East-West J. of Mathematics: Vol. 13, No 1(2011) pp. 67-78

A STRONGLY POLYNOMAL- TIME ALGORITHM FOR A CLASS OF INTEGER PROGRAMMING PROBLEMS

Vo Van Tuan Dung
HoChiMinh City University of Industry
Ho Chi Minh city, Vietnam
e-mail: vvtdung@gmail.com

Abstract

In this paper a strongly polynomial-time algorithm is proposed for solving exactly a class of integer programming problems which is basically to minimize the pointwise maximum of n affine functions under m semi-assignment constraints and upper bound constraints. The algorithm is based on a numbering technique for improving feasible solutions.

1 INTRODUCTION

Given an $m \times n$ matrix $A = (a_{ij})_{m\times n}$, where $a_{ij} \in \{0, 1\}$, and given positive integer numbers p_i $(0 < p_i \le n), i = 1, 2, \ldots, m$, consider the following optimization problem:

$$(P) \max_{1\le j\le n} \sum_{i=1}^{m} x_{ij} \to \min \tag{1}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = p_i,\ i = 1, 2, \ldots, m \tag{2}$$

$$0 \le x_{ij} \le a_{ij}, x_{ij} \text{ is an integer}, i = 1, 2, \ldots, m, j = 1, 2, \ldots, n. \tag{3}$$

This formulation shows that (P) is an integer nonlinear programming problem. However, it is easily seen that (P) may be reduced to finding an integer

Key words: Integer programming problem, maximum flow problem, strongly polynomial-time algorithm.
2000 AMS Mathematics Subject Classification: Primary 90C08; Secondary 65K05, 49M29

optimal solution of the following integer linear programming problem.

$$(Q)\min\left\{z|\sum_{j=1}^{n}x_{ij}\geq p_i,\forall i;\sum_{i=1}^{m}x_{ij}\leq z,\forall j;0\leq x_{ij}\leq a_{ij},\forall i,j;0\leq\text{is an integer}\right\}.$$

(Since z is an integer, if (Q) has a feasible solution then (Q) has at least one integer optimal solution).

It should be noted that requiring integer values for the x_{ij} variables is essential, because problem (P) may exhibit an integrality gap, i.e. if the integer restriction on x_{ij} is deleted the optimal value (1) may be reduced, as the example given at the end of this paper illustrates.

Problem (P) has $s=\sum_{i,j}a_{ij}$ $0-1$ variables and m linear constraints besides s upper bound constraints in (3). In addition, equality constraint (2) may also be replaced by the following inequality constraint without changing the optimal solution value to (P).

$$\sum_{j=1}^{n}x_{ij}\geq p_i,\ i=1,2,\ldots,m$$

Problem (P) can be interpreted as follows: there are m students and n subjects which the students have to choice to study. The number of subjects required for student i is p_i. The coefficient a_{ij} represents the willingness of student i to study subject j ($a_{ij}=1$ if student i is willing to study subject j and $a_{ij}=0$ if not). The question is how to arrange the students to study the subjects so that each student can study the number of subjects required for him and so that the numbers of students studying different subjects are as similar as possible.

One more application of (P) can be stated as follows. There are m seminars, each of which takes one day in an appropriate seminar-room. Let us consider n seminar-rooms available for the seminars. We suppose that

$$a_{ij}=\begin{cases}1, & \text{if seminar-room } j \text{ is admissible for seminar } i,\\ 0, & \text{otherwise } (i=1,2,\ldots,m; j=1,2,\ldots,n).\end{cases}$$

Moreover, we assume that each seminar occupies only one appropriate seminar room. The problem is to determine an assignment of the seminars to the semainar-rooms in order to minimize the number of days needed to hold all seminars.

Let us denote

$$x_{ij}=\begin{cases}1, & \text{if seminar } i \text{ is assigned to seminar-room } j,\\ 0, & \text{otherwise } (i=1,2,\ldots,m; j=1,2,\ldots,n).\end{cases}$$

Obviously, problem (P) with $p_i=1, i=1,2,\ldots,m$, is a mathematical model for the above problem. Objective function (1) in this case indicates the number of days needed to hold all seminars.

