb{R}_+ \setminus \{0\}$ assigned to each edge $e_k \in E$. Here, if $b_i \geq 0$ then $b_i$ is a demand, otherwise it is a supply, and $\partial_G^+(e_k)$ (resp. $\partial_G^-(e_k)$) is the initial (resp. end) vertex of $e_k \in E$.

Since our discussion is motivated by Adler and Cosares [1], we should refer to their theoretical results. They focused on a generalized transshipment problem $\mathbf{GT}$ slightly different from GMCU between these formulations. $\mathbf{GT}$ has one specified vertex on which flow conservation constraint is not imposed. Further, they considered $\mathbf{P}$ with the following coefficient matrix $\mathbf{A}$:

(A1) $\mathbf{A}$ is a pre-Leontief matrix with at most two nonzero components in each column. Observe that $\mathbf{GT}$ is a special case of this problem $\mathbf{P}$ with (A1). As a result, they proved that an optimal solution to $\mathbf{P}$ with (A1) and $\mathbf{b} \in \mathbb{R}^n$, if it exists, can be obtained in strongly polynomial time, by applying Megiddo’s algorithm [13] for TVPI (i.e., Two Variables Per Inequality). They also showed as a corollary that if $\mathbf{b}$ has fixed signs (i.e., $\mathbf{b}$ or $-\mathbf{b}$ belongs to $\mathbb{R}_+^n$) then $\mathbf{GT}$ can also be solved in strongly polynomial time.

Compared to Adler and Cosares [1], the starting point of our discussion is $\mathbf{P}$ with the following coefficient matrix $\mathbf{A}$:

(A2) $\mathbf{A}$ is a pre-Leontief matrix with exactly two nonzero components with opposite signs in each column. Note that $\mathbf{P}$ with (A2) is wider than GMCU, which means that an instance of the latter problem implies that of the former one. Main contribution is to show that if $\mathbf{b}$ has fixed signs then we obtain an optimal solution, if existing, to $\mathbf{P}$ with (A2) in strongly polynomial time. In the process of the analysis, we propose an efficient algorithm of judging the feasibility. Moreover, $\mathbf{GT}$ and its extension problem we devised can be solved by our algorithm as shown later. Here, the extension problem of $\mathbf{GT}$ has several vertices on each of which flow conservation constraint is not imposed.

Although $\mathbf{P}$ with (A1) is wider than $\mathbf{P}$ with (A2), we think that the condition (A2) is more natural than (A1) since each of $\mathbf{GT}$ and its extension problem can be transformed into GMCU. Such a transformation will be shown in the next section. Moreover, we present some theoretical and computational advantages which were not clearly indicated in [1]. These advantages are as follows: 1) we provide a new matrix decomposition theorem

\begin{equation}
\begin{aligned}
\mathbf{c}^\top \mathbf{x} = \mathbf{b},
\mathbf{x} \geq \mathbf{0},
\sum_{k=1}^{n} c_k x_k - \sum_{k: \partial_G^+(e_k) = j} x_k = b_j,
\end{aligned}
\end{equation}
with seven categories, which refines a theorem without categorization by Veinott [24]; 2) if we know a specific category to which the decomposition of $A$ with (A2) belongs, then we can easily construct an optimal solution of the original problem $P(A, b, c)$ through a child problem corresponding to such a category, or see that the original problem is unbounded right away. For example, the following fact will be shown later. Suppose that a matrix $A$ with (A2) has a decomposition belonging to Category II such that

$$A := \begin{pmatrix} A_1 & O & A_{22} \\ O & A_{21} & A_{22} \end{pmatrix},$$

where $A_1$ is a Leontief matrix and $(A_{21} A_{22})$ is a sub-Leontief (both will be defined later). If a child problem $P(A_1, b_1, c_{A_1})$ has an optimal solution $u$ then an original optimal solution is directly given as $(u^\top 0^\top 0^\top)^\top$ where $b_1$ is a subvector of $b$ corresponding to rows of $A_1$, and $c_{A_1}$ denotes a subvector of $c$ corresponding to columns of $A_1$. To explain these advantages, we take the following process including four important items (i)–(iv). After offering preliminaries in Section 2, we focus in Section 3

(i) refinement of the matrix decomposition theorem (Theorem 2 in Veinott [24]), and
(ii) classification of coefficient matrix $A$ into seven categories by imposing the condition (A2) on $A$, with a few remarks on such a categorization.

At the end of Section 3, we clarify the optimality or feasibility relations between the original problem and its child ones. In Section 4, we develop

(iii) new efficient algorithms for judging the triviality of rows and columns in $A$, to determine which category the matrix decomposition belongs to.

Here, the notion of such triviality was originally introduced by Veinott [24]. Section 5 proposes

(iv) a main flow-based efficient algorithm for the original problem.

Finally, concluding remarks are offered in Section 6. Our main contribution is the following two: 1) The developed algorithm can also determine the feasibility of the generalized transshipment problem, and our complexity is much smaller than that of [1]; 2) We showed efficient algorithms for partitioning the given coefficient matrix into submatrices by introducing the trivialities/nontrivialities explained above. Note that our complexity is the same as that of [1] if a coefficient matrix partitioned is given.

2. Preliminaries

2.1. Relations between GT and GMCU

Here we extend problem GT and show that the extended problem can be transformed into an instance of problem GMCU. Let $G := (V, E)$ be a digraph where $V := \{1, 2, \ldots, m, m + 1, \ldots, m + l\}$, $T := \{m + 1, \ldots, m + l\} \subset V$ for a certain integer $l > 0$, and $E := \{e_k \mid k = 1, \ldots, n\}$. We define the extended problem GT($T$) as the form: minimize $c(x_1, \ldots, x_n) := \sum_{k=1}^{n} c_k x_k$ subject to $\sum_{k, \partial G(e_k) = j} \gamma_k x_k - \sum_{k, \partial G(e_k) = j} x_k = b_j \ (j \in V \setminus T)$ and $x_k \geq 0$ ($k = 1, \ldots, n$), where $c_k \in \mathbb{R}$, $\gamma_k \in \mathbb{R}_+ \setminus \{0\}$ and $b_j \in \mathbb{R}$ ($j \in V \setminus T$). Note that problem GT in Section 1 is equivalent to problem GT($T$) with $l = 1$.

For $G := (V, E)$, we construct another digraph $G' := (V', E')$ by the following procedure:

- (S1) $V' := V$, $E' := E$.
- (S2) For each $t = 1, \ldots, l$ ($= |T|$), let $n := |E|$ and $n_t := n + 6(t - 1)$, and do (S2.1) and (S2.2):

Copyright © by ORSI. Unauthorized reproduction of this article is prohibited.
(S2.1) $V' \leftarrow V' \cup \{r_t, s_t\}$, $E' \leftarrow E' \cup \{e_{n+i} \mid i = 1, 2, 3\}$, $G' \leftarrow (V', E')$, where $r_t$ and $s_t$ are new vertices and $e_{n+i}$ ($i = 1, 2, 3$) are new edges satisfying $\partial^-_G(e_{n+i}) = m + t = \partial^+_G(e_{n+i+1})$, $\partial^+_G(e_{n+i+1}) = r_t = \partial^+_G(e_{n+i+2})$, and $\partial^-_G(e_{n+i+2}) = s_t = \partial^-_G(e_{n+i+3})$.

(S2.2) $V' \leftarrow V' \cup \{p_t, q_t\}$, $E' \leftarrow E' \cup \{e_{n+i} \mid i = 4, 5, 6\}$, $G' \leftarrow (V', E')$, where $p_t$ and $q_t$ are new vertices and $e_{n+i}$ ($i = 4, 5, 6$) are new edges satisfying $\partial^-_G(e_{n+i}) = m + l = \partial^+_G(e_{n+i+1})$, $\partial^+_G(e_{n+i+1}) = p_t = \partial^+_G(e_{n+i+2})$, and $\partial^-_G(e_{n+i+2}) = q_t = \partial^-_G(e_{n+i+3})$. □

Roughly speaking, $G'$ is constructed by adding to $G$ two cycles passing through vertex $m + t \in T$ for each $t = 1, \ldots, l$. Further, given parameters $c_k$, $\gamma_k$ ($k = 1, \ldots, n$), $b_j$ ($j \in V \setminus T$) in $G(T)$, we define

$$c'_k := \begin{cases} c_k & (k = 1, \ldots, n) \\ 0 & (k = n + 1, \ldots, n + 6l) \end{cases}, \quad b'_j := \begin{cases} b_j & (j = 1, \ldots, m) \\ 0 & (j \in \bigcup_{t=1}^{l\{m + t, r_t, s_t, p_t, q_t\}}) \end{cases}$$

$$\gamma'_k := \gamma_k \text{ for } k = 1, \ldots, n, \quad \text{and} \quad \gamma'_{n+i} := \begin{cases} 1 & (i = 1, 2, 4, 5) \\ 1/2 & (i = 3) \\ 2 & (i = 6) \end{cases}$$

For these parameters, we consider another problem $G^T(T)$ with the form: minimize $c'(x_1, \ldots, x_{n+6l}) := \sum_{k=1}^{n+6l} c'_k x'_k$ subject to $\sum_{k=1}^{l\{m + t, s_t, r_t, p_t, q_t\}} \gamma'_k x'_k - \sum_{k=1}^{l\{m + t, s_t, r_t, p_t, q_t\}} b'_j = 0$ and $x'_k \geq 0$ ($k = 1, \ldots, n + 6l$). Obviously, this problem is a GMCU. To complete the aim in this subsection, we verify that $G^T(T)$ is equivalent to $G(T)$ by showing the following two propositions.

**Proposition 2.1.** If $(\tilde{x}_1, \ldots, \tilde{x}_n)$ is a feasible solution to $G(T)$, then there exists a feasible solution $(\hat{x}_1, \ldots, \hat{x}_{n+6l})$ to $G^T(T)$ with $c'(\hat{x}_1, \ldots, \hat{x}_{n+6l}) = c(\tilde{x}_1, \ldots, \tilde{x}_n)$. Conversely, if $(\hat{x}_1, \ldots, \hat{x}_{n+6l})$ is a feasible solution to $G^T(T)$, then $(\tilde{x}_1 := \hat{x}_1, \ldots, \tilde{x}_n := \hat{x}_n)$ is a feasible solution to $G(T)$ with $c(\tilde{x}_1, \ldots, \tilde{x}_n) = c'(\hat{x}_1, \ldots, \hat{x}_{n+6l})$.

**Proof.** We prove only the former statement, since the latter one holds obviously. Assume that $(\tilde{x}_1, \ldots, \tilde{x}_n)$ is a feasible solution to $G(T)$. Let $\hat{x}_k := \tilde{x}_k$ for $k = 1, \ldots, n$. For each $t = 1, \ldots, l$, we define $\Delta_t := \sum_{k=1}^{l\{m + t, s_t, r_t, p_t, q_t\}} \gamma'_k x'_k - \sum_{k=1}^{l\{m + t, s_t, r_t, p_t, q_t\}} b'_j = 0$, and if $\Delta_t \geq 0$ then

$$\hat{x}_{n+i} := \begin{cases} 2\Delta_t & (i = 1, 2, 3) \\ 0 & (i = 4, 5, 6) \end{cases}$$

otherwise (i.e., $\Delta_t < 0$)

$$\hat{x}_{n+i} := \begin{cases} 0 & (i = 1, 2, 3) \\ -\Delta_t & (i = 4, 5, 6) \end{cases}$$

Then it is straightforward that $(\hat{x}_1, \ldots, \hat{x}_{n+6l})$ is a feasible solution to $G^T(T)$ satisfying $c'(\hat{x}_1, \ldots, \hat{x}_{n+6l}) = c(\tilde{x}_1, \ldots, \tilde{x}_n)$. □

The following proposition comes directly from Proposition 2.1.

**Proposition 2.2.** If $(\hat{x}^*_1, \ldots, \hat{x}^*_n)$ is an optimal solution to $G(T)$, then there exists an optimal solution $(\tilde{x}^*_1, \ldots, \tilde{x}^*_{n+6l})$ to $G^T(T)$ with $c'(\tilde{x}^*_1, \ldots, \tilde{x}^*_{n+6l}) = c(\tilde{x}^*_1, \ldots, \tilde{x}^*_n)$. Conversely, if $(\hat{x}^*_1, \ldots, \hat{x}^*_{n+6l})$ is an optimal solution to $G^T(T)$, then $(\tilde{x}^*_1 := \hat{x}^*_1, \ldots, \tilde{x}^*_n := \hat{x}^*_n)$ is an optimal solution to $G(T)$ with $c(\tilde{x}^*_1, \ldots, \tilde{x}^*_n) = c'(\hat{x}^*_1, \ldots, \hat{x}^*_{n+6l})$. □
2.2. Leontief matrices

We use \((v)_j\) to denote the \(j\)th component of a vector \(v\), and \(\text{col}(A)\) is the number of columns of a matrix \(A\). Recall that \(A\) is a \textit{pre-Leontief matrix} if each column of \(A\) has at most one positive component. Adler and Cosares [1] defined a \textit{Leontief matrix} as a pre-Leontief matrix \(A\) such that there exists a nonnegative vector \(x\) satisfying \(Ax > 0\). Related to Leontief matrices, the following notions were introduced by Veinott [24]. The \(i\)th row of a matrix \(A \in \mathbb{R}^{m \times n}\) is \textit{trivial} if

\[
\forall x \ ((x \geq 0) \land (Ax \geq 0)) \Rightarrow ((Ax)_i = 0)
\]

where \(i \in \{1, \ldots, m\}\). If the row is not trivial, then it is \textit{nontrivial}. The \(j\)th column of \(A\) is \textit{trivial} if

\[
\forall x \ ((x \geq 0) \land (Ax \geq 0)) \Rightarrow ((x)_j = 0)
\]

where \(j \in \{1, \ldots, n\}\). If the column is not trivial, then it is \textit{nontrivial}. The following proposition links a Leontief matrix and row-triviality.

