Journal of The Korea Society of Computer and Information
Vol. 20 No. 8, pp. 105-112, August 2015
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.8.105

An Assignment Problem Algorithm Using Minimum Cost Moving Method

Sang-Un Lee*

Abstract

Generally, the optimal solution of assignment problem has been obtained by Hungarian algorithm with $O(n^3)$ time complexity. This paper proposes more simple algorithm with $O(n^2)$ time complexity than Hungarian algorithm. The proposed algorithm simply selects minimum cost in each row, and classified into set S, H, and T. Then, the minimum cost is moved from S to T and $S \to H, H \to T$. The proposed algorithm can be obtain the same optimal solution as well-known algorithms and improve the optimal solution of partial unbalanced assignment problems.

▸ Keyword : Hungarian algorithm, Balanced assignment, Unbalanced assignment, Minimum cost, Optimal solution

I. Introduction

할당 문제 (assignment problem)는 수송 문제 (transportation problem)의 특별한 경우로, 다수의 공급처 (source, $S_i, i=1, 2, \cdots, m$)와 수요처 (demand, $D_j, j=1,2,\cdots,n$)가 존재하며 모든 공급량과 요구량이 항상 1인 경우이다. 또한, 수송비용 (c_{ij})이 모두 다르며, 한 공급처에서 반드시 한 수요처로만 수송이 이루어져야만 한다. 이 경우 총 수송비용의 합이 최소가 되는 최적해 $z=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij}$ 를 찾도록 $x_{ij}=1$ 을 할당하는 문제이다[1-3]. 이는 m개의 작업을 n대의 기계로 작업을 수행하는 경우 최적의 작업할당 문제에도 적용된다.

$m \times n$ 비용 행렬에서 $m=n$인 경우를 균형 할당 문제 (balanced assignment problem, BAP), $m \neq n$인 경우를 불균형 할당 문제 (unbalanced assignment problem, UAP)라 한다[3].

할당 문제를 해결하는 방법으로 거의 대부분은 헝가리안 알고리즘 (Hungarian algorithm, HA)[1-3]을 적용하고 있으며, 일부는 유전자 알고리즘 (genetic algorithm, GA)[4]을 시도하는 경우도 있다. HA의 수행 복잡도는 $O(n^3)$으로 균형 할당 문제에 대해서는 최적해 (optimal solution)를 항상 찾을 수 있다고 알려져 있다[1]. 그러나 불균형 할당 문제에 대해서는 최적의 해법을 찾지 못할 수 있기 때문에 알고리즘을 적용하기 전에 비용이 모두 0인 가상 (dummy)의 행이나 열을 추가하여 균형 할당을 만들어야만 한다[3]. 또한, HA는 마지막으로 얻은 0의 비용들 중에서 각 열에서 중복되지 않게 1개씩만 선택하여야 하는 불편함이 따르며, 0을 포함하는 최소한의 선을 m개를 긋는 과정이 반복적으로 수행되어야만 한다.

이 분야에서 제기되는 문제로, HA에 비해, 비용 행렬의 크기와 무관하게 적용할 수 있으며 균형과 불균형의 모든 경우에 대해 항상 최적 해를 찾는 단순하면서도 일반적으로 적용할 수 있는 알고리즘이 존재하느냐이다.

이러한 제기된 문제를 해결하고자, 본 논문에서는 $n \times n$ 비용행렬에서 단순히 각 행에 존재하는 최소비용만을 이동시키는 방법으로 최적 배정의 해를 찾아가는 알고리즘을 제안하였다. 제안된 알고리즘을 최소비용 이동 (minimum cost moving, MCM) 알고리즘이라 부르며, $n \times n$ 비용행렬에 대해 수행되어 수행복잡도는 $O(n^2)$이다. 결국, 제안된 알고리즘으로 최적 해를 구할 수 있다면 HA의 수행 복잡도 $O(n^3)$을 $O(n^2)$으로 개선시킨 새로운 방법임을 증명하는 셈이다.