The model of problem (P) was studied in [5] and [6]. A polynomial-time algorithm is described in [5], which reduces problem (P) to solving a finite number of maximum flow problems with $p = m + n + 2 \approx O(m + n)$ nodes and $q = \sum_{i,j} a_{ij} + m + n \approx O(mn)$ arcs. Its running time is $O(O_{MF} \times \log_2 n)$, where O_{MF} is the running time of any polynomial-time maximum flow algorithm. For instance, for Edmonds-Karp's (1969) algorithm, O_{MF} is $O(pq^2)$ and for Dinic's (1970) algorithm it is $O(p^2 q)$, where p is the number of nodes and q the number of arcs in the network (see [1], [2] for more details). Thus, the complexity of the algorithm described in [5] is $O((m+n)m^2 n^2 \log_2 n)$, i.e. $O(n^5 \log_2 n)$ when $m \approx n$, if the Edmonds-Karp algorithm is used and it is $O((m+n)^2 mn \log_2 n)$, i.e. $O(n^4 \log_2 n)$ when $m \approx n$, if the Dinic algorithm is used for maximum flow. To our knowledge, a number of other polynomial-time algorithms for various versions of the convex cost flow problem have been developed, including those of Minoux [3] and [4].

Exploiting the special structure of problem (P), below we shall develop a quite different algorithm for solving exactly (P) which has the following features:

1. it is a strongly polynomial-time algorithm; its running time is $O(m^2 n^2)$ and $O(n^4)$ when $m \approx n$. This complexity result is better than that of the network flow algorithm proposed in [5].
2. it is based on a numbering technique, commonly used in solving transportation problems, to improve feasible solutions, hence it does not make direct use of maximum flow algorithms;
3. it can easily be extended to more general objective function cases (e.g. the pointwise maximum of some increasing functions) or to the case where the a_{ij} values are arbitrary nonnegative integers.

2 PRELIMINARY RESULTS

As usual for convenience we agree that a matrix $x = \{x_{ij}\}$ whose entries satisfy (2) and (3) is called a feasible solution of (P), a feasible solution achieving the minimum of (1) is called an optimal solution of (P).

Let

$$a_i = \sum_{j=1}^{n} a_{ij},\ i = 1, 2, \ldots, m;\ b_j = \sum_{i=1}^{m} a_{ij},\ j = 1, 2, \ldots, n;\ p = \sum_{i=1}^{m} p_i > 0$$

(a_i denotes the number of subjects agreeable to student i, b_j represents the number of students willing to study subject j and p is the total number of subjects required for all the students).

It is proved in [5] that a necessary and sufficient condition for the existence of an optimal solution to (P) is that

$$a_i \geq p_i \text{ for all } i = 1, 2, \dots, m. \tag{4}$$

Condition (4) is very simple and easily checked. So, we assume that (P) satisfies this condition. It is also natural to suppose that $bj > 0$ for all $j = 1, 2, \dots, n$, because if $b_j = 0$ for some j then subject j must be deleted (i.e. there is no student who wants to study the subject).

For the sake of convenience, we associate with each feasible solution $x = \{x_{ij}\}$ of (P) a table consisting of m rows and n columns, in which each of the rows corresponds to a student and each of the columns corresponds to a subject. The cell at the intersection of row i and column j is denoted by (i, j). A feasible solution $x = \{x_{ij}\}$ of (P) will correspond to a table consisting of zeros and ones in its cells. A cell (i, j) is called black if $a_{ij} = 0$ ($x_{ij} = 0$ for all black cells (i, j), because student i is not willing to study subject j). The remaining cells will be divided into two classes: white cells if $x_{ij} = 0$ (student i is willing to study subject j, but he is not allocated for this subject) and blue cells if $x_{ij} = 1$ (student i is allocated for subject j).

Set

$$t_j^x = \sum_{i=1}^{m} x_{ij}, \ j = 1, 2, \dots, n, \text{ and } t^x = \max_{1 \leq j \leq n} t_j^x \tag{5}$$

(t_j^x is the number of students allocated for subject j and t^x is the objective function value of x).