**Proposition 2.3.** A matrix \(A \in \mathbb{R}^{m \times n}\) is Leontief if and only if \(A\) is pre-Leontief and each row is nontrivial.

**Proof.** Suppose that \(A\) is Leontief. By definition, \(A\) is pre-Leontief and there is an \(x \in \mathbb{R}^n_+\) satisfying \(Ax > 0\). Thus, we have \((Ax)_i > 0\) for each \(i = 1, \ldots, m\), which implies that each \(i\)-th row of \(A\) is nontrivial. Next, we show the converse statement. By definition, there exists an \(x_i \in \mathbb{R}^n_+\) satisfying \((Ax)_i > 0\) for any \(i = 1, \ldots, m\). Letting \(x := \sum_{i=1}^m x_i\), it follows from \((Ax_k)_i \geq 0\) \((k = 1, \ldots, m; i = 1, \ldots, m)\) that \((Ax)_i = \sum_{k=1}^m (Ax_k)_i > 0\) for each \(i = 1, \ldots, m\), implying \(Ax > 0\). By definition, \(A\) is Leontief. Note \(x \geq 0\). \(\square\)

A \textit{sub-Leontief matrix} is pre-Leontief such that each row is trivial. A system \('Ax = b, x \geq 0'\) is a Leontief (resp. pre-Leontief, sub-Leontief) substitution system if \(A\) is Leontief (resp. pre-Leontief, sub-Leontief). It is well-known that a square Leontief matrix \(A\) has the inverse matrix \(A^{-1} \geq O\) (See [24]). Initially, we observe the following simple result.

**Proposition 2.4.** If \(A\) is Leontief, then problem \(P\) with \(b \geq 0\) is feasible.

**Proof.** We prove the proposition by showing that \(D\), the dual problem of \(P\) with \(b \geq 0\), is not unbounded. Suppose \(A\) is Leontief. Then there exists \(\hat{x} \in \mathbb{R}^n_+\) with \(A\hat{x} > 0\). Setting \(\hat{b} := Ax\), we find that \(P'\) is obviously feasible where \(P'\) is \(P\) with \(\hat{b}\) in place of \(b\). If \(P'\) is unbounded, then from the duality theorem we find that \(D'\), the dual problem \(P'\), is infeasible. It follows that \(D\) is also infeasible. Otherwise \((P'\) has an optimal solution), letting \(x^* \in \mathbb{R}^n_+\) be an optimum solution of \(P'\), from the duality theorem we find that \(D'\) has an optimum solution \(y^* \in \mathbb{R}^m\) satisfying \(c^T x^* = \hat{b}^T y^*\). We claim that \(D\) also has an optimum solution. By proving the claim, we complete the proof. In order to derive contradiction, assume that \(D\) is unbounded when \(P'\) has an optimal solution. Then there exists \(\tilde{y} \in \mathbb{R}^m\) satisfying \(\hat{b}^T \tilde{y} > \hat{b}^T y^*\) and \(A^T \tilde{y} \leq c\). Let \(\hat{y} := \alpha \tilde{y} + (1 - \alpha) y^*\) for arbitrary \(\alpha \in (0, 1)\). It is clear that \(\hat{y}\) is a feasible solution of \(D'\). Also, letting \(d := b - b\), we have

\[
\hat{b}^T \hat{y} = \alpha d^T \tilde{y} + \alpha b^T \tilde{y} + (1 - \alpha) b^T y^* > \alpha d^T \tilde{y} + \alpha b^T y^* + (1 - \alpha) b^T y^* = \alpha d^T \tilde{y} + \hat{b}^T y^*.
\]

Then we have \(\hat{b}^T \hat{y} > \hat{b}^T y^*\). In fact, if \(d^T \tilde{y} > 0\) then it is trivial, otherwise \((d^T \tilde{y} < 0)\), since

\[
\hat{b}^T \hat{y} - \hat{b}^T y^* = \alpha d^T \tilde{y} + \epsilon
\]

for some \(\epsilon > 0\), the inequality holds for sufficiently small \(\alpha > 0\) satisfying \(\alpha < -\epsilon/d^T \tilde{y}\). This contradicts the optimality of \(y^*\). \(\square\)
For $A \in \mathbb{R}^{m \times n}$, we define
\[ I_A := \{ i \in \{1, \ldots, m\} : \text{The } i\text{th row is trivial in } A \}, \quad J_A := \{1, \ldots, m\} \setminus I_A. \]
\[ J_A := \{ j \in \{1, \ldots, n\} : \text{The } j\text{th column is trivial in } A \}, \quad J_A := \{1, \ldots, n\} \setminus J_A. \]

$I_A$ (resp. $J_A$) is a set of nontrivial rows (resp. columns) in $A$. Let $A'$ be a submatrix of $A$. An $i$th row of $A'$ is different from the corresponding row of $A$ in general, but we regard them as the same rows. Similarly, we adopt such a rule with respect to columns of $A'$. The following series of lemmas are our observations.

**Lemma 2.1.** Suppose $I_A \neq \emptyset$ for $A = (a_{ij}) \in \mathbb{R}^{m \times n}$. Then each $i \in I_A$ satisfies $a_{ij} > 0$ for some $j \in J_A$.

**Proof.** Assume that there exists $i \in I_A$ such that $a_{ij} \leq 0$ for any $j \in J_A$. From $i \in I_A$, we have $x \in \mathbb{R}_+^n$ satisfying $Ax \geq 0$ and $(Ax)_i > 0$. On the other hand, it follows from $(x)_j = 0$ ($j \in J_A$) and $x \geq 0$
\[ (Ax)_i = \sum_{j \in J_A} a_{ij} \cdot (x)_j + \sum_{j \in J_A} a_{ij} \cdot (x)_j \leq 0 \]
which contradicts $(Ax)_i > 0$.

The following four lemmas are shown without proofs since they are easy to verify.

**Lemma 2.2.** Suppose $J_A \neq \emptyset$ for $A \in \mathbb{R}^{m \times n}$. Then there exists $x \in \mathbb{R}_+^n$ such that $Ax \geq 0$ and $(x)_j > 0$ for each $j \in J_A$.

**Lemma 2.3.** If a row of a matrix consists only of nonpositive components, then it is trivial.

**Lemma 2.4.** If $A$ has no positive components, then $A$ is sub-Leontief.

**Lemma 2.5.** If $A \in \mathbb{R}^{m \times n}$ is Leontief, then the following statements (i)–(iii) hold: (i) $A$ has at least one positive component in each row; (ii) The number of positive components in $A$ is at most $m$; (iii) $m \leq n$.

Veinott pointed out the following lemma in [24]. We prove the lemma for later discussion.

**Lemma 2.6.** ([24], Lemma 1) Let $A = (a_{ij}) \in \mathbb{R}^{m \times n}$ be a pre-Leontief matrix with the $q$th column nontrivial. If $a_{iq} \leq 0$ for any $i \in I_A$, then we have $a_{iq} = 0$ for each $i \in I_A$.

**Proof.** For $q \in J_A$, suppose that $a_{iq} \leq 0$ for each $i \in I_A$, but that there is a negative component $a_{pq}$ for some $p \in I_A$. We consider the following two cases for the $q$th column: (a) It has exactly one positive component $a_{kq}$ for some $k$; (b) It has no positive component. Note that $k \in I_A$ holds. For both cases we will show that there exists $x' \in \mathbb{R}_+^n$ such that $Ax' \geq 0$ and $(Ax')_k > 0$, which contradicts $p \in I_A$.

Case (a): First, observe that there is $x \in \mathbb{R}_+^n$ satisfying
\[ Ax \geq 0, \quad (x)_q > 0, \quad (Ax)_k > 0. \tag{2.1} \]

The reason is as follows. From $k \in I_A$, there exists $y \in \mathbb{R}_+^n$ satisfying $Ay \geq 0$ and $(Ay)_k > 0$. Moreover, we have $z \in \mathbb{R}_+^n$ such that $Az \geq 0$ and $(z)_q > 0$ as the $q$th column is nontrivial. Thus, $x := y + z$ is a desired vector satisfying (2.1). Recalling $a_{kq} > 0$, define $x'$ by $(x')_q := (x)_q - \epsilon$ and $(x')_j := (x)_j$ $(j \in \{1, \ldots, n\} \setminus \{q\})$ where $\epsilon := \min\{(x)_q, (Ax)_k/a_{kq} \} > 0$. Then it is straightforward that $x' \geq 0$. For such a $k \in I_A$ we have

Copyright © by ORSI. Unauthorized reproduction of this article is prohibited.
It follows from \( a_{ij} \leq 0 \) \((i \in \{1, \ldots, m\} \setminus \{k\})\) that \((Ax')_i = (Ax)_i - \epsilon a_{iq} \geq (Ax)_i \) for each \( i \) not equal to \( k \). Thus, we have \( Ax' \geq 0 \). In particular, we see from \( a_{pq} < 0 \) that \((Ax')_p = (Ax)_p - \epsilon a_{pq} > (Ax)_p \geq 0\), which implies \((Ax')_p > 0\). Note that the \( q \)th column has exactly one positive component.

Case (b): By \( q \in J_A \), there is \( x \in R^m_+ \) such that \( Ax \geq 0 \) and \((x)_q > 0\). Define \( x' \) as in case (a) where \( \epsilon := (x)_q > 0 \). Similarly, we can prove \( Ax' \geq 0 \), \( x' \geq 0 \), and \((Ax')_p > 0\). □

The following lemma is useful to prove several other propositions shown below.

**Lemma 2.7.** Let \( A \in R^{m \times n} \) be a pre-Leontief matrix such that \( A = (A_1, A_2) \) for some \( A_1 \) and \( A_2 \) where \( 1 \leq k := \text{col}(A_1) < n \). If \( A_1 \) is Leontief, then any \( x_2 \in R^{n-k}_+ \) satisfies \( A_1 x_1 + A_2 x_2 \geq 0 \) for some \( x_1 \in R^k_+ \).

**Proof.** Choosing any \( x_2 \in R^{n-k}_+ \), put \( b := A_2 x_2 \) and

\[
\mu := \min \left\{ 0, \min_{i \in \{1, \ldots, m\}} (b)_i \right\}.
\]

Since \( A_1 \) is Leontief, there exists \( x' \in R^k_+ \) such that \( A_1 x' > 0 \). If \( \mu = 0 \), then it follows from \((b)_i \geq 0 \) \((i = 1, \ldots, m)\) that \( A_1 x' + A_2 x_2 > 0 \), which implies that \( x_1 := x' \) is a desired vector. Otherwise \((\mu < 0)\), define a positive scalar \( \lambda \) by \( \lambda := -\mu/\gamma \) where \( \gamma := \min_{i \in \{1, \ldots, m\}} (A_1 x')_i > 0 \). Then it is straightforward that \( x_1 := \lambda x' \geq 0 \). Moreover, the following relations hold:

\[
(A_1 x_1 + A_2 x_2)_i = \lambda \cdot (A_1 x')_i + (b)_i = -\frac{\mu}{\gamma} (A_1 x')_i + (b)_i \geq -\mu + \mu = 0 \quad (i = 1, \ldots, m),
\]

implying \( A_1 x_1 + A_2 x_2 \geq 0 \) □

If we put \( x_1 := (\lambda + 1)x' \) for \( \mu < 0 \) in the proof of Lemma 2.7, then \( A_1 x_1 + A_2 x_2 \) becomes positive, which implies the following corollary.

**Corollary 2.1.** Let \( A := (A_1, A_2) \in R^{m \times n} \) be a pre-Leontief matrix given in Lemma 2.7. If \( A_1 \) is Leontief, then so is \( A \). □

A partition theorem concerning a pre-Leontief matrix was proved in Theorem 2, [24] in which Veinott outlined a method for obtaining a desired matrix by partitioning the rows and columns of the matrix. Here, we present an algorithm implementing the method where an input matrix \( A \) is pre-Leontief.

**Algorithm for LP with Pre-Leontief Matrix**

(\textbf{Output}) pre-Leontief matrix \( A \) permuted and updated sets \( I_A, \overline{I}_A, J_A, \overline{J}_A \).

(\textbf{S1}) If \( I_A \neq \emptyset \) and \( \overline{I}_A \neq \emptyset \), then follow (\textbf{S1.1}).

(\textbf{S1.1}) Permute the rows of \( A \) so that any nontrivial row appears above any trivial row.

We exchange elements in \( I_A \) for those in \( \overline{I}_A \) corresponding to such permutations.

(\textbf{S2}) If \( J_A \neq \emptyset \) and \( \overline{J}_A \neq \emptyset \), then follow (\textbf{S2.1}).

(\textbf{S2.1}) Permute the columns of \( A \) so that any nontrivial column appears to the left of any trivial column. We exchange elements in \( J_A \) for those in \( \overline{J}_A \) corresponding to such permutations.

(\textbf{S3}) If \( \overline{I}_A \neq \emptyset \) and \( \overline{J}_A \neq \emptyset \), then we follow (\textbf{S3.1}) for all the nontrivial columns.