2장에서는 할당 문제의 최적 해를 찾는 대표적인 HA를 살펴보고 문제점을 고찰해 본다. 3장에서는 최적의 해법을 쉽게 찾는 최소비용 이동 알고리즘을 제안한다. 4장에서는 최소비용 이동 알고리즘을 다양한 균형과 불균형 할당 문제 사례들에 적용하여 최적 해를 찾는지 평가해 본다.

• First Author: Sang-Un Lee, Corresponding Author: Sang-Un Lee
*Sang-Un Lee (sulee@gwnu.ac.kr), Dept. of Multimedia Engineering, Gangneung-Wonju National University
• Received: 2015. 05. 19, Revised: 2015. 06. 09, Accepted: 2015. 06. 19.

II. Definition and Related Works

할당 문제는 공급량과 요구량이 모두 1인 수송 문제의 특별한 경우이다. 하나의 공급처는 반드시 비용을 최소로 하는 하나의 수요처만을 선택하여야만 한다. 또한, 하나의 수요처는 반드시 하나의 공급처만을 선택해야만 한다. 이는 기계에 일을 부여하는 경우, 사람에게 임무를 부여하는 경우 등에 일반적으로 적용하기 때문에 할당 문제로 부르기도 한다. 할당 문제는 식 (1)의 조건을 만족하는 최적 해를 찾는다.

$$z=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij} \tag{1}$$

$$s.t.\quad \sum_{i=1}^{m}x_{ij},\ \text{for } i=1,2,\cdots,n \quad /\!*\ \text{각 요구량} = 1$$

$$\sum_{j=1}^{n}x_{ij},\ \text{for } i=1,2,\cdots,m \quad /\!*\ \text{각 공급량} = 1$$

$$x_{ij} \geq 0,\ \text{for } \forall_{ij}$$

할당 문제의 최적 해를 찾는 HA는 헝가리 수학자인 Harold Kuhn이 1955년에 제안하고, 1957년에 James Munkres가 보완하였다. 따라서 할당 알고리즘 또는 Kuhn-Munkres 알고리즘이라 부르기도 한다[2].

HA는 먼저 불균형 할당 문제인 경우 가상의 행이나 열을 추가하여 균형 할당문제로 변환시킨다. 다음으로, 각 행에서 최소비용을 찾아 각 열의 셀 값에서 최소 비용 값을 감산한다. 각 열에 대해서도 동일하게 최소 비용을 찾아 각 행의 셀 값에서 최소비용 값을 감산한다. 다음으로 $c_{ij}=0$ 이 되는 셀에 대해 직선을 긋는다. 선에 포함되지 않은 $c_{ij}>0$ 들인 감소된 비용 행렬 (reduced cost matrix)에서 최소 비용 $_{\min}c_{ij}$를 찾아 $c_{ij}=c_{ij}-{}_{\min}c_{ij}$, 행과 열의 선이 교차된 셀 $c_{ij}=c_{ij}+c_{ij}$로 치환하고 각 열에서 0이 중첩되지 않게 선택하여 선택된 셀의 비용을 모두 더하여 최적 해 z를 얻는다.

그림 1은 Kumar[5]에서 인용된 할당 문제이다. 행은 일을, 열은 기계를, 행렬의 값 c_{ij}는 작업 수행비용이다. 4개의 작업을 4개의 기계에 중복되지 않게 할당하여 총 작업비용을 최소화시키는 제약조건을 만족하는 최적 해를 찾아야 한다.

c_{ij}		Machine 1	Machine 2	Machine 3	Machine 4	공급량 (s_i)
Job	1	1	4	6	3	1
	2	8	7	10	9	1
	3	4	5	11	7	1
	4	6	7	8	5	1
요구량 (d_j)		1	1	1	1	

Fig. 1. A_1 assignment problem

Kumar[5]가 그림 1에 대해 HA를 적용하여 최적 해를 찾은 결과는 그림 2와 같다. Step 2를 2회, Step 3을 1회 수행하여 0을 모두 포함하는 최소한의 선을 m개 얻었으며, 작업을 $x_{11}=x_{23}=x_{32}=x_{44}=1$ 로 할당하여 최적해 $z=1+10+5+5=21$ 을 얻었다.