For any feasible solution $x = \{x_{ij}\}$ of (P), according to (2) and (5), we have

$$\sum_{j=1}^{n} t_j^x = \sum_{j=1}^{n} \sum_{i=1}^{m} x_{ij} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} = \sum_{i=1}^{m} p_i = p \tag{6}$$

Column j is called full if $t_j^x = t^x$ and deficient if $t_j^x \leq t^x - 2$. It should be noted that the notions of blue cells, white cells, full columns and deficient columns are defined with respect to a given feasible solution.

The following proposition gives a simple criterion for an optimal solution of (P).

Proposition 1. *Let x be a feasible solution of (P). If x has no deficient column, i.e.*

$$t^x = t_{j_0}^x \textit{ for some } j_0 \in \{1, 2, \dots, n\} \textit{ and } \forall j \neq j_0, \tag{7}$$

then x is an optimal solution of (P).

Proof. From (6) and (7) it follows that

$$p = \sum_{j=1}^{n} t_j^x > n(t^x - 1) \tag{8}$$

Suppose the contrary that there exists a feasible solution y of (P) better than x, i.e. such that

$$t_j^y \geq t^x - 1 \text{ for all } j = 1, 2, \ldots, n \tag{9}$$

Combining (6) and (9) yields

$$p = \sum_{j=1}^{n} t_j^y \leq n(t^x - 1)$$

contrary to (8). Thus, x is an optimal solution. □

Consider now a feasible solution $x = \{x_{ij}\}$ of (P). Let C be an alternating sequence of white and blue cells (with respect to x) joining column j_0 to column j_k of the form

$$C = \{(i_0, j_0), (i_0, j_1), \ldots, (i_{k-1}, j_{k-1}), (i_{k-1}, j_k)\}, (k \geq 1), \tag{10}$$

where $(i_t, j_t), t = 0, 1, \ldots, k-1$, are white cells ($x_{i_t j_t} = 0$), white $(i_t, j_{t+1}), t = 0, 1, \ldots, k-1$, blue cells ($x_{i_t j_{t+1}} = 1$). Here all the row indices $i_0, \ldots, i_{k-1}$ are distinct and so are all the column indices $j_0, \ldots, j_k$. Let us introduce the following transformation of x in such a sequence.

Transformation A. Change every white cell in the sequence to blue one and every blue cell to white one. That is, we set

$$y_{i_t j_t} = 1, y_{i_t j_{t+1}} = 0, t = 0, 1, \ldots, k-1, y_{ij} = x_{ij}, (i, j) \notin C.$$

Since in each of the rows $i_t (t = 0, 1, \ldots, k-1)$ there are just one white cell and one blue cell of C, $y = \{y_{ij}\}$ satisfies (2) and (3), i.e. y is also a feasible solution of (P).

Proposition 2. *Let x be a feasible solution of (P). If there exists an alternating sequence of white and blue cells joining a deficient column to a full column, then x can be changed to a new feasible solution y of (P) which is better with respect to the objective function (1) or has smaller number of full columns than x.*

Proof. Let C be an alternating sequence of white and blue cells joining a deficient column, say j_0 to full column, say j_k. Applying Transformation A in C, we obtain a new feasible solution $y = \{y_{ij}\}$ of (P). Since in each of the columns $j_t (t = 1, 2, \ldots, k-1)$ there are just one white cell and one blue cell of C, we have

$$t_j^y = t_j^x, \forall j \neq j_0, j_k. \tag{11}$$

On the other hand, as column j_0 has only one cell of C (white cell (i_0, j_0)), we obtain

$$t_{j_0}^y = t_{j_0}^x + 1 \tag{12}$$

and as column j_k has only one cell of C (blue cell (i_{k-1}, j_k)), we get

$$t^y_{j_k} = t^x_{j_k} - 1 \tag{13}$$

As column j_0 is deficient, from (11) - (13) it follows that if j_k is a unique full column with respect to x then $t^y = t^x - 1$, i.e. y is better than the current solution x. In the opposite case, we have $t^y = t^x$, i.e. y is no worse than x, but y has at least one full column fewer than x (as j_k will not be a full column with respect to y). □

We have another optimality criterion.