(\textbf{S3.1}) Permute those columns of \( A \) so that each nontrivial column having a positive
component in some nontrivial row appears to the left for any nontrivial column satisfying that each component in nontrivial row is nonpositive.

Now, we present the partition theorem with a condition added concerning the sets with respect to trivialities/nontrivialities of the rows and columns, since the theorem without the condition needs more refinement. For example, there is no Leontief submatrix in such a partition when any row of a pre-Leontief matrix is trivial. We provide our proof of the theorem for readers’ convenience.

**Theorem 2.1.** ([24], Theorem 2) Let $A \in \mathbb{R}^{m \times n}$ be pre-Leontief, and all of $I_A$, $T_A$, $J_A$, and $\overline{T}_A$ non-empty. After PermuteRowsColumns above, $A$ can be partitioned into $A := \begin{pmatrix} A_1 & A_3 \\ A_4 & A_2 \end{pmatrix}$ so as to satisfy the following conditions (i)–(iii): (i) $A_1$ is Leontief; (ii) $A_4 = O$; (iii) $A_2$ is sub-Leontief.

**Proof.** Let $p$ be the number of nontrivial columns in $A$ such that each of them has exactly one positive component in some row nontrivial in $A$. It is straightforward that $0 < p \leq |\overline{T}_A|$. Suppose that $A$ is partitioned into $\begin{pmatrix} A_1 & A_3 \\ A_4 & A_2 \end{pmatrix}$ after carrying out PermuteRowsColumns where $A_1 \in \mathbb{R}^{|\overline{T}_A| \times p}$, $A_3 \in \mathbb{R}^{|\overline{T}_A| \times (n-p)}$, $A_4 \in \mathbb{R}^{|J_A| \times p}$, and $A_2 \in \mathbb{R}^{|J_A| \times (n-p)}$. Note that each column of $A_1$ has a positive component. Since any row of $(A_1 A_3)$ is nontrivial in $A$, there is $x := \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbb{R}^n_+$ satisfying $Ax \geq 0$ and $(A_1 A_3)x = A_1 x_1 + A_3 x_2 > 0$. Since any positive component in $A_4$ appears in some column trivial in $A$, it follows from $A_1 x_1 + A_3 x_2 > 0$ and $A_3 x_2 \leq 0$ that $A_1 x_1 > 0$, which implies from $x_1 \geq 0$ that $A_1$ is Leontief, i.e., (i) is done. Any component in $A_4$ is nonpositive. In fact, $A$ is pre-Leontief and the note above holds. Hence, Lemma 2.6 shows $A_4 = O$, which implies (ii). Assume that $A_2$ is not sub-Leontief. Then there exists a $q$th row of $A_2$ satisfying $A_2 x'_0 \geq 0$ and $(A_2 x'_0)_q > 0$ for some $x'_0 \in \mathbb{R}^{n-p}_+$. Since $A_1$ is Leontief, it follows from Lemma 2.7 that there is $x'_1 \in \mathbb{R}^{n-p}_+$ such that $A_1 x'_1 + A_3 x'_2 \geq 0$. From $A_4 = O$ and $A_2 x'_2 \geq 0$, we have $A_4 x'_1 + A_2 x'_2 = A_2 x'_2 \geq 0$, which implies $x'_2 := \begin{pmatrix} x'_1 \\ x'_2 \end{pmatrix} \geq 0$ and $Ax'_2 \geq 0$. Furthermore, we know $(Ax'_2)|_{T_A + q} = (A_2 x'_2)_q > 0$, which contradicts that the $(|\overline{T}_A| + q)$th row of $A$ is trivial in $A$. So, (iii) holds.

### 2.3. Graphs and generalized minimum cost flow problem

A digraph is *simple* if each edge is defined uniquely as an ordered pair of two distinct vertices.

Throughout, we deal only with simple digraphs, so we simply refer to them as ‘graphs’. For a graph $G$, the vertex (resp. edge) set of $G$ is also denoted by $V(G)$ (resp. $E(G)$). Recall that, for an edge $e \in E(G)$, $\partial^+_G(e)$ and $\partial^-_G(e)$ are the end and initial vertices of $e$, respectively, where $e = (\partial^+_G(e), \partial^-_G(e))$. We define $\partial_G(v) := \partial^+_G(v) \cup \partial^-_G(v)$ for $\partial^+_G(v) := \{ e \in E(G) : \partial^+_G(e) = v \}$ where double-signs correspond. For $E' \subseteq E(G)$, we write $\overrightarrow{E}' := \{ \overrightarrow{e} : e \in E' \}$ and $\overleftarrow{E}' := E' \cup \overrightarrow{E}'$ where $\overrightarrow{e}$ is the reverse edge of $e$ defined by $\overrightarrow{e} = (\partial^-_G(e), \partial^+_G(e))$. For a graph $G$ with $E' \subseteq E(G)$ (resp. $V'' \subseteq V'(G)$), $G'[E'] := (V', E')$ (resp. $G'[V''] := (V'', E'')$) is a subgraph of $G$ induced by $E'$ (resp. $V''$) where $V' := \bigcup_{e \in E'} \{ \partial^+_G(e), \partial^-_G(e) \}$ (resp. $V'' := \{ e \in E(G) : \partial^+_G(e), \partial^-_G(e) \} \subseteq V''$). Given a graph $G$, a *walk* is a sequence of vertices and edges $(v_1, e_1, v_2, \ldots, v_k, e_k, v_{k+1})$ such that $e_i = (v_i, v_{i+1})$ ($i \in \{1, \ldots, k\}$). For such a walk, $v_1$ (resp. $v_{k+1}$) is the initial (resp. end) vertex. A *trail* is a walk whose edges are distinct from each other. We also consider a subgraph $G' := \{ (v_1, \ldots, v_{k+1}), \{ e_1, \ldots, e_k \} \}$ in $G$. $G'$ is a *path* if its walk is a trail and the vertices are all different. On the other hand, $G'$ is

© by ORSJ. Unauthorized reproduction of this article is prohibited.
a cycle if $G'$ is a trail, the initial and the end vertices of the trail coincide and other vertices are distinct from each other. If $G'$ is a path whose trail is $W := (v_1, e_1, v_2, \ldots, v_k, e_k, v_{k+1})$, then we state that ‘$G'$ is a path from $v_1$ to $v_{k+1}$’, ‘$G'$ is a $v_1$-$v_{k+1}$-path’, ‘$v_{k+1}$ is reachable from $v_1$ in $G'$’. For a path $P$ and two vertices $a, b$ in $P$, $P_{[a,b]}$ is a subpath of $P$ from $a$ to $b$. Given a cycle $C$ and a length function $l : E(G) \to \mathbb{R}$ in a graph $G$, the length of $C$ is $l(C) := \sum_{e \in E(C)} l(e)$. A cycle $C$ is negative with respect to $l$ if $l(C) < 0$. For a graph $G$ and an edge $e \in E(G)$ such that $\tilde{e} \notin E(G)$, reversing $e$ means constructing a graph $(V(G), E(G) \cup \{\tilde{e}\} \setminus \{e\})$. Consider a graph $P$ with $\delta_P(x) = \delta_P(y) = 1$ and $\delta_P(v) = 2$ $(v \in V(P) \setminus \{x, y\})$ for some distinct vertices $x, y \in V(P)$. If we can obtain an $x$-$y$-path by reversing some edges in $E(P)$, then $P$ is called an undirected path between $x$ and $y$. An undirected cycle is similarly defined. A graph $G$ is connected if there exists an undirected path between $x$ and $y$ for any pair of vertices $x, y \in V(G)$. The maximal connected subgraphs of a graph $G$ are connected components of $G$.

We use a theory of generalized flows to solve the subproblems in the following section. First, we formulate the generalized minimum cost flow problem in a network. Suppose that $G^0$ is a simple connected graph satisfying $\tilde{e} \notin E(G^0)$ for each $e \in E(G^0)$. For $G^0$, a capacity function $u^0$, and a gain function $\gamma^0$ on $E(G^0)$, we consider a network $\mathcal{N}^0 := (G^0, u^0, \gamma^0)$ where both $u^0$ and $\gamma^0$ are positive rational and $u^0$ may be infinite. Each $\gamma^0(e) (e \in E(G^0))$ is represented as $\gamma^0(e) = p(e)/q(e)$ for two positive integers $p(e)$ and $q(e)$. Given $\mathcal{N}^0$, a rational cost function $c^0$ on $E(G^0)$, and a rational demand/supply function $\beta^0$ on $V(G^0)$, the generalized minimum cost flow problem $\text{GMC}^0$ in $\mathcal{N}^0$ is formulated as follows:

\[
\begin{align*}
\text{GMC}^0 & \quad \text{Minimize } c^0(f^0) := \sum_{e \in E(G^0)} c^0(e) f^0(e) \text{ s.t.} \\
\sum_{e \in \delta^-_{G^0}(v)} \gamma^0(e) f^0(e) - \sum_{e \in \delta^+_{G^0}(v)} f^0(e) = \beta^0(v) \quad (\forall v \in V(G^0)), \\
0 & \leq f^0(e) \leq u^0(e) \quad (\forall e \in E(G^0)).
\end{align*}
\]  

A function $f^0$ satisfying all the capacity constraints (2.4) is a pseudoflow in $\mathcal{N}^0$. If a pseudoflow in $\mathcal{N}^0$ satisfies (2.3), then it is a flow in $\mathcal{N}^0$. A flow is a circulation if $\beta^0 = 0$. The cost of a pseudoflow $f^0$ is $c^0(f^0)$. From here, we omit ‘generalized’ unless specifically required.

Given $\mathcal{N}^0$, $c^0$, and $\beta^0$, we define a new graph $G$ by $V(G) = V(G^0)$ and $E(G) := E(G^0) \cup \tilde{E}(G^0)$. We also define $c : E(G) \to \mathbb{R}$, $u : E(G) \to \mathbb{R}_+ \cup \{\infty\}$, and $\gamma : E(G) \to \mathbb{R}_+ \{0\}$ by

\[
c(e) := \begin{cases} 
c^0(e) & (e \in E(G^0)) \\
-\frac{c^0(e)}{\gamma^0(e)} & (e \in \tilde{E}(G^0)) \end{cases}, \quad u(e) := \begin{cases} 
u^0(e) & (e \in E(G^0)) \\
0 & (e \in \tilde{E}(G^0)) \end{cases}, \quad \gamma(e) := \begin{cases} 
\gamma^0(e) & (e \in E(G^0)) \\
\frac{1}{\gamma^0(e)} & (e \in \tilde{E}(G^0)) \end{cases}.
\]

Then $\mathcal{N} := (G, u, \gamma)$ is an extended network with respect to $\mathcal{N}^0$. The problem $\text{GMC}$, which is equivalent to $\text{GMC}^0$, is formulated as follows:

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
Definition 2.1. A cycle in $G$ of $N$ is a pseudoflow $f$ for a pseudoflow $g$. Proposition 2.5. A function $g$ satisfying (2.7) and (2.8) is a pseudoflow in $N$. If there is a pseudoflow $g$ in $N$, then $g$ is a flow in $N$. In particular, a pseudoflow $g$ with $\delta^N_g(v) = 0$ for some distinct vertices $x, P, y$.

(i) If $f^0$ is a pseudoflow in $N^0$, then the function $g$ defined by (2.9) is a pseudoflow in $N$.

(ii) If $g$ is a pseudoflow in $N$, then a restriction $g|E(G^0)$ is a pseudoflow in $N^0$ and we have

\[ g(e) \geq 0 \quad (e \in E(G^0)) \quad \text{and} \quad g(e) \leq 0 \quad (e \in \overline{E(G^0)}). \]

(iii) We have $c(g) = 2c^0(f^0)$ for a pseudoflow $g$ in $N$ where $f^0 := g|E(G^0)$.

(iv) If $g$ is a pseudoflow in $N$, then we have

\[ c^N_g(v) = \sum_{e \in \delta^N_g(v)} \gamma(e) f^0(e) - \sum_{e \in \delta^N_g(v)} f^0(e) \quad (v \in V(G)), \]

where $H := G_+(g)$ and $f^0 := g|E(G^0)$. A cycle in $G$ is flow-generating (resp. flow-absorbing, unit-gain) if $\gamma(C) > 1$ (resp. $\gamma(C) < 1$, $\gamma(C) = 1$) where $\gamma(C) := \Pi_{e \in E(C)} \gamma(e)$. To decompose a pseudoflow in $N$, we introduce the following six subgraph types.

Definition 2.1. Fundamental subgraphs of six types in $G$ are defined as follows.

(i) If there is an $x-y$-path $P$ in $G$ with $|E(P)| \geq 1$ for some distinct vertices $x, y$, then $T_1(x, P, y) := P$ is a subgraph of Type 1. Here, $x$ and $y$ are the initial and terminal vertices of $T_1(x, P, y)$, respectively.

(ii) If there is a pair of a flow-generating cycle $C$ and an $x-y$-path $P$ with $|E(P)| \geq 0$ in $G$ satisfying $V(C) \cap V(P) = \{x\}$ for some vertices $x, y$, then $T_2(C, x, P, y) := (V(C) \cup V(P), E(C) \cup E(P))$ is a subgraph of Type 2. Here, $y$ is the terminal vertex of $T_2(C, x, P, y)$.

(iii) If there is a pair of an $x-y$-path $P$ with $|E(P)| \geq 0$ and a flow-absorbing cycle $C$ in $G$ satisfying $V(C) \cap V(P) = \{y\}$ for some vertices $x, y$, then $T_3(x, P, y, C) := (V(P) \cup V(C), E(P) \cup E(C))$ is a subgraph of Type 3. Here, $x$ is the initial vertex of $T_3(x, P, y, C)$. 