Lee[6]은 HA가 갖고 있는 문제점을 다음과 같이 4가지를 제시하였다. 첫 번째로 불균형 할당문제인 경우 부정확한 결과를 얻을 수 있어 가상 행이나 열을 추가하여 $m\times n(m=n)$인 균형 할당을 만드는 과정이 필요하다. 두 번째로, 비용 행렬이 크고 0을 다수 포함하는 경우 0을 모두 포함하는 최소한의 라인을 긋는 방법이 어렵다. 세 번째로, 최종적으로 얻은 0 값들에 대해 각 열을 기준으로 중첩되지 않게 1개씩만 선택하는 어려움이 있다. 네 번째로, HA 의 수행 복잡도는 $O(n^3)$으로 큰 비용행렬인 경우 최적 해법을 도출하기 위해 많은 시간이 소요된다.

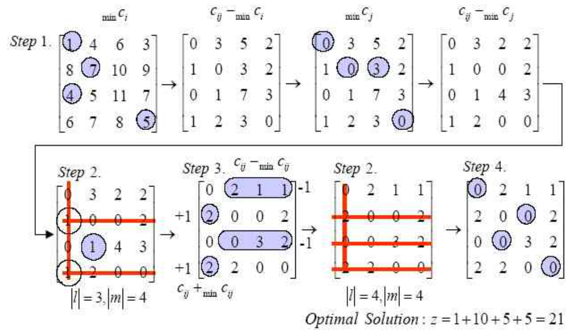


Fig. 2. Hungarian algorithm for A_1 assignment problem

할당문제는 $m\times n(m=n)$ 비용 행렬에서 행과 열이 1:1로 중복없이 최소의 합이 되는 비용 셀 c_{ij}를 선택하는 것이 목적이다. 이를 수행하는 과정에서 행과 열의 양측을 모두 만족시키는 유일한 방법이 없으며, 선택을 하는 과정에서 특정 행이나 열을 중복하여 선택하는 경우가 발생된다. 이러한 문제점을 해결하면서 HA보다 단순한 방법으로, Lee[6]는 $m\times n$ 비용행렬에서 최소비용 $_{\min}c_{ij}$를 선택하고, 교환 최적화 기법을 적용한 알고리즘을 제안하였으며, Lee[7]은 $m\times n$ 비용행렬에서 최대비용 $_{\max}c_{ij}$를 삭제하는 기법을 제안하였고, Lee[8]은 각 행과 열에서 최소비용을 선택하고, 행과 열을 이동시키는 방법을 제안하였으며, Lee[9]는 각 행의 최소 비용을 선택하고, 중복 선택된 경우 최대비용이 적은 열부터 이동시키는 기법을 제안하였다. 3장에서는 Lee[9]와 반대 개념을 적용하여 중복 선택된 경우 최소비용을 이동시키고, 보다 큰 비용을 남겨두는 방법을 제안한다.

III. Minimum Cost Moving Algorithm

일반적으로, $n\times n$ 행렬의 행과 열 양쪽의 최적 해 (최소비용)을 모두 만족시키도록 n쌍을 선택하는 방법은 존재하지 않는다. 이 문제를해결할 수 있는 한 가지 방법으로 어느 한쪽에 대해서만 최적 값을 선택하고 나머지 한쪽은 최악의 값을 받아들이는 방법이 있을 수 있다. 이 방법의 결과에 대해 최악의 선택을 당한 쪽에는 보상을, 최적의 선택을 한 쪽에서는 양보

를 하여 균형된 최적 해를 찾는 조정 과정이 추가로 필요하다. 즉, 배정과 조정의 2단계를 수행해야 한다.

본 장에서는 HA의 수행 복잡도 $O(n^3)$을 $O(n^2)$으로 향상시키면서, 균형과 불균형 할당 문제 모두에 즉시 적용할 수 있는 또 다른 방법으로, 최소비용 이동 알고리즘을 제안한다. 제안된 방법은 위의 배정과 조정의 2단계 수행 방법에서 조정 과정을 제외하고, 단순히 배정과정 만을 수행하여 최적 해를 얻고자 한다.