Proposition 3. *Let x be a feasible solution of (P). If there is no alternating sequence of white and blue cells joining a deficient column to a full column, then x is an optimal solution of (P).*

Proof. Suppose the contrary that there is a feasible solution $y = \{y_{ij}\}$ of (P) better than $x = \{x_{ij}\}$,i.e. such that

$$t^y = \max_{1\le j\le n} t^y_j = \max_{1\le j\le n} t^x_j = t^x_{j_0} \text{ for some } j_0 \in \{1, 2, \ldots, n\}, \tag{14}$$

where t^x, t^y are defined by (5) with respect to x and y respectively. We show that this leads to a contradiction. Indeed, from (14) we have $t^y_{j_0} < t^x_{j_0}$. It follows from (5) that there exists one row i_0 such that $y_{i_0j_0} = 0,\ x_{i_0j_0} = 1$ (i.e. (i_0, j_0) is a blue cell with respect to x). Moreover, as both x and y satisfy (2) with $i = i_0$, there is one column $j_1 \neq j_0$ such that $y_{i_0j_1} = 1,\ x_{i_0j_1} = 0$ (i.e. (i_0, j_1) is a white cell with respect to x). If we still have $t^y_{j_1} \le t^x_{j_1}$, there exists $i_1 \neq i_0$ such that $y_{i_1j_1} = 0, x_{i_1j_1} = 1$ (i.e. (i_1, j_1) is a blue cell with respect to x), and also by (2) there must be one column $j_2 \neq j_1$ such that $y_{i_1j_2} = 1,\ x_{i_1j_2} = 0$ (i.e. (i_1, j_2) is a white cell with respect to x). If $j_2 \neq j_0$, we continue this process until either of the following cases occurs.

Case A. A column $j_r \neq \{j_0, \ldots, j_{r-1}\}$ with $t^x_{j_r} < t^y_{j_r}$ is reached. From (14) we have $t^x_{j_r} < t^y_{j_r} < t^x$, which implies that $t^x_{j_r} \le t^x - 2$ as $t^x_{j_r}, t^y_{j_r}$ and t^x are integers. So column j_r is deficient. Thus, in this case we obtain an alternating sequence of white and blue cells of the form

$$(i_{r-1}, j_r), (i_{r-1}, j_{r-1}), \ldots, (i_0, j_1), (i_0, j_0), (r \ge 1)$$

that joins the deficient column j_r to the full column j_0, contrary to the hypothesis of the proposition.

Case B. We obtain a cycle of cells of the form

$$C = \{(i_s, j_s), (i_s, j_{s+1}), \ldots, (i_t, j_t), (i_t, j_s), (i_s, j_s)\}, (0 \le s < t, t \ge 1)$$

or

$$C = \{(i_s, j_{s+1}), (i_{s+1}, j_{s+1}), \ldots, (i_{t-1}, j_t), (i_s, j_t), (i_s, j_{s+1})\}, (0 \le s < t, t \ge 2),$$

where $y_{i_u j_u} = 0,\ y_{i_{u-1} j_u} = 1 (s \le u \le t),\ y_{i_t j_s} = 1$ or $y_{i_s j_t} = 0$ Setting

$$\bar{y}_{i_u j_u} = 1,\ \bar{y}_{i_{u-1} j_u} = 0\ (s \le u \le t),\ \bar{y}_{i_t j_s} = 0$$

or

$$\bar{y}_{i_s j_t} = 1,\ \bar{y}_{ij} = y_{ij},\ \forall (i,j) \notin C,$$

we get a new feasible solution $\bar{y}$ with $t^{\bar{y}} = t^y$ (because $t_j^{\bar{y}} = \sum_{i=1}^{m} \bar{y}_{ij} = t_j^y$ for all $j = 1, \dots, n$).

If $\bar{y}$ still differs from x, the above process will be repeated with y replaced by $\bar{y}$. As the number of components by which $\bar{y}$ and x differ will decrease by at least four when Case B occurs, after a finite number of repetitions we must have $\hat{y} = x$ and, at the same time, $t^{\hat{y}} = t^y$, i.e. $t^x = t^{\hat{y}} = t^y$, which contradicts (14). □

One question now to be solved is how to find an alternating sequence of white and blue cells joining a deficient column to a full column, whenever such a sequence exists. To answer this question we consider the following procedure for rows and columns numbering.

