การแก้ไขปัญหาขนส่งและการจัดงานขนาดใหญ่ด้วยเทคนิคเพิ่มคอลัมน์

วรุตม์ บุญภักดี *[1] พีรยุทธ์ ชาญเศรษฐิกุล[2]
ภาควิชาวิศวกรรมอุตสาหการ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยเกษตรศาสตร์
กรุงเทพฯ 10900, ประเทศไทย

บทคัดย่อ

การศึกษานี้มีวัตถุประสงค์เพื่อแก้ปัญหาขนส่งและการจัดงานที่มีตัวแปรตัดสินใจจำนวนมากเกินไปเมื่อใช้ซอฟท์แวร์มาตรฐานสำหรับกำหนดการเชิงเส้น เพื่อก้าวข้ามข้อจำกัดดังกล่าว งานวิจัยนี้นำเสนอวิธีการเพิ่มคอลัมน์เพื่อแก้ปัญหาทั้งสองที่มีขนาดใหญ่ โดยพัฒนาให้อยู่ในรูปของโปรแกรมแมทแล็ป รุ่นปี 2010 และทำการคำนวณเชิงทดลองเปรียบเทียบกับคำสั่งฯ จากกล่องเครื่องมือ “linprog” ซึ่งผลการใช้งานโปรแกรม พบว่าขนาดใหญ่ที่สุดในการแก้ปัญหาขนส่งและการจัดงานด้วยวิธีการเพิ่มคอลัมน์ คือ 16,333 ลูกค้า x 100 โรงงานและ 12,900 งาน x 12,900 พนักงาน ในขณะที่ขีดจำกัดของคำสั่ง “linprog” คือ 11,063 ลูกค้า x 100 โรงงานและ 7,400งาน x 7,400 พนักงาน ภายใต้ฮาร์ด์แวร์ทดลองตัวเดียวกัน ส่วนเวลาเฉลี่ยที่ใช้ในการแก้ปัญหาฯด้วยวิธีการเพิ่มคอลัมน์จะมากกว่าในปัญหาขนาดเล็ก แต่ความแตกต่างระหว่างของเวลาที่ใช้ในการแก้ปัญหาด้วยวิธีทั้งสองมีค่าลดลงอย่างมีนัยสำคัญ เมื่อจำนวนตัวแปรเพิ่มขึ้น ผลการทดลองครั้งนี้ยังพบว่าเวลาในการแก้ปัญหาฯด้วยวิธีการเพิ่มคอลัมน์มีความเกี่ยวข้องโดยตรงกับจำนวนคอลัมน์ที่ถูกสร้างขึ้น และยังพบด้วยว่าจำนวนเงื่อนไข/ข้อจำกัดที่มีขนาดใหญ่ของปัญหาขนส่งมีอิทธิพลต่อเวลาในการแก้ปัญหาฯมากกว่าจำนวนตัวแปรที่มีจำนวนมากเกินไป

คำสำคัญ : ปัญหาการวางแผนขนส่ง, ปัญหาการจัดงาน, การเพิ่มคอลัมน์, วิธีซิมเพล็กซ์

*Corresponding author. E-mail: Warutboon@yahoo.com
[1]Doctoral Student in the Faculty of Engineering, Kasetsart University
[2]Associate Professor in the Faculty of Engineering, Kasetsart University

Solving the Linear Programming Model of Large-Scale Transportation and Assignment Problems using the Column Generation Technique

Warut Boonphakdee[*1] **Peerayuth Charnsethikul**[2]
Department of Industrial Engineering, Kasetsart University
Bangkok 10900, Thailand

Abstract

The aim of this study was to solve the transportation problems (TPs) and assignment problems (APs) involved with too many decision variables when using a regular state-of-the-art linear programming (LP) software. To overcome its limitations, a column generation method was developed and used to solve both problems with large-scale sizes. The method was coded using MATLAB 2010b and was used to compute experimentally as compared to the use of the regular LP-solving toolbox "linprog.". The results of this experiment revealed that the largest sizes for the column generation method to solve both TP and AP were 16,333 customers x 100 plants/warehouses and 12,900 tasks x 12,900 assignees, respectively, while the largest sizes for the "linprog" to solve both were limited to 11,062 customers x 100 plants and 7,400 tasks x 7,400 assignees respectively using the same hardware. The average computational time of the column generation method was greater in small-size problems; nevertheless, the difference between the computing times consumed by both methods reduced significantly when the number of variables was increased. This experiment revealed that the computation time of the TP was related directly to the number of new columns generated. Also, the large number of constraints of the TP model for the column generation technique can be more influential on computation time performances than the large number of variables.