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
(iv) If there is a cycle $C$ with unit gain in $G$, then $\mathcal{T}_4(C) := C$ is a subgraph of Type 4.

(v) If there is a triple of a flow-generating cycle $C_1$, a flow-absorbing cycle $C_2$, and an $x$-$y$-path $P$ with $|E(P)| \geq 1$ in $G$ satisfying that $V(C_1) \cap V(C_2) = \emptyset$, $V(C_1) \cap V(P) = \{x\}$, and $V(P) \cap V(C_2) = \{y\}$ for some distinct vertices $x, y$, then $\mathcal{T}_5(C_1, x, P, y, C_2) := (V(C_1) \cup V(P) \cup V(C_2), E(C_1) \cup E(P) \cup E(C_2))$ is a subgraph of Type 5.

(vi) If there is a pair of a flow-generating $C_1$ and a flow-absorbing cycle $C_2$ satisfying $V(C_1) \cap V(C_2) \neq \emptyset$, then $\mathcal{T}_6(C_1, C_2) := (V(C_1) \cup V(C_2), E(C_1) \cup E(C_2))$ is a subgraph of Type 6.

A pseudoflow corresponding to each fundamental subgraph above is specified as follows.

**Definition 2.2.** For these six subgraph types, the 6 types of corresponding pseudoflows are as follows.

(i) If $g$ satisfies $G_+(g) = T_1(x, P, y)$, $e^N_g(x) < 0$, $e^N_g(y) > 0$, and $e^N_g(v) = 0 \ (\forall v \in V(G) \setminus \{x, y\})$ for some $T_1(x, P, y)$, then $g$ is a pseudoflow of Type 1. Here, $x$ and $y$ are the initial and terminal vertices of $g$, respectively.

(ii) If $g$ satisfies $G_+(g) = T_2(C, x, P, y)$, $e^N_g(y) > 0$, and $e^N_g(v) = 0 \ (\forall v \in V(G) \setminus \{y\})$ for some $T_2(C, x, P, y)$, then $g$ is a pseudoflow of Type 2. Here, $y$ is the terminal vertex of $g$.

(iii) If $g$ satisfies $G_+(g) = T_3(x, P, y, C)$, $e^N_g(x) < 0$, and $e^N_g(v) = 0 \ (\forall v \in V(G) \setminus \{x\})$ for some $T_3(x, P, y, C)$, then $g$ is a pseudoflow of Type 3. Here, $x$ is the initial vertex of $g$.

(iv) If $g$ satisfies $G_+(g) = T_4(C)$ and $e^N_g(v) = 0 \ (\forall v \in V(G))$ for some $T_4(C)$, then $g$ is a pseudoflow of Type 4.

(v) If $g$ satisfies $G_+(g) = T_5(C_1, x, P, y, C_2)$ and $e^N_g(v) = 0 \ (\forall v \in V(G))$ for some $T_5(C_1, x, P, y, C_2)$, then $g$ is a pseudoflow of Type 5.

(vi) If $g$ satisfies $G_+(g) = T_6(C_1, C_2)$ and $e^N_g(v) = 0 \ (\forall v \in V(G))$ for some $T_6(C_1, C_2)$, then $g$ is a pseudoflow of Type 6.

Any $g$ in Definition 2.2 is called a fundamental pseudoflow in $\mathcal{N}$. Wayne [25] used a circuit in place of any fundamental pseudoflow of Type $t \in \{4, 5, 6\}$ and a bicycle in place of any fundamental subgraph of Type $t \in \{5, 6\}$. The following theorem is known as a flow decomposition theorem.

**Theorem 2.2.** ([25], Lemma 1) Let $g$ be a pseudoflow in $\mathcal{N} = (G, u, \gamma)$ satisfying $g \neq 0$. Then $g$ can be decomposed into fundamental pseudoflows $g_1, \cdots, g_p$ of Types 1 to 6 in $\mathcal{N}$ so that $g = \sum_{i=1}^p g_i$, for some positive number $p$. In particular, $g|E(G^0)$ can be decomposed into pseudoflows $g_i|E(G^0)$ ($i = 1, \cdots, p$) in $\mathcal{N}^0 = (G^0, u^0, \gamma^0)$ so as to satisfy $g|E(G^0) = \sum_{i=1}^p (g_i|E(G^0))$. □

A pseudoflow is negative if its cost is negative. Similarly, a fundamental subgraph is negative if the cost of the corresponding pseudoflow is negative. Note that $\mathcal{N}$ has a negative circuit (i.e., negative fundamental pseudoflow of Type 4, 5 or 6) if and only if $\mathcal{N}^0$ also does. Let $g$ be a circulation in $\mathcal{N}$. The residual capacity function $u_g$ is defined as $u_g(e) := u(e) - g(e)$ for each $e \in E(G)$. The residual graph $G_+(u_g)$ is a graph consisting of positive residual capacity edges in $G$. The following theorem characterizes an optimal circulation in $\mathcal{N}$, which minimizes the cost.

**Theorem 2.3.** ([25], Theorem 2) A circulation $g$ in $\mathcal{N}$ has the minimum cost if and only if there is no negative circuit in $G_+(u_g)$. □

From previous sections, we may solve two subproblems $P(A_1, b_1, c_{A_1})$ and $P(A_2, 0, c_{A_2})$ to resolve the parent problem $P(A, b, c)$. In the following, we take a different approach from [1]. To realize this aim, we apply a kind of variable-scaling technique. Let $P(A', b', c')$ be a subproblem of the parent problem $P(A, b, c)$, where $A' := (a'_{ij}) \in \mathbb{R}^{k \times l}$ is a submatrix.
of $A$ with $k \leq m$ and $l \leq n$, $b' \in \mathbb{R}^k$ (resp. $c' \in \mathbb{R}^l$) is a subvector of $b$ (resp. $c$) corresponding to rows (resp. columns) of $A'$. Suppose that each column of $A'$ has exactly two nonzero components with opposite signs. Let $i_1, i_2, \ldots, i_l$ be row indices of $A'$ such that $a'_{i_1} < 0$, $a'_{i_2} < 0, \ldots, a'_{i_l} < 0$. We define $\bar{A} := (\bar{a}_{i,j}) \in \mathbb{R}^{k \times l}$ and $\bar{c} \in \mathbb{R}^l$ as $\bar{a}_{h,j} := a'_{i,j}/|a'_{i,j}| \ (h = 1, \ldots, k; \ j = 1, \ldots, l)$ and $(\bar{c})_{j} := (c')_{j}/|a'_{i,j}| \ (j = 1, \ldots, l)$, respectively. Then $P(\bar{A}, b', \bar{c})$ is referred to as a \textit{variable-scaling problem} of $P(A', b', c')$. The following lemma, which is easy to verify, implies that problem $P(A', b', c')$ is equivalent to its variable-scaling problem $P(\bar{A}, b', \bar{c})$.

\textbf{Lemma 2.8.} For $\mathbf{v}, \mathbf{w} \in \mathbb{R}^k$ satisfying $(\mathbf{w})_j = |a'_{i,j}|(\mathbf{v})_j \ (j = 1, \ldots, l)$, we find that $A'\mathbf{v} = \bar{A}\mathbf{w}$ and $(c')^\top \mathbf{v} = \bar{c}^\top \mathbf{w}$. \hfill $\square$

Note that each column of $\bar{A}$ has two nonzero components, one equals to $-1$ and the other is positive. Let $G^0$ be a graph with incident matrix $\bar{A}$ where $v_i \in V(G^0)$ (resp. $e_j \in E(G^0)$) is the vertex (resp. edge) corresponding to the $i$th row (resp. $j$th column) of $\bar{A}$. Further, define a gain of $e_j$, denoted by $\gamma^0(e_j)$, as the positive component of the $j$th column in $\bar{A}$. We call $\mathcal{N}^0 := (G^0, \bar{u}^0 := \infty, \gamma^0)$ a network \textit{corresponding to} $\bar{A}$.

\section{Refinement of Leontief Matrix Decomposition}

Suppose that $A \in \mathbb{R}^{m \times n}$ is a pre-Leontief matrix throughout this section.

\subsection{Leontief matrix after PermuteRowsColumns}

The following theorem and accompanying three corollaries contain new results compared to Theorem 2.1. A crucial point is that $A_3$ may have a column nontrivial in $A$.

\textbf{Theorem 3.1.} Let $A$ be a partitioned matrix obtained after \textbf{PermuteRowsColumns} in Theorem 2.1. Then we have the following statements (i)–(iv).

(i) $A_1$ is Leontief, each column of $A_1$ is nontrivial in $A$, and each row of $A_1$ is nontrivial in $A$.

(ii) $A_2$ is sub-Leontief, each row of $A_2$ is trivial in $A$, and if $A_2$ has a nonzero column nontrivial in $A$ then the column has a positive component in $A_2$.

(iii) $(A_1, A_3)$ is Leontief.

(iv) If $A_3$ has at least one column nontrivial in $A$, then $A$ can be partitioned as $A := \begin{pmatrix} A_1 & A_{31} & A_{32} \\ O & A_{21} & A_{22} \end{pmatrix}$ such that $A_{31}$ consists of columns nontrivial in $A$ and $A_{31} \leq O$.

\textit{Proof.} The first parts of both (i) and (ii) follow from Theorem 2.1. Then, the remaining parts follow from \textbf{PermuteRowsColumns}. Note that if a column of $A_2$ is a nonzero vector nontrivial in $A$ then it follows from the contraposition of Lemma 2.6 that the column of $A_2$ has a positive component in some row trivial in $A$. Since $(A_1, A_3)$ is clearly pre-Leontief, it follows from (i) and Corollary 2.1 that we have (iii). Statement (iv) follows from \textbf{PermuteRowsColumns}. \hfill $\square$

In the above discussion, there is some ambiguity. For example, consider the case when a pre-Leontief matrix $A$ can be partitioned by \textbf{PermuteRowsColumns} into four submatrices as $\begin{pmatrix} A_1 & A_3 \\ O & A_2 \end{pmatrix}$, but cannot into six ones as $\begin{pmatrix} A_1 & A_{31} & A_{32} \\ O & A_{21} & A_{22} \end{pmatrix}$. Then $A_3$ has two possibilities: (i) $A_3$ consists of nonpositive columns nontrivial in $A$ (implying $J_A = \emptyset$), or (ii) $A_3$ consists of columns trivial in $A$. To remove this ambiguity, we introduce a rule of indexing submatrices of $A$. For $A$ with $J_A \neq \emptyset$ after applying \textbf{PermuteRowsColumns}, each submatrix in $A$ is indexed as follows:

\begin{itemize}
  \item $A_1$ has a positive component in some row nontrivial in $A$
  \item $A_2$ has a positive component in some row trivial in $A$
  \item $A_{31}$ has a positive component in some column nontrivial in $A$
  \item $A_{32}$ has a positive component in some column trivial in $A$
  \item $A_{21}$ has a positive component in some column nontrivial in $A$
  \item $A_{22}$ has a positive component in some column trivial in $A$
\end{itemize}
1. $A_1$ denotes, if existing, that each column of $A_1$ is nontrivial in $A$ and has a positive component, and that each row of $A_1$ is nontrivial in $A$.
2. $A_{31}$ denotes, if existing, that each column of $A_{31}$ is nontrivial in $A$ and has no positive component, and that each row of $A_{31}$ is nontrivial in $A$.
3. $A_{32}$ denotes, if existing, that each column (resp. row) of $A_{32}$ is trivial (resp. nontrivial) in $A$.
4. $A_{21}$ denotes, if existing, that each column (resp. row) of $A_{21}$ is nontrivial (resp. trivial) in $A$.
5. $A_{22}$ denotes, if existing, that each column and each row of $A_{22}$ is trivial in $A$.

According to this notation, the partitioned matrix $A$ in Theorem 3.1 which does not satisfy the condition of (iv) is defined by $A := \begin{pmatrix} A_1 & A_{32} \\ O & A_{22} \end{pmatrix}$ in place of $\begin{pmatrix} A_1 & A_3 \\ O & A_2 \end{pmatrix}$.

Each of the following three corollaries deals with a pre-Leontief matrix $A = (a_{ij}) \in \mathbb{R}^{m \times n}$ with $J_A \neq \emptyset$ in which at least one of $I_A$, $I_A$ and $J_A$ is empty.

**Corollary 3.1.** Suppose $I_A \neq \emptyset$, $I_A \neq \emptyset$, and $J_A = \emptyset$, and let $A$ be partitioned by **PermuteRowsColumns**. If there exists $j \in J_A$ satisfying $a_{ij} \leq 0$ for any $i \in I_A$, then $A$ is partitioned into $A := \begin{pmatrix} A_1 & A_{31} \\ A_4 = O & A_{21} \end{pmatrix}$ with $A_{31} \leq O$. Otherwise (each $j \in J_A$ satisfies $a_{ij} > 0$ for a certain $i \in I_A$), $A$ is partitioned into $A := \begin{pmatrix} A_1 \\ A_4 = O \end{pmatrix}$. In both cases, $A_1$ is Leontief.

**Proof.** This comes directly from (i) and (iv) of Theorem 3.1. \hfill \square

**Corollary 3.2.** Suppose $I_A = \emptyset$, and let $A$ be a matrix obtained after **PermuteRowsColumns**. Then the following two claims are true.