Rows and Columns Numbering Procedure

First of all, we assign number 0 to each column j which is full ($t_j^x = t^x$). If column j is numbered, we assign number j to each row i which has not yet been numbered and has $x_{ij} = 1$ ((i,j) is a blue cell). Then, if row i is numbered, we assign number i to each column j which has not yet been numbered and has $a_{ij} - x_{ij} = 1$ (this is equivalent to $a_{ij} = 1, x_{ij} = 0$, i.e. (i,j) is a white cell) and so on. The above procedure must stop after at most $m + n$ times of numbering rows and columns.

Proposition 4. *An alternating sequence of white and blue cells joining a deficient column to a full column exists if and only if there is at least one deficient column that is numbered.*

Proof. Suppose there exists a sequence of form (10) joining deficient column j_0 to full column j_k. We claim that j_0 will be numbered using the above numbering procedure. Indeed, if column j_0 is not numbered row i_0 cannot be numbered either, as (i_0, j_0) is a white cell. Then j_1 cannot be numbered either as (i_0, j_1) is a blue cell, and so on. In the end, j_k cannot be numbered, contrary to the fact that full column j_k was first assigned number 0.

We now turn to the froof of suffciency. Suppose that a deficient column, say j_0, is assigned number i_0 ((j_0, j_0) is a white cell) and row i_0 is assigned number $j_1 \neq j_0$, ((i_0, j_1) is a blue cell). Let column j_1 be assigned a number not equal to 0, for instance, $i_1 \neq i_0$ ((i_1, j_1) is a white cell), and row i_1 be assigned number $j_2 \neq j_0, j_1$ ((i_1, j_2) is a blue cell). If column j_2 is assigned a number not equal to 0, we continue searching. As the number of columns is finite (equal to

n), eventually we must determine a column $j_k \neq j_t, t = 0, 1, \ldots, k-1$, assigned a 0, i.e. j_k is a full column, and the required sequence is

$$C = \{(i_0, j_0), (i_0, j_1), \ldots, (i_{k-1}, j_{k-1}), (i_{k-1}, j_k)\}, (k \geq 1),$$

where $(i_t, j_t), t = 0, 1, \ldots, k-1$, are white cells, while $(i_t, j_{t+1}), t = 0, 1, \ldots, k-1$, are blue cells. □

3 THE STRONGLY POLYNOMIAL - TIME ALGORITHM FOR (P)

From the above results we are now in a position to develop an algorithm for solving (P).

Numbering Algorithm for (P)

Step 0: Create a table consisting of m rows and n columns. Each row corresponds to a student and each column corresponds to a subject. Make a cell (i, j) black if $a_{ij} = 0$ (black cells will be not changed through the course of solving the problem).

Step 1: Initialization. For each row i, from 1 to m, we write 1 in the non black cells of the row from left to right until having a total of p_i ones, then we write 0 in the remaining cells of the row. As a result, we obtain an initial feasible solution $x^1 = \{x^1_{ij}\}$ of (P). The algorithm may also be started with any feasible solution of (P). Set $k = 1$ and go to Step 2.

Step 2: Test for optimality. For the obtained feasible solution x^k, we agree that the cells with 1 are called blue cells, and the non black cells with 0 are called white cells. Compute

$$t^k_j \equiv t^{x^k}_j = \sum_{i=1}^{m} x^k_{ij},\ j = 1, 2, \ldots, n,\ \text{and}\ t^k \equiv t^{x^k} = \max_{1 \leq j \leq n} t^k_j$$

Column j is said to be a full column if $t^k_j = t^k$ called a *deficient column* if $t^k_j \leq t^k - 2$ If no deficient column exists then x^k is an optimal solution of (P) (by Proposition 1). Otherwise, perform the numbering of rows and columns as described in Section 2. If there is no deficient column that is numbered then x^k is also optimal (by Proposition 3 and 4). In the opposite case, we must have a sequence C of form (10) that consists of alternating white and blue cells and joins a deficient column j_0 to a full column j_k (by Proposition 4). Go to Step 3.

Step 3: Solution improvement. Applying Transformation A in the sequence C obtained in Step 2, we get a new feasible solution y which is better ($t^y < t^k$) or has a fewer number of full columns than x^k (by Proposition 2). Set $x^{k+1} = y$ and $k \leftarrow k + 1$, then return to Step 2.