Keywords: Transportation Planning Problem, Assignment Problem, Column Generation, Simplex Method

*Corresponding author. E-mail: Warutboon@yahoo.com
[1]Doctoral Student in the Faculty of Engineering, Kasetsart University
[2]Associate Professor in the Faculty of Engineering, Kasetsart University

1. Introduction

The decision variables in network flow problems, i.e. minimum cost flow, maximum flow, shortest path, the transportation (TP), and assignment problems (AP) can be solved as a special method. TP can be directly solved by the stepping stone and the modified distribution method and AP can be also solved by the Hungarian algorithm. Normally, the classical simplex method still enables to solve the both problems. However, this method involves the large number of variable, whereas the column generation technique enhances efficiently to solve both problems with creating less variables. Applications of the transportation and assignment problems tend to require a very large number of related constraints and variables, so solving these may require a vigorous computation effort [13]. The reviewed paper [1] indicated that the first successful method of G.B. Dantzig [15] could solve both the transportation and transshipment problems efficiently. This method uses the primal simplex method to manipulate the network structure that is applied to transportation problems and is sometimes referred to as the Row-Column Sum method or the modified distribution (MODI) method. The methods were implemented and solved TPs with many decision variables under a special LP structure.

In case there are too many variables, the structure can be handled and solved using Bender's partitioning procedure. Such a block-diagonal structure can also be solved using Rosen's primal partitioning algorithm and Dantzig-Wolfe decomposition algorithm [2]. However, an efficient algorithm for solving transportation problems can improve over the existing algorithms of the "primal-dual" type. This algorithm can be adapted to an n-by-n assignment problem. Furthermore, an auxiliary technique of simplifying the original network by means of "reduction" and "induction" was also introduced as a useful tool to treat large-scale problems [3]. The implementation of the primal transportation model was solved under a variety of large fully-dense, randomly-generated transportation and assignment problems ranging in sizes up to m = n = 3,000 [4]. G.B. Danzig considers the technique to reduce the computational requirement of large systems from two points of view: (1) decrease the number of iterations that the variants of the simplex method have been proposed to replace the usual Phase I of the simplex method, and (2) find a compact from for the inverse and/or by taking advantage of any special structure of the system of equations [14]. For solving the AP, a new algorithm is proposed which can reduce the expected computation time needed to find an exact solution by limiting the search space as much as possible within the scope in which no optimal solution is missed [5].

Nowadays, spreadsheet programs can efficiently solve these problems; however these programs cannot solve many variables due to the limitation of the edition attached. Using Excel 2010 solves the LP with a large number of variables, while the attached LP simplex engine of Excel's solver is limited merely up to 200 variables and 100 constraints. Also, some programs provide a source code object to solve large-scale linear programs; i.e. the "linprog" command in MATLAB software can solve large-scale LP with the condition that the total number of coefficients in matrix form must be restricted to less than 10^8 elements. MATLAB's optimization toolbox is widely applied to solve many type of the linear programming model; e.g. the primal-dual interior-point LP [17] the augmented langrangian method for the large-scale LP [18] and the binary integer programming [19]. The solving method for an LP model with too many variables can be presented as follows. The large-scale TP is considered to be divided to two sub-problems: one using an aggregation that consolidates several neighboring sources (destinations) and other using a process of partial disaggregation [6].

Column-generation approaches are among the efficient approaches to solving LP with too many constraints/variables with an early application found in a cutting stock problem [7]. Implementation of the row and column generation technique can solve the one-dimensional cutting stock problem efficiently and effectively [8]. A hybrid-column generation and constraint-programming solution approach can be used to quickly produce solutions for operations management and also to produce close-to-optimal solutions for long- and mid-term planning scenarios [9]. A column-generation approach code can be developed by systematically programming to solve the corresponding linear program and to identify the new column leading to a better solution; the process is repeated until the optimal solution is achieved [10]. With the railway crew scheduling problem [11], the proposed method can reduce the computation time by improving the convergence of column generation with the new dual inequalities. This aim of this paper was to propose a column-generation approach to solve large-scale transportation and assignment problems when hardware limitations of memory requirements were encountered.

2. Methodology

2.1 Transportation Problem

The transportation problem is a linear network optimization with an application for industrial companies having several plants, warehouses, sale offices, and distribution outlets [12]. In the classical transportation model, there are *m* supply points with items available to be shipped to *n* demand points with s_i = Supply amount at source i and d_j = Demand amount at destination j and c_{ij} = Unit transportation cost from source i to destination j for $i = 1,2,..,m$ and $j=1,2,..,n$.

The general transportation problem is a special type of linear programming problem [13]. Let Z be the distribution cost and $x_{ij}\,(i=1,2,\dots,m;\ j=1,2,\dots,n)$ be the number of units to be distributed from source i to destination j. The linear programming formulation of this problem is as follows.

$$\text{Minimize} \qquad Z=\sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$$

$$\begin{aligned} &\text{subject to} && && (1)\\ && \sum_{j=1}^{n} x_{ij} &= s_i && \text{for } i=1,2,\dots,m\\ && \sum_{i=1}^{m} x_{ij} &= d_i && \text{for } j=1,2,\dots,n\\ &\text{and} & x_{ij} &\ge 0, && \text{for all } i \text{ and } j. \end{aligned}$$

The constraint coefficients of the linear programming model have a special structure as shown in Fig. 1. Any linear programming problem that joins this special formulation does not correspond to transportation as illustrated in the assignment problem. For transportation problems, where every s_i and d_j has integer values, all of the basic variables/allocations (including an optimal one) also have integer values.