(i) All columns of $A$ are nontrivial (i.e., $J_A = \emptyset$).
(ii) If there exists $j \in J_A$ satisfying $a_{ij} \leq 0$ for any $i \in \{1, \ldots, m\}$, then $A$ can be partitioned into $A := \begin{pmatrix} A_1 & A_{31} \end{pmatrix}$ with $A_{31} \leq O$. Otherwise, $A := A_1$ is not partitioned. In both cases, $A_1$ is Leontief.

**Proof.** To prove claim (i), assume that $A$ has at least one trivial column (i.e., $J_A \neq \emptyset$). Then $A$ can be partitioned as $A := \begin{pmatrix} A_1 & A_3 \end{pmatrix}$, where $A_1 \in \mathbb{R}^{m \times p}$ is Leontief, $A_3 \in \mathbb{R}^{m \times (n-p)}$ has at least one column trivial in $A$, and $p$ is the number defined in the proof of Theorem 2.1. Suppose that the $q$th column $\{q \in \{1, \ldots, n-p\}\}$ of $A_3$ is trivial in $A$ (i.e., $p+q \in J_A$). Choosing a vector $x_3 \in \mathbb{R}^{n-p}$ with $(x_3)_q > 0$ arbitrarily, we find from Lemma 2.7 that $A_1x_1 + A_3x_3 \geq 0$ holds for some $x_1 \in \mathbb{R}^n$. This contradicts $p + q \in J_A$. Hence, $J_A = \emptyset$ holds. Then claim (ii) is obviously true due to the permutation mechanism. \hfill \square

**Corollary 3.3.** Suppose $I_A = \emptyset$, and let $A$ be partitioned by **PermuteRowsColumns** modified as follows: (S3) with (S3.1) is replaced by

(S3') Permute the nontrivial columns of $A$ so that any nonpositive nontrivial column appears to the left of any nontrivial column with a positive component.

Then $A$ is a sub-Leontief matrix satisfying the following conditions:

(A) (Case of $J_A \neq \emptyset$) The following three cases can be considered:

(i) If all nontrivial columns of $A$ are nonpositive, then $A$ is partitioned as $A := \begin{pmatrix} O & A_{21} \end{pmatrix}$ with each column of $A_{22}$ trivial in $A$. 

---

*Copyright © by ORSI. Unauthorized reproduction of this article is prohibited.*
(ii) If some (not all) nontrivial columns of $A$ are nonpositive, then $A := (O \ A_{21} \ A_{22})$.

(iii) If each nontrivial column of $A$ has a positive component, then $A := (A_{21} \ A_{22})$.

Note that each column of $A_{21}$ (resp. $A_{22}$) is nontrivial (resp. trivial) in $A$ in both cases (ii) and (iii).

(B) (Case of $J_A = \emptyset$) Note that all columns of $A$ are nontrivial.

(iv) If all columns of $A$ are nonpositive, then $A = O$ (not partitioned).

(v) If some (not all) columns of $A$ are nonpositive, then $A := (O \ A_{21})$, where each column of $A_{21}$ has a positive component.

(vi) If each column of $A$ has a positive component, then $A := A_{21}$ is not partitioned.

Proof. It is obvious from $\bar{I}_A = \emptyset$ that $A$ is sub-Leontief. In each case, it is straightforward to discern from the permutation mechanism that $A$ is partitioned as stated above. In cases (i), (ii), (iv) and (v), the appearance of a zero matrix comes from Lemma 2.6.

3.2. Refined decomposition and categorization

Later, we will consider a network flow problem related to a specific pre-Leontief matrix satisfying the following assumption.

Assumption 3.1. Each column of $A$ has exactly two nonzero components with opposite signs. That is, $A$ satisfies (A2) in Section 1.

Theorem 3.2. Let $A \in \mathbb{R}^{m \times n}$ be a pre-Leontief matrix with Assumption 3.1, and all of $I_A$, $\bar{I}_A$, $J_A$, and $\bar{J}_A$ non-empty. Suppose that $A$ is partitioned as (iv) in Theorem 3.1. Then $A$ is refined as

$$A := \begin{pmatrix} A_1 & A_{31} = O & A_{32} \\ O & A_{21} & A_{22} \end{pmatrix}$$

so as to satisfy the following conditions (i)–(iii):

(i) $A_1$ satisfies (i) of Theorem 2.1.

(ii) Any column of $A_{31} = O$ is nontrivial in $A$.

(iii) Any row of $A_{21}$ is trivial in $A$, and $(A_{21}, A_{22})$ is sub-Leontief.

Proof. Let $A_i$ ($i = 1, 2, 3, 4$) and $p$ be submatrices and the number defined in the proof of Theorem 2.1 where $A_1 \in \mathbb{R}^{|I_A| \times p}$, $A_4 = O \in \mathbb{R}^{|J_A| \times p}$, $A_3 \in \mathbb{R}^{|\bar{J}_A| \times (n-p)}$, and $A_2 \in \mathbb{R}^{|I_A| \times (n-p)}$.

Moreover, we divide $\begin{pmatrix} A_3 \\ A_2 \end{pmatrix}$ into two submatrices such that $\begin{pmatrix} A_{31} \\ A_{21} \end{pmatrix} \in \mathbb{R}^{m \times (|I_A|-p)}$ and $\begin{pmatrix} A_{32} \\ A_{22} \end{pmatrix} \in \mathbb{R}^{m \times |J_A|}$. We find from (iv) in Theorem 3.1 that any column of $A_{31}$ is nontrivial in $A$, which implies (ii) where we prove $A_{31} = O$ later. From (ii) in Theorem 3.1, (iii) holds. The remaining task is to prove $A_{31} = O$. We know $A_{31} \leq O$ by (iv) in Theorem 3.1. Assuming that there exists a pair $(k, l) \in \{1, \cdots, |I_A|\} \times \{1, \cdots, \text{col}(A_{31})\}$ such that $\beta_{kl} < 0$ where $A_{31} = (\beta_{ij})$, we will derive a contradiction. From Assumption 3.1, we have $\gamma_{k'l'} > 0$ for some $k' \in \{1, \cdots, |I_A|\}$ where $A_{21} = (\gamma_{ij})$. Here, observe the following properties for the $(p + l)$th column of $A$.

$$\beta_{il} = 0 \ (\forall i \in \{1, \cdots, |\bar{I}_A|\}\{k\}), \ \gamma_{il} = 0 \ (\forall i \in \{1, \cdots, |I_A|\}\{k'\}).$$

Note that the $(p + l)$th column of $A$ corresponds to the $l$th column of $\begin{pmatrix} A_{31} \\ A_{21} \end{pmatrix}$. Letting $U := (A_1 \ A_{31} \ A_{32})$ and $L := (O \ A_{21} \ A_{22})$, there is $x \in \mathbb{R}^n_+$ such that $Ux > 0$ and $Lx = 0$. 

Copyright © by ORSI. Unauthorized reproduction of this article is prohibited.
For \( \epsilon \) satisfying \( 0 < \epsilon \leq \frac{(U x)_k}{\beta_{kl}} \), define \( x' \) by \((x')_{p+l} := (x)_{p+l} + \epsilon \) and \((x')_j := (x)_j \) for each \( j \) not equal to \( p + l \). Then, we easily have \( x' \geq 0 \). Moreover, the following relations hold:

\[
(U x')_k = (U x)_k + \epsilon \cdot \beta_{kl} \geq -\epsilon \cdot \beta_{kl} + \epsilon \cdot \beta_{kl} = 0, \\
(U x')_i = (U x)_i + \epsilon \cdot 0 > 0 \quad (i \in \{1, \ldots, |\overline{J}_A|\}\{k\}), \\
(L x')_{k'} = (L x)_{k'} + \epsilon \cdot \gamma_{kl} > 0, \\
(L x')_i = (L x)_i + \epsilon \cdot 0 = 0 \quad (i \in \{1, \ldots, |J_A|\}\{k'\}),
\]

which imply \( A x' \geq 0 \). We know \((L x')_{k'} > 0\) contradicting \(|\overline{J}_A| + k' \in I_A\).}

For a pre-Leontief matrix \( A = (a_{ij}) \in \mathbb{R}^{m \times n} \) satisfying Assumption 3.1 and \( \overline{J}_A \neq \emptyset \), we obtain the following three corollaries directly from Corollaries 3.1–3.3.

**Corollary 3.4.** Suppose \( I_A \neq \emptyset, \overline{I}_A \neq \emptyset \) and \( J_A = \emptyset \). If there exists \( j \in \overline{J}_A \) satisfying \( a_{ij} \leq 0 \) for any \( i \in \overline{I}_A \), then \( A \) is partitioned into \( A := \begin{pmatrix} A_1 & A_{31} = O \\ A_4 = O & A_{21} \end{pmatrix} \). Otherwise, \( A := \begin{pmatrix} A_1 \\ A_4 = O \end{pmatrix} \). In both cases, \( A_1 \) is Leontief.

**Proof.** This corollary comes almost directly from Corollary 3.1. The remaining task is to prove \( A_{31} = O \), which can be accomplished as per the proof of Theorem 3.2. \( \Box \)

**Corollary 3.5.** Suppose \( I_A = \emptyset \). Then \( A := A_1 \) is Leontief and not partitioned.

**Proof.** This is a special case of Corollary 3.2. Since \( A \) has no nonpositive columns under Assumption 3.1, we obtain this corollary. \( \Box \)

**Corollary 3.6.** Suppose \( \overline{I}_A = \emptyset \). Then \( A \) is a sub-Leontief matrix. If \( J_A \neq \emptyset \), then \( A \) is partitioned into \( A := \begin{pmatrix} A_{21} & A_{22} \end{pmatrix} \). Otherwise (\( J_A = \emptyset \)), \( A := A_{21} (\neq O) \) is not partitioned.

**Proof.** This is a special case of Corollary 3.3. Since \( A \) has no zero column vectors under Assumption 3.1, we obtain this corollary. \( \Box \)

To summarize Theorem 3.2 and Corollaries 3.4–3.6, we categorize a pre-Leontief matrix \( A \) satisfying Assumption 3.1 and \( \overline{J}_A \neq \emptyset \) by partitioning as follows:

**Category I:** \( A := \begin{pmatrix} A_1 & A_{32} \\ O & A_{22} \end{pmatrix} \), where \( A_1 \) is Leontief and \( A_{22} \) is sub-Leontief.

**Category II:** \( A := \begin{pmatrix} A_1 & O \\ O & A_{21} \end{pmatrix} \), where \( A_1 \) is Leontief and \((A_{21} & A_{22})\) is sub-Leontief.

**Category III:** \( A := \begin{pmatrix} A_1 \\ O \end{pmatrix} \), where \( A_1 \) is Leontief.

**Category IV:** \( A := \begin{pmatrix} A_1 & O \\ O & A_{21} \end{pmatrix} \), where \( A_1 \) is Leontief and \( A_{21} \) is sub-Leontief.

**Category V:** \( A := A_1 \) is Leontief and not partitioned.

**Category VI:** \( A := \begin{pmatrix} A_{21} & A_{22} \end{pmatrix} \), where \( A \) is sub-Leontief.

**Category VII:** \( A := A_{21} \) is sub-Leontief and not partitioned.

A pre-Leontief matrix \( A \) is categorized into I, II, III or IV when \( \overline{I}_A \neq \emptyset \) and \( I_A \neq \emptyset \), into V when \( \overline{I}_A \neq \emptyset \) and \( I_A = \emptyset \), and into VI or VII when \( \overline{I}_A = \emptyset \) and \( I_A \neq \emptyset \). (See Figure 1.)

**Remark** Recall the partitioned matrix \( A \) in Theorem 3.2. Let \( A_{21} = (\gamma_1 \cdots \gamma_t) \) where \( t := \text{col}(A_{21}) \). Then we find the following two properties:
Figure 1: Submatrices belonging to Categories I to VII after partitioning a pre-Leontief
matrix $A$

(i) row($A_{21}$) ≥ 2 and $\gamma_j \neq 0$ for any $j = 1, \ldots, t$ where row($A_{21}$) is the number of rows of $A_{21}$.

(ii) $t \geq 2$ and $\gamma_1, \ldots, \gamma_t$ are linearly dependent.

The remark is verified as follows: $A$ with Assumption 3.1 implies from $A_{31} = O$ that $\gamma_j$ has exactly two nonzero components with opposite signs for any $j = 1, \ldots, t$. This justifies (i). The former part of (ii) (i.e., $t \geq 2$) is shown as follows: Choose any $x \in \mathbb{R}^n_+$ with $Ax \geq 0$. Letting $L := (O A_{21} A_{22})$, we see $Lx = 0$ since each row of $A_{21}$ is trivial in $A$. Suppose $t = 1$. Since any column of $A_{22}$ is trivial in $A$, we have $Lx = \gamma_1 \cdot (x)_{p+1} = 0$ where $p$ is the number defined in the proof of Theorem 2.1. From (i) of the remark, we see $(x)_{p+1} = 0$, which contradicts $p + 1 \in J_A$, implying $t \geq 2$. Let us show the latter part of (ii). From Lemma 2.2, there is an $x^* \in \mathbb{R}^n_+$ such that $Ax^* \geq 0$ and $(x^*)_j > 0$ for each $j = 1, \ldots, |J_A|$. Note $|J_A| = p + t$. Then we have $Lx^* = \sum_{i=1}^t \gamma_i \cdot (x^*)_p = 0$ and $(x^*)_j > 0$ $(j = p + 1, \ldots, p + t)$, implying the linear dependency.