Proposition 5. *The above algorithm terminates after a finite number of steps.*

Proof. After each improvement in Step 3, either a better feasible solution or a solution with a fewer number of full columns than the previous one is obtained. Since the objective function (1) of the problem can take on only a finite number of positive integer values and since the number of columns in the problem is also finite (equal to n), the number of steps cannot increase indefinitely. □

Complexity of the Numbering Algorithm

We now analyze the complexity of the Numbering Algorithm. In order to bound the running time of the algorithm, we evaluate the number of arithmetic operations needed in each step of the algorithm in the worst case.

Step 1. An initial feasible solution and the corresponding values t_j^1 ($j = 1, 2, \ldots, n$) can be computed in $O(m \times n)$ arithmetic operations.

Step 2. As shown in (11) - (13), the time needed to update t_j^k ($j = 1, \ldots, n$) and t^k is at most $O(m + n)$. Determining the full columns and the deficient columns requires $O(n)$ arithmetic operations. The numbering of rows and columns can be performed in $O(m \times n)$ arithmetic operations. The operation of searching for an alternating sequence C of white and blue cells joining a deficient column to a full column is bounded by $O(m+n)$ arithmetic operations (using numbers assigned to the rows and columns as shown in the proof of Proposition 4). Therefore, Step 2 requires $O(m \times n)$ arithmetic operations in the worst case.

Step 3. The solution improvement in a sequence obtained in Step 2 is dominated by $O(m+n)$ arithmetic operations, because there are at most $(m+n)$ cells in such a sequence.

Step 2 and 3 are repeated several times. After each repetition either the objective value or the number of full columns is reduced by one unit. Since the objective function (1) can take on at most $\max\limits_{1\leq j\leq n} b_j \leq m$ integer values and there are n columns, the number of repetitions is bounded by $O(m \times n)$. Consequently, the algorithm requires $O((m\times n)(m\times n))$ or $O(m^2n^2)$ arithmetic operations. When $m \approx n$ the running time of the algorithm is $O(n^4)$. Hence, we have established the following result.

Proposition 6. *The Numbering Algorithm solves (P) in $O(m^2n^2)$ arithmetic operations.*

Remark 1. *This complexity result is better than the complexity of the network flow algorithm described in [5] for the same problem (P), which is $O((m + n)^2 mn \log_2 n)$ if the Dinic algorithm is used and which is $O((m+n)m^2n^2 \log_2 n)$ if the Edmonds-Karp algorithm is used for maximum flow.*

Illustrative example. Solve problem (P) whose data are as follows: $m =$

$5, n = 5, p_1 = 1, p_2 = 2, p_3 = 1, p_4 = 2, p_5 = 1$ and

$$A = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 \end{pmatrix}$$

Summing up the elements of A in each row and each column yields $a_1 = 2, a_2 = 3, a_3 = 2, a_4 = 3, a_5 = 2, b_1 = 2, b_2 = 3, b_3 = 2, b_4 = 3, b_5 = 2$ and $p = 7$.

At the completion of Step 1 of the Numbering Algorithm we obtain an initial feasible solution of (P).

$$x^1 = \begin{array}{cc} \begin{pmatrix} X & 1 & X & 0 & X \\ 1 & X & 1 & X & 0 \\ X & 1 & X & 0 & X \\ 1 & X & 1 & X & 0 \\ X & 1 & X & 0 & X \end{pmatrix} & \begin{matrix} 2 \\ \\ 2 \\ \\ 2 \end{matrix} \\ \begin{matrix} & 0 & & 1 & \end{matrix} & \end{array}$$

(black cells are marked by X, the last column indicates the numbers assigned to the rows and the last row the numbers assigned to the columns).