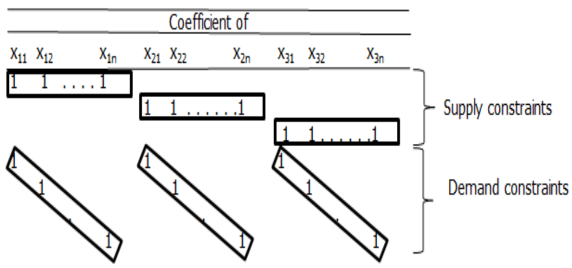


Fig.1 Constraint coefficients of the transportation problem

Additionally, the right-hand side consists of supply (s_i) and demand (b_j) respectively.

2.2 The assignment problem

The AP is a special type of TP-based linear programming problem where assignees are assigned to perform tasks [13]. For example, assignees might be employees that need to be given work assignments. Assigning people to jobs is a common application of the assignment problem. However, the assignees need not be people; they also can be machines or vehicles, or plants, or even time slots to be assigned tasks. The formulation of assignment problems satisfies the following assumptions.

1. The number of assignees and the number of tasks are the same (this number is denoted by *n*).

2. Each assignee is to be assigned to exactly one task.

3. Each task is to be performed by exactly one assignee.

4. There is a cost c_{ij} associated with the assignee *i* (*i* =1, 2, ..., *n*) performing task *j* (*j* =1, 2, ..., *n*).

5. The objective is to determine how all *n* assignments should be paired to order to minimize the total cost.

The mathematical model for the assignment problem uses a binary variable as follows:

$$x_{ij} = \begin{cases} 1 & \text{if assignee } i \text{ performs task } j \\ 0 & \text{if not,} \end{cases} \quad \text{for } i=1,2,...,n \text{ and } j=1,2,...,n \qquad (2)$$

The binary variables are important for representing yes/no decisions. Let Z be the total cost; the assignment problem in a linear programming formulation is then as follows.

$$\text{Minimize} \quad Z = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$

subject to

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \text{for } i=1,2,...,n \qquad (3)$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \text{for } j=1,2,...n$$

$$\text{and} \quad x_{ij} \geq 0, \qquad \text{for all } i \text{ and } j$$

$$(x_{ij} \text{ binary}, \qquad \text{for all } i \text{ and } j)$$

To compare this model (without the binary restriction) with the TP model, both are similar in their structure. In fact, the assignment problem is just a special type of TP, where the sources now are assignees and the destination are tasks and where the number of sources *m* = the number of destinations *n,* and every supply s_i =1 and every demand d_j =1 for all possible *i* and *j*.

The constraint coefficients of its corresponding LP model have the special structure shown in Fig. 2. For the AP, where every s_i and d_j has an integer value equal to one, similar to the TP, all basic variables (assignments) are binary automatically. Therefore, in this case, the binary restriction is redundant.

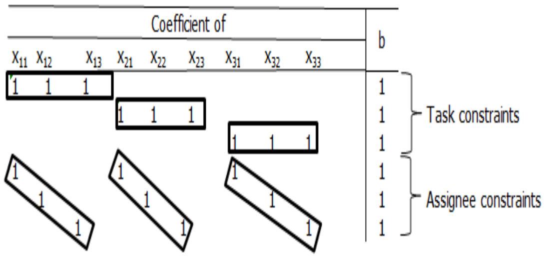


Fig.2 Constraint coefficients with right-hand sides for the AP model

2.3. The primal and dual relationship

Taha [16] has stated that for the special structure of the LP representing the transportation model, the associated dual problem can be written as

$$\text{Maximize} \quad z = \sum_{i=1}^{m} s_i u_i + \sum_{j=1}^{n} d_j v_j$$

$$\text{subject to} \qquad (4)$$

$$u_i + v_j \leq c_{ij} \qquad \text{all } i \text{ and } j$$

$$u_i \text{ and } v_j \text{ unrestriced}$$

; where

u_i = the Dual variable of the constraint associated (shadow price) with source i

v_j = Dual variable of the constraint associated (shadow price) with destination j .

The objective-function coefficients or reduced cost of the variable x_{ij} are theoretically equal to the difference between the right-and the left-hand sides of the corresponding dual constraint, that is, $c_{ij}^{*} = c_{ij} - u_i - v_j$ [14]. However, this quantity must be equal to zero for each basic variable.