3.3. Relations between a parent problem and its subproblems

In the following, we consider $P(A, b, c)$, which is defined in Section 1, with Assumption 3.1 and $b \geq 0$. We also call the specific problem $P(A, b, c)$ a parent problem. For a submatrix $B$ of $A$ and a feasible solution $x$ of $P(A, b, c)$, let $x_B$ (resp. $c_B$) be a subvector of $x$ (resp. $c$) corresponding to columns of $B$. The following lemma describes relations between the parent problem and its subproblems.

Lemma 3.1. Suppose that $A$ is partitioned into $A := \left( \begin{array}{c} A_1 & A_{31} = O & A_{32} \\ O & A_{21} & A_{22} \end{array} \right)$ as in Theorem 3.2 for the parent problem $P(A, b, c)$. Let $\left( \begin{array}{c} b_1 \\ b_2 \end{array} \right) (= b) \in \mathbb{R}^m_+$ be a partitioned vector corresponding to the partitioned matrix. Then the following statements (i)–(iii) hold.

(i) If there is a feasible solution $x$ of $P(A, b, c)$, then we have $x_{A_{22}} = 0$ and $b_2 = 0$, and $x_{A_1}$ (resp. $x_{A_{21}}$) is a feasible one of $P(A_1, b_1, c_{A_1})$ (resp. $P(A_{21}, b_2, c_{A_{21}})$ with $b_2 = 0$).
Algorithm for LP with Pre-Leontief Matrix

(ii) If \( u \) is a feasible solution of \( P(A_1, b_1, c_{A_1}) \), then \((u^\top, v^\top, 0^\top)^\top\) is feasible in \( P(A, b, c) \) for any feasible solution \( v \) of \( P(A_{21}, b_2, c_{A_{21}}) \) where \( b_2 = 0 \).

(iii) Suppose that \( P(A, b, c) \) is feasible and that \( c_{A_{21}}^\top x \in A_{21} \geq 0 \) for any feasible solution \( x \) of the parent problem. If there exists an optimal solution \( x^* \) of \( P(A, b, c) \), then \( x^*_{A_1} \) is optimal in \( P(A_1, b_1, c_{A_1}) \) whose optimal value is equal to that of the parent problem. Conversely, if we have an optimal solution \( u^* \) of \( P(A_1, b_1, c_{A_1}) \), then \((u^*)^\top, 0^\top, 0^\top)^\top\) is optimal in the parent problem and both of these optimal values are equal.

**Proof.** (i): Suppose that there is a feasible solution \( x \) of \( P(A, b, c) \). Divide \( x \) into \((x_{A_1}^\top, x_{A_{21}}^\top, x_{A_{22}}^\top)^\top\) so as to satisfy
\[
A_1 x_{A_1} + A_{32} x_{A_{22}} = b_1 \geq 0, \quad A_{21} x_{A_{21}} + A_{22} x_{A_{22}} = b_2 \geq 0, \quad x_{A_1} \geq 0, \quad x_{A_{21}} \geq 0, \quad x_{A_{22}} \geq 0.
\]
Since each column of \( A_{22} \) is trivial in \( A \) and each row of \( A_{21} \) is trivial in \( A \), we know from \( x_{A_{22}} = 0 \) that \( b_2 = A_{22} x_{A_{22}} = 0 \), which implies \( b_2 = 0 \). \( A_1 x_{A_1} = b_1 \), and \( A_{21} x_{A_{21}} = 0 \). Moreover, it follows from \( x_{A_1} \geq 0 \) and \( x_{A_{21}} \geq 0 \) that \( x_{A_1} \) (resp. \( x_{A_{21}} \)) is a feasible solution of \( P(A_1, b_1, c_{A_1}) \) (resp. \( P(A_{21}, 0, c_{A_{21}}) \)).

(ii): Suppose that \( P(A_1, b_1, c_{A_1}) \) has a feasible solution \( u \). Choose any \( v \geq 0 \) satisfying \( A_{21} v = 0 \). Letting \( x := (u^\top, v^\top, 0^\top)^\top \), it follows from \( A_{21} v = 0, A_1 u = b_1, \) and \( b_2 = 0 \) that
\[
A x = \begin{pmatrix}
A_1 u + A_{32} v \\
A_{21} v + A_{22} v
\end{pmatrix} = \begin{pmatrix}
b_1 \\
0
\end{pmatrix} = b,
\]
which implies from \( x \geq 0 \) that \( x \) is feasible in the parent problem.

(iii): Suppose that \( x^* \) is an optimal solution of \( P(A, b, c) \). Since the parent problem is feasible, we know from (i) that \( b_2 = 0 \) and \( x_{A_{22}}^* = 0 \), which implies that \( x_{A_1}^* \) is feasible in \( P(A_1, b_1, c_{A_1}) \). We show that \( x_{A_1}^* \) is also optimal in the subproblem. Assume that there is a feasible solution \( u \) of \( P(A_1, b_1, c_{A_1}) \) satisfying \( c_{A_1}^\top u < c_{A_1}^\top x_{A_1}^* \). It follows from \( b_2 = 0 \) and \( u \geq 0 \) that \( x^* = (u^\top, 0^\top, 0^\top)^\top \) is feasible in the parent problem. By \( c_{A_{21}}^\top x_{A_{21}}^* \geq 0 \), we have
\[
c^\top x^* = c_{A_1}^\top u < c_{A_1}^\top x_{A_1}^* \leq c_{A_1}^\top x_{A_1}^* + c_{A_{21}}^\top x_{A_{21}}^* = c^\top x^*,
\]
which contradicts the optimality of \( x^* \). Here, note \( x_{A_{22}}^* = 0 \). Hence, \( x_{A_1}^* \) is optimal in \( P(A_1, b_1, c_{A_1}) \). Similarly, we can prove the latter part in (iii).

\[
\square
\]

4. Triviality Judge

In this part, we propose our two efficient algorithms, called **RowTrivialityJudge** and **ColumnTrivialityJudge**, for determining whether the rows and columns of a matrix with Assumption 3.1 are trivial or not respectively.

**Proposition 4.1.** Given a problem \( P(A, b, c) \) with \( b \geq 0 \) and Assumption 3.1, let \( P(\bar{A}, \bar{b}, \bar{c}) \) be a variable-scaling problem of \( P(A, b, c) \), and \( N^0 := (G^0, u^0 := \infty, s^0) \) be a network corresponding to \( \bar{A} \). Then the following two claims are true:

(A) The \( i \)-th row of \( \bar{A} \) is nontrivial if and only if \( G^0 \) has at least one Type 2 fundamental subgraph with terminal vertex \( v_i \in V(G^0) \) where \( v_i \) is a vertex corresponding to the \( i \)-th row.

(B) \( P(A, b, c) \) is infeasible if and only if there exists an index \( i \in \{1, \ldots, m\} \) such that the \( i \)-th row of \( \bar{A} \) is trivial and \( (b_i)_0 > 0 \).

**Proof.** Denote by \( N := (G, u, \gamma) \) an extended network with respect to \( N^0 \).

Copyright © by ORSI. Unauthorized reproduction of this article is prohibited.
(Proof of (A)) First, we verify the necessity. Suppose that the $i_0$th row of $\bar{A}$ is nontrivial. Then we find from Lemma 2.8 that there exists $\mathbf{w} \in \mathbb{R}^+_n$ satisfying $\bar{A}\mathbf{w} \geq 0$ and $(\bar{A}\mathbf{w})_{i_0} > 0$. We can define a pseudoflow $g$ in $\mathcal{N}$ as $g(e_j) := (\mathbf{w})_j$ and $g(\bar{e}_j) := -\bar{\gamma}^0(e_j)(\mathbf{w})_j$ for $e_j \in E(G^0)$ ($j = 1, \ldots, n$). Note that $\bar{e}^N_i(v_i) = (\bar{A}\mathbf{w})_i \geq 0$ ($\forall i = 1, \ldots, m$) and $\bar{e}^N_i(v_{i_0}) = (\bar{A}\mathbf{w})_{i_0} > 0$. We find from Theorem 2.2 that $g$ can be decomposed into fundamental pseudoflows $g_1, \ldots, g_p$ of Type $t \in \{2, 4, 5, 6\}$ so that $g = \sum_{i=1}^p g_i$ for some positive number $p$. Among $g_i$ ($i = 1, \ldots, p$), at least one pseudoflow must be of Type 2 with terminal vertex $v_{i_0}$. Hence, $G^0$ has a Type 2 fundamental subgraph with terminal vertex $v_{i_0}$. Next, we show the sufficiency. Suppose that $G^0$ has a Type 2 fundamental subgraph with terminal vertex $v_{i_0}$, say $\mathcal{F}$. We can define a Type 2 fundamental pseudoflow $g$ with terminal vertex $v_{i_0}$ such that $G_+(g) = \mathcal{F}$. Note that $\bar{e}^N_i(v_i) = 0$ ($i \in \{1, \ldots, m\} \setminus \{i_0\}$) and $\bar{e}^N_i(v_{i_0}) > 0$. Defining $\mathbf{w} \in \mathbb{R}^+_n$ as $(\mathbf{w})_j := g(e_j)$ ($j = 1, \ldots, n$), we find that $\bar{A}\mathbf{w} \geq 0$ and $(\bar{A}\mathbf{w})_{i_0} > 0$. From Lemma 2.8, the $i_0$th row of $\bar{A}$ is nontrivial.

(Proof of (B)) Note that $\bar{b} \geq 0$ holds. Since the sufficiency is obvious, we verify the necessity only. Assume that each index $i \in \{1, \ldots, m\}$ satisfies either that $i$th row of $\bar{A}$ is nontrivial or $(\bar{b})_i = 0$. By suitable permutation of rows in $\bar{A}$ and corresponding permutation of components in $\bar{b}$, we partition $\bar{A}$ and $\bar{b}$ as $\bar{A} := \begin{pmatrix} U \\ L \end{pmatrix}$ and $\bar{b} := \begin{pmatrix} b_U \\ b_L \end{pmatrix}$, where all rows of $U$ (resp. $L$) are nontrivial (resp. trivial) in $\bar{A}$ ($U$ or $L$ may have no rows). From the assumption, if $i$th row of $\bar{A}$ is trivial then $(\bar{b})_i = 0$, which implies $b_L = 0$. Suppose that $\bar{A}$ belongs to Category II, which implies $\bar{A} = \begin{pmatrix} A_1 & O & A_{32} \\ O & A_{21} & A_{22} \end{pmatrix}$. Then we see from Proposition 2.4 and row trivialities that there exist $\mathbf{w}_U \in \mathbb{R}^+_n$ and $\mathbf{w}_L \in \mathbb{R}^{(|V| - p)}_+$ such that $A_1\mathbf{w}_U = b_U$ and $A_{21}\mathbf{w}_L = 0$, respectively. Note that $A_1$ is Leontief. Letting $\mathbf{w}^+ := (\mathbf{w}_U^\top \mathbf{w}_L^\top 0)$, and we find that $\mathbf{w} \in \mathbb{R}^+_n$ and $\bar{A}\mathbf{w} = \bar{b}$. Similarly, we can show the feasibility for $\bar{A}$ belonging each of the remaining categories. To summarize, we see that $P(\bar{A}, \bar{b}, \bar{c})$ is feasible. 

Similarly as Proposition 4.1, we can show the following proposition.

**Proposition 4.2.** Let $\mathcal{N}^0 := (G^0, v^0, \bar{\gamma}^0)$ be the network corresponding to a variable-scaling problem $P(\bar{A}, \bar{b}, \bar{c})$ with $\bar{b} \geq 0$ and Assumption 3.1. Then the $j$th column of $\bar{A}$ is nontrivial if and only if $G^0$ has at least one of fundamental subgraphs of Type $t \in \{2, 4, 5, 6\}$ containing the edge $e_j \in E(G^0)$ where $e_j$ is an edge corresponding to the $j$th column.

By MBF we mean Moore-Bellman-Ford algorithm for finding shortest paths from a source. Denote by $r_v$ (resp. $c_v$) the row (resp. column) index of $\bar{A}$ corresponding to a vertex $v$ (resp. an edge $e$) in $G^0$. Given a cycle $C$ and a path $P$ in $G^0$ satisfying $V(P) \cap V(C) \neq \emptyset$, **Intersection**($P, C, \text{mode}$) is the following function, which returns a vertex shown below.

**Intersection**($P, C, \text{mode}$)

**Output** The vertex $v$ nearest (resp. farthest) from its initial vertex of $P$ such that $v \in V(P) \cap V(C)$, if mode is “min” (resp. “max”).

**S1** Denote the trail of $P$ by $(w_1, e_1, \ldots, w_{q-1}, e_{q-1}, w_q)$ where $q := |V(P)|$. If mode is “min” (resp. “max”), let $i^*$ be the minimum (resp. maximum) index $i \in \{1, \ldots, q\}$ with $w_i \in V(C)$. Set $v \leftarrow w_i$, and return $v$.

In the following paragraphs from (1) to (3), we give three key procedures in order to determine the trivialities of rows and columns of a matrix.

(1) We choose an edge $e^*$ in network $\mathcal{N}^0 := (G^0, v^0, \gamma^0)$. **DetectType3Graph**($v^-, \mathcal{N}^0$) is the following procedure for finding a Type 3 fundamental subgraph with $v^- := \partial_{G^0}(e^*)$ as
its initial vertex in \(G^0\). More formally, the procedure is described as follows.

**DetectType3Graph\((v^-, N^0)\)**

**(Output)** An edge set \(F^-\) of a Type 3 fundamental subgraph with \(v^-\) as its initial vertex in \(G^0\) if it exists, and a vertex set \(S^-\) in \(G^0\) reachable from \(v^-\).