Step 2. Summing up the elements in each column of x^1 yields

$$t_1^1 = 2,\ t_2^1 = 3,\ t_3^1 = 2,\ t_4^1 = t_5^1 = 0 \text{ and } t^1 = 3.$$

Column 2 is full. Columns 4, 5 are deficient. Column 2 is first numbered with a 0. We search column 2 in x^1 for a 1 (blue cell) and find it in rows 1, 3, 5, so these rows are numbered with a 2 (subscript of column 2). We now search numbered row 1 for a 0 (white cell) and find it in column 4 (not yet numbered), so column 4 is numbered with a 1 (subscript of row 1). At this point, deficient column 4 is numbered with a 1 (row 1), row 1 is numbered with a 2 (column 2). Column 2 is full. Thus, we obtain the sequence of cells: (1,4) - (1,2) joining deficient column 4 to full column 2.

Step 3. Changing x^1 in the sequence of cells just determined in Step 2 gives a new feasible solution

$$\begin{array}{cc} \begin{pmatrix} X & 0 & X & 1 & X \\ 1 & X & 1 & X & 0 \\ X & 1 & X & 0 & X \\ 1 & X & 1 & X & 0 \\ X & 1 & X & 0 & X \end{pmatrix} & \begin{matrix} \\ 1 \\ 2 \\ 1 \\ 2 \end{matrix} \\ \begin{matrix} 0 & 0 & 0 & & 2 \end{matrix} & \end{array}$$

First return to Step 2. Summing up the elements in each column of x^2 yields

$$t_1^2 = t_2^2 = t_3^2 = 2,\ t_4^2 = 1,\ t_5^2 = 0 \text{ and } t^2 = 2$$

Columns 1, 2, 3 are full. Column 5 is deficient. The numbering procedure now gives the sequence of cells: (2,5) - (2,1) joining deficient column 5 to full column 1.

Step 3. Changing x^2 in this new sequence of cells gives a new feasible solution

$$x^3 = \begin{pmatrix} X & 0 & X & 1 & X \\ 0 & X & 1 & X & 1 \\ X & 1 & X & 0 & X \\ 1 & X & 1 & X & 0 \\ X & 1 & X & 0 & X \end{pmatrix}$$

Second return to Step 2. Summing up the elements in each column of x^3 yields

$$t_1^3 = 1,\ t_2^3 = t_3^3 = 2,\ t_4^3 = t_5^3 = 1 \text{ and } t^3 = 2$$

Now columns 2 and 3 are full, but there is no deficient column, so x^3 (with x replaced by 0) is an optimal solution of (P). The optimal function value is $t^* = t^3 = 2$.

Remark 2. *It can easily be verified that one of the optimal continuous solutions of (P) without integer restriction is*

$$x_{opt} = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{2}{3} & 0 & \frac{2}{3} & 0 & \frac{2}{3} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{2}{3} & 0 & \frac{2}{3} & 0 & \frac{2}{3} \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 \end{pmatrix}$$

with the optimal value $f_{opt} = \frac{3}{2} < t^* = 2$. Our example therefore shows an instance featuring a strict integrality gap.

Remark 3. *The proposed algorithm can also be applied to the case where the a_{ij} values are arbitrary nonnegative integers. But one question still unanswered is whether the algorithm remains strongly polynomial in this case. To answer this question it requires further modifications and investigations.*

Problem (P) with more general objective function (e.g. the pointwise maximum of some increasing functions) will be studied in a subsequent paper.

ACKNOWLEDGMENT

The authors would like to thank Prof. Tran Vu Thieu for his helpful commnets and suggestions.

References

[1] R.K. Ahuja, T.L . Magnanti and J.B. Orlin. Network Flows: *Theory, Algorithms and Applications.* Prentice Hall, N.J., 1993.

[2] S.Khuller, Y.J. Sussman, W. Gasarch. Advanced Algorithms. *Lectures* CMSC 858K, Jan 30 - May - 13, 1997.

[3] M.Minoux. *A polynomial algorithm for minimum quadratic cost flow problems.* European J. Oper. Res., 18 (1984), 377- 387.

[4] M. Minoux. *Solving integer minimum cost flow with separable convex cost objective polynomially.* Math. Proc. Study 26 (1986), 237 - 239.

[5] N.D. Nghia and V.V.T. Dung. *A polynomial-time algorithm for solving a class of discrete optimization problems*, Journal of Computer Science and Cybernetics, Vol. 15, 1(1999), 8 - 13 (in Vietnamese).

[6] N.H. Xuong. *Mathematiques Discretes et Informatiques.* Masson, Paris 1992.