2.4. Column generation

Column generation [8] extends the technique introduced in the decomposition algorithm that uses the simplex method and generates the constraint coefficient data associated with the additional variable only as needed in order to improve the objective function *Z*. The column generation approach reaches for an optimal solution of a great number of linear programs where not all of the columns have been carefully considered, i.e., variables in the matrix of constraint. The original problem includes all of the problem characteristics and again is called a master problem (MP), whereas a subset of a MP column is called the restricted master problem (RMP) or the initial basic feasible (BF) solution. Only a few subsets of the MP column (basic variable) will be in the optimal solution and all other columns (non-basic variable) can be omitted. In a minimization problem, the column (basic variable) that has negative reduced cost must be selected to add as an attaching column with the matrix of the RMP. This procedure terminates when the reduced cost of all columns must be positive; consequently, solving the RMP by adding the last column can lead to an optimal solution. A master problem is given in equation (1) and (3), where *n* is very enormously large, and a direct solution solved by a regular state-of-the-art LP software is not feasible due to out-of-memory requirements. Then, the restricted master problem can be truncated as follows:

The constraint of the RMP can be constructed using a general procedure [13]; i.e., the northwest corner rule (NWC), the least cost method (LCM), Vogel's approximation method (VAM), and Russell's approximation method (RAM). The constructing of the RMP or an initial basic feasible (BF) solution is an important step in solving the TP and AP. This paper alters the Northwest corner rule to construct the initial matrix of the RMP. The advantage of this rule is its simplicity, while its disadvantage is due to non-cost considerations during allocations. Next, the column-generation method for solving the linear programming model of the transportation and assignment problems with too many variables is systematically described in the following steps.

First, start with the RMP generated by the northwest corner rule resulting in *J = m+n-1* variables and *m+n* constraints. It is an initial basic feasible solution, which can be improved later using the optimization tool.

The initial BF solution or RMP of the assignment problem can be constructed with *J = 2n-1* variables and *2n* constraints. The coefficients of constraint are allocated by arranging the coefficient on a diagonal line from the northwest corner to southeast corner of the RMP. Therefore, this RMP does not concern the target cost as well as the RMP of the TP does.

After constructing the RMP, the classical simplex method solves this linear program and gives the optimal shadow price $(u_1, u_2, ..., u_m; v_1, v_2, ..., v_n)$ for all constraints. Other variables with $x_{J+1} = x_{J+2} = ... = x_n = 0$ are non-basic solutions. The optimality test of this solution is to compute c^{*}_{ij}. If c^{*}_{ij} for all *i and j* reduced costs of any variable in the MP is less than zero, the column-generation approach accounts for adding a variable to the RMP. Then, the simplex method solves this new RMP repeatedly until the reduced costs of all of the variables in the MP are at least zero. The last RMP is the optimal solution for the problems. An efficient column-generation approach depends on the initial basic solution; that is, it should be closely optimal [10]. In summary, the proposed column-generation approach can be illustrated in the following flowchart.

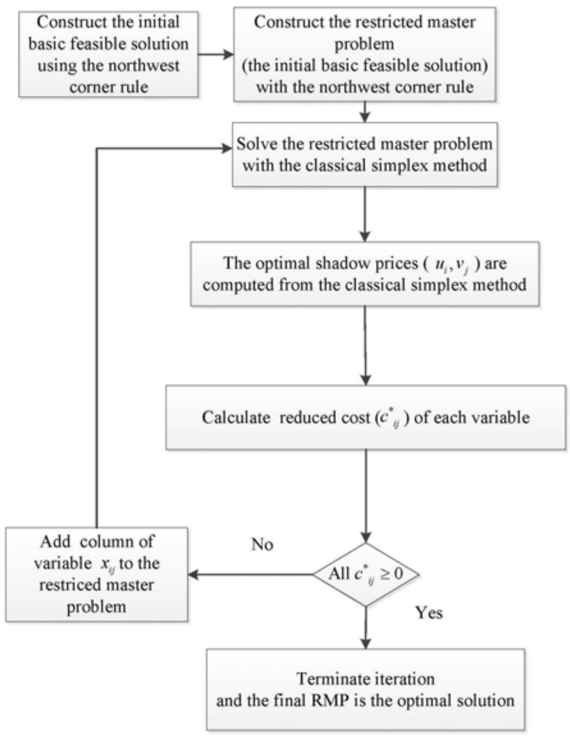


Fig. 3 Flow chart of the column-generation approach

2.5 A numerical example

The MATLAB 2010b was used to write the source code. This paper employs two approaches for the TP. First, approach T1 constructs the initial BF solution using the northwest corner rule and the column-generation technique to solve the corresponding LP sequentially; and the second approach T2 constructs the full constraint coefficients of TP (Fig.1) and solves the relevant LP using the simplex method. For the AP, approach A1 constructs the initial BF solution from a diagonal line matrix and uses the column-generation technique to solve the LP, and approach A2 constructs the full constraint coefficient of the AP (Fig.2) and solves the LP using the simplex method. This framework of experiment is illustrated in Fig. 4.