일반적으로, $n \times n$ 비용행렬에 대해 각 행에 대해 최소 비용을 선택하면 특정 열에 대해서는 중복 선택되는 경우가 발생한다. 따라서, 중복되지 않는 n쌍의 최적 해를 선택하기 위해서는 열에서 중복 선택된 값들 중 비용증가분이 보다 적은 셀을 이동시켜야 최적 해에 보다 가까워 질 수 있을 것이다. 이러한 개념에 기반하여 본 장에서는 최소비용 이동 알고리즘으로 다음 방법을 적용하였다. 3개의 작업과 3대의 기계가 존재하며, 각 작업을 각 기계에서 작업하는 비용은 차이가 있다고 다음과 같이 가정하여 보자.

i \ j		Machine		
		M_1	M_2	M_3
Job	J_1	c_{11}	c_{12}	c_{13}
	J_2	c_{21}	c_{22}	c_{23}
	J_3	c_{31}	c_{32}	c_{33}

→

i \ j		Machine		
		M_1	M_2	M_3
Job	J_1	c_{11}		
	J_2	c_{21}		
	J_3		c_{32}	
선택 개수		$\lvert c_{\cdot 1}\rvert \geq$ S	$\lvert c_{\cdot 2}\rvert =$ H	$\lvert c_{\cdot 3}\rvert =$ T

할당문제에서의 목표 (goal)는 $|c_{\cdot 1}| = |c_{\cdot 2}| = |c_{\cdot 3}| = 1$ 을 선택하는 것이다. 여기서, $|c_{\cdot 1}| \geq 2$를 집합 S (source), $|c_{\cdot 2}| = 1$를 집합 H (hub), $|c_{\cdot 3}| = 0$를 집합 T (sink or Target)이라 하면 $S = \{c_{11}, c_{21}\}, H = \{c_{32}\}, T = \{\phi\}$ 이다. 따라서, M_1에 과다하게 할당된 $c_{11}, c_{21} \in S$ 중의 어느 하나가 $M_3 \in T$로 이동되어야만 한다. 여기서 이동 방법은 c_{11}에 대해서는 $c_{11} \xrightarrow{c_{ST}} c_{13}$, $\left\{c_{11} \xrightarrow{c_{SH}} c_{12} + c_{32} \xrightarrow{c_{HT}} c_{33}\right\}$ 의 2가지 방법이, c_{21}에 대해서는 $c_{21} \xrightarrow{c_{ST}} c_{23}$, $\left\{c_{21} \xrightarrow{c_{SH}} c_{22} + c_{32} \xrightarrow{c_{HT}} c_{33}\right\}$ 의 2가지 방법이 있다. 제안된 알고리즘은 $S \rightarrow T$ 이동의 c_{ST}들을 모두 계산하여 ${}_{\min}c_{ST}$를 결정한다. 다음으로 $H \rightarrow T$ 이동의 c_{HT}는 $c_{HT} < {}_{\min}c_{ST}$에 대해서만 $c_{SH} + c_{HT}$를 계산하여 최종적으로 ${}_{\min}\{{}_{\min}c_{ST}, {}_{\min}(c_{SH} + c_{HT})\}$를 이동시킨다. 최소비용 이동 (MCM) 알고리즘은 $n \times n$ 비용행렬에 대해 수행되므로 수행 복잡도는 $O(n^2)$이며, 다음과 같이 수행된다.

$m > n$인 불균형 할당 문제는 편의상 행렬을 교환한 $n < m$으로 변경, n행, m열로 취급.