**(S1)** Find a set \(S^-\) of vertices in \(G^0\) reachable from \(v^-\), and let \(H^- := G^0[S^-]\) and obtain a length function \(l^- : E(H^-) \rightarrow \mathbb{R}\) by \(l^-(e) := \log \gamma^0(e) (e \in E(H^-))\). By using MBF with \(v^-\) as a source, find a negative cycle \(C^-\) with respect to \(l^-\) (i.e., \(l^- (C^-) < 0\)) in \(H^-\). If we have such a cycle \(C^-\) then go to \((S2)\), stop with \(F^- := \emptyset\) and \(S^-\) as output otherwise.

**(S2)** Choose a vertex \(w^-\) in \(C^-\) and find a shortest \(v^- - w^-\)-path \(P^-\) in \(H^-\) where the length of each edge in \(P^-\) is 1. Let

\[
v \leftarrow \text{Intersection}(P^-, C^- \cap \text{"min"}), \quad F^- \leftarrow E(P^-_{[v^- w^-]} \cup E(C^-).\]

Terminate with \(F^-\) and \(S^-\) as output.

**Proposition 4.3.** \(\text{DetectType3Graph}(v^-, N^0)\) runs in \(O(m^- \cdot n^-)\) time where \(m^- = |V(H^-)|\) and \(n^- = |E(H^-)|\).

**Proof.** (S1) of \(\text{DetectType3Graph}(v^-, N^0)\) runs in \(O(m^- \cdot n^-)\) time. In fact, we may apply MBF only once in order to detect a negative cycle \(C^-\) with respect to \(l^-\) defined in (S1) of the procedure. The complexity of (S2) in the procedure is \(O((m^-)^2)\). In fact, one application of Dijkstra algorithm gives us a desired shortest path in \(H^-\) where \(H^-\) is a graph defined in (S1) of the procedure. Thus, we have this proposition. Note that \(n^- \geq m^-\) if the procedure visits (S2). \(\square\)

**(2)** \(\text{DetectType2Graph}(v^+, N^0)\) is the following procedure for finding a Type 2 fundamental subgraph with \(v^+ := \partial^+_{G^0}(e^*)\) as its end vertex in \(G^0\) where \(e^*\) is the edge in (1). More formally, the procedure is described as follows.

**DetectType2Graph\((v^+, N^0)\)**

**(Output)** An edge set \(F^+\) of a Type 2 fundamental subgraph with \(v^+\) as its end vertex in \(G^0\) if it exists, and a vertex set \(S^+\) in \(G^0\) reachable to \(v^+\).

**(S1)** Find a set \(S^+\) of vertices in \(G^0\) reachable to \(v^+\), and let \(H^+ := G^0[S^+]\) and obtain a length function \(l^+ : E(H^+) \rightarrow \mathbb{R}\) by \(l^+(e) := \log \gamma^0(e) (e \in E(H^+))\). By using a reverse version of MBF with \(v^+\) as a sink, find a negative cycle \(C^+\) with respect to \(l^+\) (i.e., \(l^+ (C^+) < 0\)) in \(H^+\). If there is such a cycle \(C^+\) then go to \((S2)\), terminate with \(F^+ := \emptyset\) and \(S^+\) as output otherwise.

**(S2)** Choose a vertex \(w^+\) in \(C^+\) and find a shortest \(w^+ - v^+\)-path \(P^+\) in \(H^+\) where the length of each edge in \(P^+\) is 1. Let

\[
v \leftarrow \text{Intersection}(P^+, C^+ \cap \text{"max"}), \quad F^+ \leftarrow E(P^+_{[w^+ v^+]} \cup E(C^+).\]

Terminate with \(F^+\) and \(S^+\) as output.

Similarly as Proposition 4.3, we have:

**Proposition 4.4.** \(\text{DetectType2Graph}(v^+, N^0)\) runs in \(O(m^+ \cdot n^+)\) time where \(m^+ = |V(H^+)|\) and \(n^+ = |E(H^+)|\). \(\square\)
(3) We make the following assumption for \( S^+ + H^+ \) in (2) of DetectType2Graph\((v^+, N^0)\).

\[ (A') \] The edge \( e^* \) satisfies \( \partial_{G^0}(e^*) \subseteq S^+ \), and there is no flow-generating cycle in \( H^+ \).

Note that \( e^* \in E(H^+) \) and that there is no negative cycle in \( H^+ \) with respect to \( l^+ \) under the assumption \((A')\). DetectType4Graph \((e^*, N^0)\) with \( F^0 \) as output is the following procedure, which uses an idea proposed by Wayne [29], for finding a Type 4 fundamental subgraph with \( e^* \) in \( G^0 \) if it exists.

DetectType4Graph\((e^*, N^0)\)

\textbf{(Output)} An edge set \( F^0 \) of a Type 4 fundamental subgraph with \( e^* \) in \( H^+ \) if it exists.

\begin{enumerate}[(S1)]
  \item Find a vertex set \( S^+ \) in \( G^0 \) reachable to \( v^+ := \partial_{G^0}(e^*) \), and let \( H^+ : = G^0[S^+] \) and define \( l^+(e) : = -\log \gamma^0(e) \ (e \in E(H^+)) \). By applying a reverse version of MBF with \( v^+ \) as a sink, calculate the length \( d(v) \) of a shortest \( v-v^+ \)-path with respect to \( l^+ \) in \( H^+ \) for each \( v \in V(H^+) \). Compute reduced lengths \( \overline{t^+}(e) := l^+(e) + d(\partial_{G^0}(e)) - d(\partial_{G^0}(e)) \) \((e \in E(H^+))\), and find an edge set \( R := \{e \in E(H^+) : \overline{t^+}(e) = 0\} \) consisting of zero reduced lengths. Find a cycle \( C^0 \) with \( e^* \) in \( G^0[R] \). If there is such a cycle \( C^0 \) then terminate with output \( F^0 := E(C^0) \), terminate with output \( F^0 := \emptyset \) otherwise.
\end{enumerate}

Similarly as Propositions 4.3, we can show the following proposition.

**Proposition 4.5.** DetectType4Graph\((e^*, N^0)\) runs in \( O(m^+ \cdot n^+) \) time under the assumption \((A')\). \qed

Now, we propose ColumnTrivialityJudge\((N^0, \bar{A})\) which is the following algorithm for judging whether the column with index \( c_e \) of \( \bar{A} \) is trivial or not for each edge \( e \) in \( G^0 \).

ColumnTrivialityJudge\((N^0, \bar{A})\)

\textbf{(Output)} A set \( X \) (resp. \( Y \)) of edges which correspond to trivial (resp. nontrivial) columns of \( \bar{A} \).

\begin{enumerate}[(S1)]
  \item Let \( D \leftarrow E(G^0), X \leftarrow \emptyset, \) and \( Y \leftarrow \emptyset \).
  \item While \( D \neq \emptyset \), do the following (S2.1) – (S2.3) after choosing an edge \( e^* \) in \( D \) and letting \( v^- := \partial_{G^0}(e^*) \) and \( v^+ := \partial_{G^0}(e^*) \).
    \begin{enumerate}[(S2.1)]
      \item Run DetectType3Graph\((v^-, N^0)\) with \( F^- \) and \( S^- \) as output.
      \item Run DetectType2Graph\((v^+, N^0)\) with \( F^+ \) and \( S^+ \) as output.
      \item If \( F^+ \neq \emptyset \) do (S2.3.1), otherwise do (S2.3.2).
    \end{enumerate}
    \begin{enumerate}[(S2.3.1)]
      \item If \( F^- = \emptyset \) do (S2.3.1a), otherwise do (S2.3.1b).
        \begin{enumerate}[(S2.3.1a)]
          \item If \( v^- \notin V(G^0[F^+]) \), then let \( Y \leftarrow Y \cup F^+ \cup \{e^*\} \) and \( D \leftarrow D \setminus (F^+ \cup \{e^*\}) \).
        \end{enumerate}
      \item Find a flow-absorbing (resp. flow-generating) cycle \( C^- \) (resp. \( C^+ \)) in \( G^0[F^-] \) (resp. \( G^0[F^+] \)). If \( V(C^-) \cap V(C^+) \neq \emptyset \), then let \( Y \leftarrow Y \cup E(C^-) \cup E(C^+) \), \( D \leftarrow D \setminus (E(C^-) \cup E(C^+)) \), and return to (S2). Otherwise \((V(C^-) \cap V(C^+) = \emptyset)\), choose a vertex \( w^+ \) in \( C^+ \) and a vertex \( w^- \) in \( C^- \), and find a shortest \( w^+ - w^- \)-path \( P \) in \( G^0[S^- \cup S^+] \) where the length of each edge in \( P \) is 1. Let \( x^- \leftarrow \text{Intersection}(P, C^-,”min”), x^+ \leftarrow \text{Intersection}(P_{[w^+ \to x^-]}, C^+,”max”), \)
        \begin{enumerate}[(S2.3.2)]
          \item If \( v^- \notin S^+ \), then let \( X \leftarrow X \cup \{e^*\} \) and \( D \leftarrow D \setminus \{e^*\} \), and return to (S2).
        \end{enumerate}
      \item If \( v^- \notin S^+ \), run DetectType4Graph\((e^*, N^0)\) with \( F^0 \) as output.
    \end{enumerate}
  \end{enumerate}
If $F^0 = \emptyset$, then let $X \leftarrow X \cup \{e^*\}$ and $D \leftarrow D \setminus \{e^*\}$, and return to (S2).
Otherwise ($F^0 \neq \emptyset$), return to (S2) after letting $Y \leftarrow Y \cup F^0$ and $D \leftarrow D \setminus F^0$.

(S3) Terminate with $X$ and $Y$ as output.

It is straightforward that the above algorithm is valid from Proposition 4.2 and the following proposition which is easy to verify.

**Proposition 4.6.** Let $e^* \in D$ be selected in (S2) of 

$\text{ColumnTrivialityJudge}(\mathcal{N}^0, \bar{A})$.

(i) The case when $F^+ \neq \emptyset$ and $F^- = \emptyset$: By (S2.3.1a), we find that if $v^- \notin V(G^0[F^+])$ (resp. $v^- \in V(G^0[F^+])$) then we obtain a fundamental subgraph of Type 2 with $e^* \notin F^+$ (resp. $e^* \in F^+)$.

(ii) The case when $F^+ \neq \emptyset$ and $F^- \neq \emptyset$: By (S2.3.1b), we find that if $V(C^-) \cap V(C^+) \neq \emptyset$ (resp. $V(C^-) \cap V(C^+) = \emptyset$) then we obtain a fundamental subgraph of Type 6 with $e^* \in F^+ \cup F^-$ (resp. of Type 5 with $e^* \notin F^+ \cup F^-$).

(iii) The case when $F^+ = \emptyset$: By (S2.3.2), if $v^- \in S^+$ and $F^0 \neq \emptyset$ then we find a fundamental subgraph of Type 4 with $e^* \in F^0$, otherwise it is impossible for $e^*$ to belong to an edge set of a fundamental subgraph of Type $t \in \{2, 4, 5, 6\}$. 

From Proposition 4.6, we have the following theorem:

**Theorem 4.1.** 

$\text{ColumnTrivialityJudge}(\mathcal{N}^0, \bar{A})$ runs in $O(mn^2)$ time.

**Proof.** Each of (S1) and (S3) of the algorithm takes at most $O(n)$ time. (S2) of the algorithm is repeated at most $n$ times. In fact, $|D|$ decreases at least one every time (S2) is repeated. Consider one loop from (S2.1) to (S2.3). From Proposition 4.3, (S2.1) takes $O(mn)$ time. From Proposition 4.4, we can carry out (S2.2) in $O(mn)$ time. (S2.3.1) takes $O(n)$ time. From Proposition 4.4, (S2.3.2) takes $O(mn)$ time. Summarizing the above discussion, we have this theorem.

Next, we propose 

$\text{RowTrivialityJudge}(\mathcal{N}^0, \bar{A})$ which is the following easy algorithm for judging whether the row with index $r_v$ of $\bar{A}$ is trivial or not for each vertex $v$ in $G^0$.

$\text{RowTrivialityJudge}(\mathcal{N}^0, \bar{A})$

(\textbf{Output}) A set $T$ (resp. $N$) of vertices corresponding to trivial (resp. nontrivial) rows of $\bar{A}$.

(S1) Let $U \leftarrow V(G^0), T \leftarrow \emptyset$, and $N \leftarrow \emptyset$.

(S2) While $U \neq \emptyset$, do the following (S2.1).

(S2.1) Choose a vertex $v$ in $U$ and run $\text{DetectType2Graph}(v, \mathcal{N}^0)$ with $F^+$ and $S^+$ as output. If $F^+ = \emptyset$ then let $T \leftarrow T \cup \{v\}$, let $N \leftarrow N \cup \{v\}$ otherwise. Return to (S2) after letting $U \leftarrow U \setminus \{v\}$.

Similarly as Theorem 4.1, we can show the following theorem.

**Theorem 4.2.** 

$\text{RowTrivialityJudge}(\mathcal{N}^0, \bar{A})$ runs in $O(m^2n)$ time.

We can efficiently determine the feasibility of a system of $\bar{A}x = b$ and $x \geq 0$ with $b \geq 0$, after executing $\text{RowTrivialityJudge}$ with $T$ and $N$ as output.

$\text{FeasibilityJudge}(\bar{A}, b, T)$

(\textbf{Output}) Feasibility of a system of $\bar{A}x = b$ and $x \geq 0$.