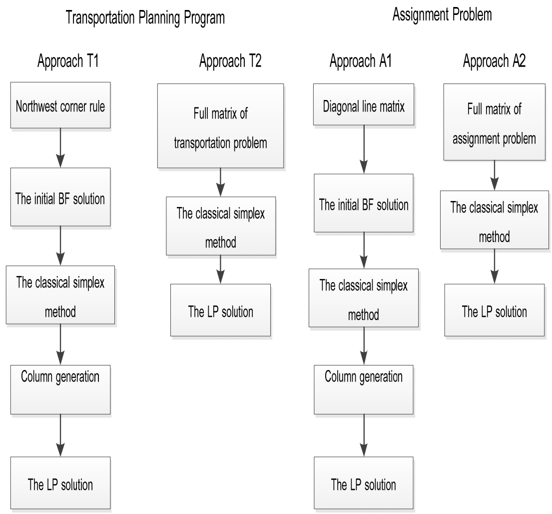
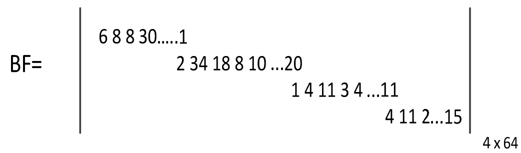
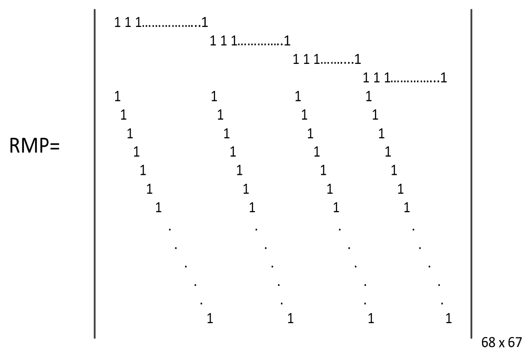


Fig.4 The framework of Approach T1, T2, and A1, A2

Each approach in Fig. 4 was programmed and executed using each source code of the MATLAB software. How to solve TP using the column generation technique that can illustrate a simple test problem from OR library [20]. The ran4x64 test problem contained 4 plants [112 109 131 318] , 64 customers [6 8 8 ...15] and 256 cost per unit [1 1 4 4 7...9]. This test problem was solved as follows:

1. Construct the basic feasible solution using the northwest corner rule and the restricted MP as follow:





2. Solving the RMP using the classical simplex method.

[RMP] [x] = [b]

$$\underbrace{\left|\begin{array}{cccc}
111\ldots1 & & & \\
 & 111\ldots1 & & \\
 & & 111\ldots1 & \\
 & & & 111\ldots1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\vdots & \vdots & \vdots & \vdots \\
1 & 1 & 1 & 1
\end{array}\right|}_{68 \times 67}
\underbrace{\left|\begin{array}{c} X1 \\ X2 \\ X3 \\ \vdots \\ x68 \end{array}\right|}_{68 \times 1}
=
\underbrace{\left|\begin{array}{c} 112 \\ 109 \\ 131 \\ 318 \\ 6 \\ 8 \\ 8 \\ \vdots \\ 15 \end{array}\right|}_{68 \times 1}$$

3. The shadow prices are calculated from command “linprog”in step 2 as follow:

1st Iteration: The shadow prices are

$$u_i = -[-1 -2\ 3\ 2] = [1\ 2\ 3\ 2] \qquad \text{for } i = 1,2,3,4$$
$$v_j = -[0\ 0\ \text{-}3\ \text{-}3 \ldots \text{-}11] = [0\ 0\ 3\ 3 \ldots 11] \text{for } j = 1,2 \ldots 64$$

4. Calculate the reduced cost (c^*_{ij}) of each variable.

1st Iteration:

$$c^*_{i,j} = c_{i,j} - u_i - v_j$$
$$c^*_{1,1} = c_{1,1} - u_1 - v_1 = 1 - 1 - 0 = 0$$
$$c^*_{1,2} = c_{1,2} - u_1 - v_2 = 1 - 1 - 0 = 0$$
$$c^*_{1,3} = c_{1,3} - u_1 - v_3 = 4 - 1 - 3 = 0$$
$$. \quad . \quad . \quad . \quad . \quad . \quad .$$
$$c^*_{1,13} = c_{1,13} - u_1 - v_{13} = 7 - 1 - 8 = -2$$
$$. \quad . \quad . \quad . \quad . \quad . \quad .$$
$$c^*_{2,23} = c_{2,23} - u_2 - v_{23} = 2 - 2 - 13 = -13$$
$$. \quad . \quad . \quad . \quad . \quad . \quad .$$
$$c^*_{4,64} = c_{4,64} - u_4 - v_{64} = 9 - (-2) - 11 = 0$$

So $c^*_{2,64}$ = -13 is a minimal reduced cost that constraint supply is i = 2 and constraint demand is j = 23. There is any reduced cost from 1st iteration less than zero.

5. Add the new supply constraint i=2 and the new demand constraint i=27 on the last column of the restricted MP [68 x 67] as follows:

$$\text{New RMP=} \underbrace{\left|\begin{array}{cccc}
111\ldots1 & & & \\
 & 111\ldots1 & & \\
 & & 111\ldots1 & \\
 & & & 111\ldots1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 \\
\vdots & \vdots & \vdots & \vdots \\
1 & 1 & 1 & 1
\end{array}\right|}_{68 \times 67}
+
\underbrace{\left|\begin{array}{c} . \\ 1 \\ \vdots \\ 1 \\ \vdots \end{array}\right|}_{68 \times 1}
\begin{array}{l} \\ \text{row2} \\ \\ \text{row27} \\ \\ \end{array}$$

6. Turn to step 2 and calculate the new reduced cost. If any reduced costs are negative the iteration will proceed later.

7. Finally, the 46th iterations had calculated until all the reduced costs were nonnegative. The last RMP [68 x 112] was solved LP by the classical simplex method. The objective function of this problem was involved 1806.

This study interests the largest scale size and the computation time. The proposed programs used random numbers to generate the data for the number of customers, plants/warehouses and cost per unit. All developed programs were performed using a Toshiba “Satellite” Notebook with processor: Intel ® core ™i5-2430M CPU@2.40GHz 2.74 GB usable RAM and the operating system was Windows 7.

3. Results and Discussion

3.1 The largest scale sizes

When the scale sizes of Approach T1 and A1 were expanded to the largest scale sizes, the scale size was difficult to achieve because of the long period of operation. The largest sizes of Approach T1 and A1 were 16,333 customers x 100 plants and 11,063 customers x 100 plants. Approach T1 could solve a larger number of customers than Approach T2 at about 47.64%, while the largest scale sizes of Approach A1 and A2 was able to reach 12,900 tasks x 12,900 assignees and 7,400 tasks and 7,400 assignees. Approach A1 could solve a greater number of tasks/assignees than Approach A2 at about 74.32%, as shown in Table 1.

Table 1 The largest scale sizes

Problem	Experiments	The largest scale sizes	% Expansion
TP	Approach T1	16,333x100	47.64%
	Approach T2	11,063x12,900	
AP	Approach A1	12,900X12,900	74.32%
	Approach A2	7,400X7,400	

3.2 The computation time

This experiment was carried out on a commercial notebook for solving the LP solution, which could be used simply and inexpensively. If these were performed on a high-performance server, the computation time would have certainly been lower. The amount of computation time is a significant criterion for the operation of software; accordingly, a large computation time is unsatisfactory for implementation. In Table 2, Approach T1 was difficult to search the computation time due to long time consumed for column generation, while Approach T2 could reach easily a smaller computation time due to the fewer iterations of the simplex method.

Table 2 The scale sizes, the computation time, and new columns and iterations of Approach T1 and T2

Items	TP	No. of customers x no. of plants/warehouses				
		1x 100	10x 100	100x 100	1000 x100	10000 x100
The avg. computation time (sec.)	T1	66	1120	10,254	>43,200	>43,200
	T2	0.45	26	3,567	>43,200	>43,200
Ratio of the avg. computation time T1 and T2	T1 & T2	147	43	3	-	-
No. of new column	T1	317	1,114	1,648	-	-
No. of iterations	T1	305	1,246	1,213	-	-
No. of experiment		20	10	10	2	2

According to Table 2, the number of customers increases and the computation time of Approach T1 and T2 also rises sharply until the number of customer is up to 1,000.These average computation times were over 43,200 sec.; however, the solutions of all conditions can be available if the program displays “busy” at the bottom of the screen. While the ratio between the computation time of T1 and T2 was reduced sharply, it indicates that the difference in the computation time of T1 and T2 diminished closely. This ratio was reduced from 147 to 3 times as shown in Fig. 5.

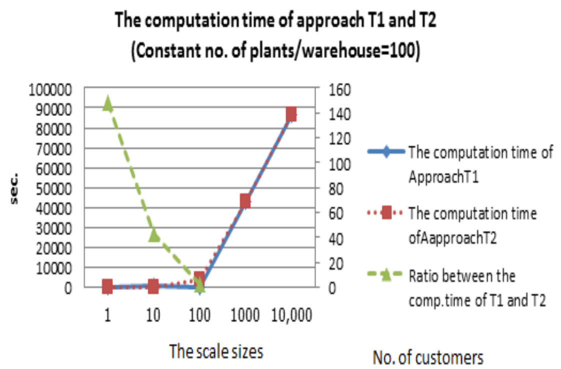


Fig.5 The average computation time of Approach T1, T2

In Fig. 5, the computation time of the scale size 100 x 100 Approach T1 started to rise highly because the column-generation approach had to spend computation time to solve the RMP matrix that was added by the large number of columns.