(S1) Let $I_T := \{r_v : v \in T\}$ be the set of indices corresponding to vertices in $T$. If there exists an index $i \in I_T$ with $(b)_i > 0$, then terminate with “The system is infeasible”,

Copyright © by ORSI. Unauthorized reproduction of this article is prohibited.
otherwise terminate with “The system is feasible”.

We have the following theorem.

**Theorem 4.3.** Given the set $T$ of vertices corresponding to trivial rows of $\bar{A}$, Feasibility-Judge $(\bar{A}, b, T)$ runs in $O(m)$ time.

**Corollary 4.1.** We can determine the feasibility of problem $GT$ in $O(m^2 n)$ time.

**Proof.** We can transform problem $GT$ into a GMCU with some graph such that the number of the vertices (resp. edges) is $m + 5$ (resp. $n + 6$). After executing RowTrivialityJudge and FeasibilityJudge, we have this corollary from Theorems 4.2 and 4.3.

5. Our Algorithms

5.1. Main algorithm

Given the parent problem $P(A, b, c)$ with Assumption 3.1 and $b \geq 0$, denote its dual problem by $D(A, c, b)$. Then, our main algorithm is described as follows.

**FlowBasedMainAlgorithm**($A, b, c$)

**(Output)** Feasibility of $P(A, b, c)$ and its optimal solution if it exists.

(S1) Make a variable-scaling problem $P(\bar{A}, b, \bar{c})$ from the problem $P(A, b, c)$, and construct network $\mathcal{N}^0 := (G^0, u^0 := \infty, \gamma^0)$ corresponding to $\bar{A}$. Find a set $T$ of vertices corresponding to trivial rows of $\bar{A}$ by RowTrivialityJudge($\mathcal{N}^0, \bar{A}$). Check the feasibility of $P(\bar{A}, b, \bar{c})$ by FeasibilityJudge($\bar{A}, b, T$). If the system is infeasible, then we stop.

(S2) By running ColumnTrivialityJudge ($\mathcal{N}^0, \bar{A}$) with $X$ and $Y$ as output, find two sets $J_A$ and $\bar{J}_A$ from $X$ and $Y$, respectively. Also, find a set $I_A$ corresponding to $T$. Let $I_A := \{1, \ldots, m\} \setminus I_A$.

(S3) Partition $A$ by PermuteRowsColumns($A, I_A, \bar{I}_A, J_A, \bar{J}_A$), categorize $A$ into one of Categories I - VII, and select either (S3.1) or (S3.2).

(S3.1) If $A$ belongs to Category II, IV, VI or VII (i.e., $A$ includes (or coincides with) $A_{21}$), go to (S4).

(S3.2) If $A$ belongs to Category I, III or V (i.e., $A$ includes $A_1$ but not $A_{21}$), go to (S5).

(S4) We transform $P(A_{21}, 0, c_{A_{21}})$ into a variable-scaling problem $P(M, 0, d)$. Letting $G^0$ be a graph with $M$ as its incident matrix and denote by $\mathcal{N}^0 := (G^0, u^0 := \infty, \gamma^0)$ a corresponding network again, we decide whether or not there is a negative circuit in the extended network $\mathcal{N} := (G, u, \gamma)$. If such a circuit exists, then we stop (the parent problem is unbounded). Otherwise, if $A$ belongs to Category II or IV (i.e., $A$ includes $A_1$) then go to (S5), if not (i.e., $A$ belongs to VI or VII) then stop (the optimal solution of the parent problem is 0).

(S5) If $D(A_1, c_{A_1}, b_1)$ is infeasible, then we stop. (The parent problem is unbounded. Note that the problem is feasible.) Otherwise, we find an optimal solution $y$ of the dual problem. Then we perform FindPrimalOptimalSolution($A_1, b_1, c_{A_1}, y$) and stop (an optimal solution of the parent problem is easily obtained).

**Remark** In (S5), we know from [24] that an optimal solution of $D(A_1, c_{A_1}, b_1)$ is a maximal vector in the feasible region and that such a vector can be found by an algorithm in [3] whose complexity is given in Theorem 5.1 below. An algorithm to find a feasible solution of $P(A_1, b_1, c_{A_1})$ will be shown later. Note that the feasible solution is also optimal.
Theorem 5.1. ([3], Section 5) An algorithm by Cohen and Megiddo can solve a TVPI problem in $O(mn \log n + mn^2 \log^2 m)$ time where $m$ (resp. $n$) is the number of variables (resp. inequalities). Note that Adler and Cosares quoted an algorithm by Megiddo [13] whose complexity is $O(m^2 n \log n)$.

The following algorithm finds an optimal solution of $P(A_1, b_1, c_{A_1})$ if it exists with the aid of an optimal solution $y$ of the dual problem where its arguments $A$, $b$, and $c$ are used in place of $A_1$, $b_1$, and $c_{A_1}$, respectively.

**FindPrimalOptimalSolution**($A, b, c, y$)

(Output) An optimal solution of $P(A, b, c)$.

(S1) To import the complementary slackness condition into $P(A, b, c)$, let $l \leftarrow \text{col}(A)$ and $I \leftarrow \{i \in \{1, \ldots, l\} : (c - A^T y)_i > 0\}$. If $|I| = l$, then we stop with an optimal solution 0. After deleting all the columns $j \in I$ from $A$, denote by $A$ the remaining matrix.

(S2) We transform $P(A, b, c)$ into a variable-scaling problem $P(\tilde{A}, \tilde{b}, \tilde{c})$. Let $\mathcal{N}^0 := (G^0, v^0 := \infty, \gamma^0)$ be a network corresponding to $A$. Extend $\mathcal{N}^0$ to $\mathcal{N} := (G, u, \gamma)$ as before. Let $g(e) \leftarrow 0$ ($e \in E(G)$) and $B \leftarrow \{v \in V(G) : (b)_v > 0\}$.

(S3) While $B \neq \emptyset$, do the following (S3.1) and (S3.2) after choosing $v \in B$.

(S3.1) After running **DetectType2Graph**($v, \mathcal{N}^0$) with $F^+$ and $S^+$ as output, let $F \leftarrow G[F^+ \cup F^+]$.

(S3.2) Find a Type 2 pseudoflow $h$ in $\mathcal{N}$ with positive flows in edges of $F$ such that

$$
\epsilon^N_h(v) = (b)_v > 0, \quad \epsilon^N_h(w) = 0 \quad (w \in V(F) \setminus \{v\}).
$$

Letting $g \leftarrow g + h$ and $B \leftarrow B \setminus \{v\}$, return (S3).

(S4) Reduce $g$ to a feasible solution of the parent problem and stop with such a solution.

**Proposition 5.1.** For each $v \in B$ in (S3), we can find a Type 2 fundamental pseudoflow $h$ in $\mathcal{N}$ with $v$ as the terminal vertex in **FindPrimalOptimalSolution**.

**Theorem 5.2.** **FindPrimalOptimalSolution**($A_1, b_1, c_{A_1}, y$) runs in strongly polynomial time, whose complexity is $O(m^2 n)$.

**Proof.** From $A \in \mathbb{R}^{m \times n}$, each of (S1) and (S2) takes $O(mn)$ time. (S3) is repeated at most $m$ times. Consider one loop of (S3). We see from Proposition 4.4 that the complexity in (S3.1) is $O(mn)$. (S3.2) takes $O(n)$ time. The complexity in (S4) is $O(n)$. Summarizing the foregoing discussion, the total complexity of **FindPrimalOptimalSolution** is $O(m^2 n)$.

5.2. Complexity

Assuming $m \leq n$, the complexity of our main algorithm is as follows.

**Theorem 5.3.** If $m \leq n$, our main algorithm runs in strongly polynomial time, and its complexity is given by $O(mn \cdot \max\{n, m \log^2 m\})$.

**Proof.** From Theorems 4.2 and 4.3, (S1) takes $O(m^2 n)$ time. From Theorem 4.1, we see that the complexity of (S2) is $O(mn^2)$. From Theorem 5.1, the feasibility problems of dual problems in (S3) and (S4) can be solved in $O(mn \log n + m^2 n \log^2 m)$ time. From Theorem 5.2, (S5) takes $O(m^2 n)$ time. Comparing $mn^2$ with $m^2 n \log^2 m$, we obtain this result.

Copyright © by ORSJ. Unauthorized reproduction of this article is prohibited.
6. Conclusion
We considered an LP with a specific pre-Leontief coefficient matrix and proposed a new strongly polynomial time flow-based algorithm. Our problem is similar to that of Adler and Cosares [1] studied in the sense that a demand function given in their model is satisfied at each vertex except for a distinguished one while we impose the demand condition on all the vertices. By incorporating trivialities/nontrivialities of the rows and columns of a matrix introduced by Veinott, we derived a new matrix decomposition theorem which refined their theorem in [1]. By the help of our matrix decomposition theorem and the first efficient method for deciding such trivialities and nontrivialities, we developed a strongly polynomial time flow-based algorithm. By the way, they do not provide any method for finding how to divide the columns of the coefficient matrix into two sets when we partition the matrix. So, our complexity is the same as that of [1] when a coefficient matrix partitioned is given. Especially, the developed algorithm can also determine the feasibility of the generalized transshipment problem, and our complexity is much smaller than that of [1]. Importantly, finding an optimal flow for such a problem in strongly polynomial time has been a longstanding issue where the given network is capacitated. It would be prudent and fruitful for future research to implement our algorithm while tackling the minimum cost flow problem.

Acknowledgement We deeply thank two anonymous reviewers who provided a great deal of insight and expertise that significantly assisted our research.

References
[1] I. Adler and S. Cosares: A strongly polynomial algorithm for a special class of linear programs. Operations Research, 39 (1991), 955–960.
[2] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin: Network Flows (Prentice Hall, New Jersey, 1993).
[3] E. Cohen and N. Megiddo: Improved algorithms for linear inequalities with two variables per inequality. SIAM J. Comput., 23-6 (1994), 1313–1347.
[4] L.K. Fleischer and K.D. Wayne: Fast and simple approximation schemes for generalized flow. Mathematical Programming, 91 (2002), 215–238.
[5] A.V. Goldberg, S.A. Plotkin, and É. Tardos: Combinatorial algorithms for the generalized circulation problem. Mathematics of Operations Research, 16 (1991), 351–381.
[6] D. Goldfarb and Z. Jin: A faster combinatorial algorithm for the generalized circulation problem. Mathematics of Operations Research, 21 (1996), 529–539.
[7] D. Goldfarb, Z. Jin, and Y. Lin: A polynomial dual simplex algorithm for the generalized circulation problem. Mathematical Programming, 91 (2002), 271–288.
[8] D. Goldfarb, Z. Jin, and J.B. Orlin: Polynomial-time highest-gain augmenting path algorithms for the generalized circulation problem. Mathematics of Operations Research, 22 (1997), 793–802.
[9] M. Gondran and M. Minoux: Graphs and Algorithms (Wiley Series in Discrete Mathematics and Optimization, 1984).
[10] W.S. Jewell: Optimal flow through networks with gains. Operations Research, 10 (1962), 476–499.
[11] N. Karmarkar: A new polynomial-time algorithm for linear programming. Combinatorica, 4 (1984), 373–395.
[12] L.G. Khachiyan: Polynomial algorithms in linear programming. Zh. Vychisl. Mat. Mat. Fiz., 20 (1980), 53–72.
Algorithm for LP with Pre-Leontief Matrix

[13] N. Megiddo: Towards a genuinely polynomial algorithm for linear programming. *SIAM J. Compt.*, **12-2** (1983), 347–353.

[14] J.D. Oldham: Combinatorial approximation algorithms for generalized flow problems. *Journal of Algorithms*, **38** (2001), 135–169.

[15] N. Oliver and L.A. Vegh: A simpler and faster strongly polynomial algorithm for generalized flow maximization. *Journal of the ACM*, **67-2** (2020), 1–26.

[16] K. Onaga: Optimal flows in general communication networks. *Journal of the Franklin Institute*, **283** (1967), 308–327.

[17] T. Radzik: Faster algorithms for the generalized network flow problem. *Mathematics of Operations Research*, **23** (1998), 69–100.

[18] T. Radzik: Improving time bounds on maximum generalized flow computations by contracting the network. *Theoretical Computer Science*, **312** (2004), 75–97.

[19] M. Shigeno: A survey of combinatorial maximum flow algorithms on a network with gains. *Journal of the Operations Research Society of Japan*, **47** (2004), 244–264.

[20] É. Tardos and K.D. Wayne: Simple generalized maximum flow algorithms. In *International Conference on Integer Programming and Combinatorial Optimization* (Springer, 1998), 310–324.

[21] K. Truemper: On max flows with gains and pure min-cost flows. *SIAM Journal on Applied Mathematics*, **32** (1977), 450–456.

[22] P.M. Vaidya: Speeding up linear programming using fast matrix multiplication. In *Proc. 30th IEEE Annual ACM Symposium on Foundations of Computer Science* (1989), 332–337.

[23] L.A. Vegh: A strongly polynomial algorithm for generalized flow maximization. *Mathematics of Operations Research*, **42-1** (2017), 179–211.

[24] A.F. Veinott: Extreme points of Leontief substitution systems. *Linear Algebra and its applications*, **1** (1968), 181–204.

[25] K.D. Wayne: A polynomial combinatorial algorithm for generalized minimized cost flow. *Mathematics of Operations Research*, **27-3** (2002), 445–459.

Akira Nakayama  
Symbiotic Systems Science  
Fukushima University  
Kanayagawa 1, Fukushima City  
Fukushima 960-1296, Japan  
E-mail: nakayama@sss.fukushima-u.ac.jp