Table 3 The scale sizes, the computation time, and new column and iterations of Approach A1 and A2

Items	AP	No. of tasks x No. of assignees		
		100 x 100	1000 x 1000	7000 x 7000
The avg. Compu tation. time (sec.)	A1	82	34,610	>86,400
	A2	2.14	6,446	>86,400
Ratio of the avg. computation A1 and A2 (times)	A1& A2	38	5	-
No. of new columns	A1	378	2,969	-
No. of iterations	A1	382	7,894	-
No. of experiments		20	5	2

According to Table 3, the circumstance of the computation time of Approach A1 and A2 was similar to Approach T1 and T2. The ratio between the computation time of A1 and A2 reduced sharply when the number of customers rose. This implies that the ratio between the computation time of A1 and A2 reduces to come nearer from 38 to 5 times shown as Fig. 6.

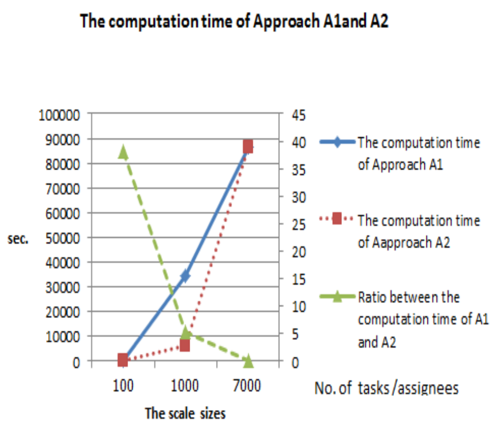


Fig. 6 The computation time of Approach A1 and A2

In Fig. 6, the computation time of the scale size 1,000 x 1,000 increased greatly due to long time for generating the new column.

When approach T1 was executed with 100 customers x 100 plants/warehouses, the computation times varied from 893 to 21,309 sec., as shown in Fig. 7. This indicates that the negative reduced costs of new column have been constructed: accordingly, the column generation must add more columns until the reduced costs of all columns are nonnegative.

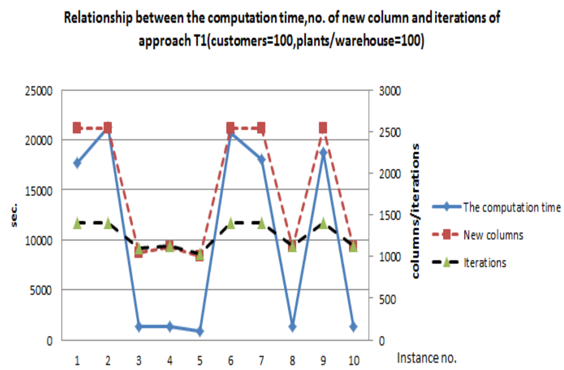


Fig. 7 Relationship between the computation time and no. of new columns and iterations of the scale size 100 customers x 100 plants/warehouses of Approach T1

In Fig. 7, the line of computation times, new columns, and iterations for 100 customers x 100 plants/warehouses using Approach T1 had the same characteristic. The computation time and the number of new columns corresponded directly; consequently, the computation time must depend on the number of new columns. When the numbers of customers or plants/warehouses rise greatly, the computation time also will increase corresponding. This is indicated in Fig. 8.

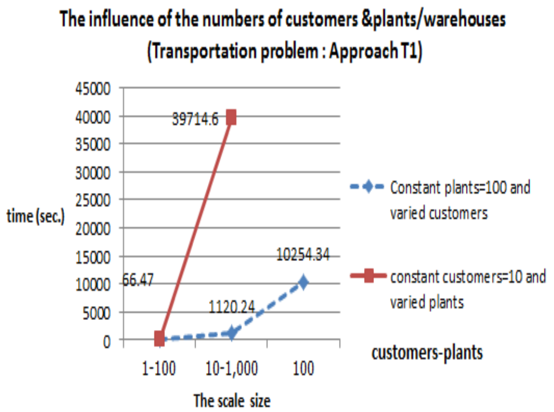


Fig. 8 The influence of the numbers of customers and plants/warehouses affects the computation time

In Fig. 8, for the column-generation technique for the TP (Approach T1), the number of customers increases 10 times and the number of plants/warehouses is constant (100), so its computation time rises about 9 to 16 times, while the number of plants/warehouses also increases 10 times and the number of customers is constant (10); consequently, its computation time expanded by about 600 times. The experiment of the column-generation method indicates that the number of plants/warehouses influences the computation time greater than the number of customers.

4. Conclusion

This paper has proposed a column-generation approach to solve the LP for the large-scale transportation and assignment problem. The column-generation technique can solve the LP of transportation problem on the largest scale size: 16,333 customers x 100 plants/warehouses, whereas the classical simplex method can compute only 11,063 customers x 100 plants/warehouses. The column-generation technique can be executed with too many variables, which increases the number of customers be 47.64% from the classical simplex method. For the assignment problem, the column generation can be performed on the largest scale sizes: 12,900 tasks x 12,900 assignees, while the classical simplex method can compute only 7,400 tasks x 7,400 assignees. This technique can be used with huge variables, which will expand the number of tasks/assignees by 74.32% from the classical simplex method. The column-generation method for transportation depends on the number of new columns generated. If more columns are added, the computation time will also be changing the numbers of customers and plants/warehouses directly affects the computation time. The impact of the large number of plants/warehouses (constrains) on the computation time is greater than the large number of customers (variables).

5. Recommendations

This paper can be developed the result as follows:

1. The efficient initial BF solution will help to reduce the computation time for solving the LP. This paper uses the northwest corner rule to construct the BF. This method is not to consider a unit cost, whereas the least cost method or VAM concern a unit cost that affects to reduce new column-generation and the computation time.

2. The source code of Approach T1 and A1 can be developed with Microsoft Visual C++ so that the computation time should decrease significantly.

6. References

[1] A. Charnes, D. Karney, D. Klingman, J. Stutz and F. Glover, “Past, present and future of large scale transshipment computer codes and applications.” Computers & Operations Research, Vol. 2, pp. 71-83, 1973.

[2] A. M. Geoffrion, “Elements of large scale mathematical programming: Part II : synthesis of algorithms and bibliography,” Management

Science, Vol.16, No.11, Theory Series, pp.676-691, 1970.

[3] N. Tomizawa, “On some techniques useful for solution of transportation network problems.” Network, Vol. 1(2), pp. 173–194, 1971.

[4] D. L. Miller, J. F. Pekny and G. L. Thompson, “Solution of large dense transportation problems using a parallel primal algorithm.” Management Science Research Report No.546, pp. 1-25, 1988.

[5] Y. Shoichiro and K. Tamotsu, “An efficient algorithm for the linear assignment problem,” Electronics and Communications in Japan (Part III: Fundamental Electronic Science), Volume 73, Issue 12, pp. 28–36, 1990.

[6] E. Balas, “Solution of Large Scale Transportation Problems through Aggregation,” Operations Research, Vol. 13, No. 1, pp. 82-93, 1963.

[7] C. Barnhart, E. L. Johnson, G. L. Nemhauer, M. W. P. Saverlsbergh, “Branch-and-price: Column generation for solving huge integer programs,” Operations Research, vol. 46, No. 3, pp. 316-329, 1998.

[8] E. J. Zak, “Row and column generation technique for a multistage cutting stock problem,” Computers & Operations Research, Vol.29, pp. 1143-1156, 2002.

[9] S. Gabtensi and M. Gronkvist, “Combining column generation and constraint programming to solve the tail assignment problem,” Annals of Operations Research, Vol. 171, pp.61–76, 2009.

[10] P. Charnsethikul, “A column generations approach for the products mix based semi-infinite linear programming model,” International Journal of Management Science and Engineering Management, Vol. 6(2), pp.106-109, 2011.

[11] T. Nishi, Y. Muroi and M. Inuiguchi, “Column generation with dual inequalities for railway crew scheduling problems,” Public Transportation Vol. 3, pp. 25-42, 2011.

[12] H. M. Wagner, Principles of operations research with applications to managerial decisions, 2nd edition, New Delhi: Prentice-Hall of India Private Limited, 1975.

[13] F. S. Hillier and G. I. Lieberman, Introduction to operations research, 9th edition, Singapore: McGraw-Hill, 2010.

[14] S. I. Gass, Linear programming methods and applications, 5th edition, Dover Publications, 2010.

[15] G. B. Danzig and M. N. Thapa, Linear programming: 2: Theory and extensions, New York: Springer-Verlag, 2003.

[16] H. A. Taha, Operations research: An introduction, 9th edition, Singapore, Pearson international edition, 2010.

[17] K. Edlund, L. E. Sokoler and J. B. Jorgensen, “A Primal-Dual Interior-Point Linear Programming Algorithm for MPC.” Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference Shanghai, P.R. China, December 16-18, 2009.

[18] Yu. G. Evtushenko, A. I. Golikov and N. Mollavery, “Augmented Lagrangian method for large-scale linear programming problems” Optimization Methods and Software,Vol. 20, No. 4–5, pp. 515–524, 2005.

[19] W. Soon and H.Q. Ye, “Currency arbitrage detection using a binary integer programming model” International Journal of Mathematical Education in Science and Technology, Vol. 42, No. 3, pp. 369–376, 2011.

[20] http://people.brunel.ac.uk/~mastjjb/jeb/info.html (accessed on December 3, 2013)