Step 1. for $i = 1, 2, \cdots, m \quad (m \leq n)$
 ${}_{\min}c_i$ 선택, 단, 동일 값 존재시 미선택 셀 선택
 $|c_j| \geq 2 \in S$, $|c_j| = 1 \in H$, $|c_j| = 0 \in T$

Step 2. $T = \{\phi\}$ 일 때까지 T 집합의 j열에 대해 기본적으로 L-R 순서로 다음과 같이 수행. $|T| \geq 2$인 경우 R-L로 검증, 만약, L-R보다 z가 최소값이면 R-L로 z 결정.
 (1) $S \rightarrow T$ 이동의 ${}_{\min}c_{ST}$ 계산
 (2) $H \rightarrow T$ 이동의 $c_{HT} < {}_{\min}c_{ST}$에 대해 $c_{SH} + c_{HT}$ 계산.
 (3) ${}_{\min}\{{}_{\min}c_{ST}, {}_{\min}(c_{SH} + c_{HT})\}$ 을 선택, 이동.
 불균형 할당문제의 경우 $S = \{\phi\}$ 일 때까지 위 (1) ~ (3) 수행. $S = \{\phi\}$ 이면 T 집합의 j열에 대해서는 $c_{ST} < 0$ 만 이동

주어진 데이터가 $m > n$인 불균형 할당 문제인 경우, MCM은 편의상 $n < m$으로 변경하여 적용하였다. 여기서의 의미는 열이 행보다 개수가 보다 크도록 행렬을 바꾸어 배치하였음을 의미한다. 왜냐하면 $n < m$인 경우 n쌍만을 선택할 수 있으며, $m - n$의 열은 선택되지 않기 때문이다. 따라서, 알고리즘 수행 횟수를 줄이기 위해 이러한 방법을 적용하였다.

그림 1의 A_1 문제에 MCM 알고리즘을 적용한 과정은 그림 3에 제시되어 있다.

$$
{}_{\min}c_i \begin{array}{c}\rightarrow\\ \rightarrow\\ \rightarrow\\ \rightarrow\end{array}
\begin{bmatrix} 1 & 4 & 6 & 3 \\ 8 & 7 & 10 & 9 \\ 4 & 5 & 11 & 7 \\ 6 & 7 & 8 & 5 \end{bmatrix}
\rightarrow
\begin{bmatrix} 1 & 4 & 6 & 3 \\ 8 & 7 & 10 & 9 \\ 4 & 5 & 11 & 7 \\ 6 & 7 & 8 & 5 \end{bmatrix}
\begin{array}{l} +5 \\ +3(+1) \\ +7 \\ +3(+2) \end{array}
\rightarrow
\begin{bmatrix} 1 & 4 & 6 & 3 \\ 8 & 7 & 10 & 9 \\ 4 & 5 & 11 & 7 \\ 6 & 7 & 8 & 5 \end{bmatrix}
$$

$z = 1 + 10 + 5 + 5 = 21$

Fig. 3. MCM algorithm for A_1 problem

A_1 할당문제는 4×4의 균형할당 문제로 행과 열을 교환하지 않는다. Step 1에서 ${}_{\min}c_i$를 선택한 결과 $S = \{1\}, c_{11} = 1, c_{31} = 4,\ H = \{2, 4\}, c_{22} = 7, c_{44} = 5, T = \{3\}$을 얻는다. Step 2에서 ${}_{\min}\left\{c_{11}(1) \xrightarrow{c_{ST}(5)} c_{13}(6), c_{31}(4) \xrightarrow{c_{ST}(7)} c_{33}(11)\right\}$로 ${}_{\min}c_{ST} = 5$를 얻는다. 따라서 $H = \{2, 4\}, c_{22} = 7, c_{44} = 5$에 대해서는 $c_{22}(7) \xrightarrow{c_{HT}(3)} c_{23}(10)$와 $c_{44}(5) \xrightarrow{c_{HT}(3)} c_{43}(8)$로 $c_{HT} = 3 < {}_{\min}c_{ST} = 5$를 충족시켜 이동 대상 후보들로 결정된다. ${}_{\min}\left\{c_{31}(4) \xrightarrow{c_{SH}(1)} c_{32}(5) + c_{22}(7) \xrightarrow{c_{HT}(3)} c_{23}(10), c_{44}(5) \xrightarrow{c_{SH}(2)} c_{42}(7) \xrightarrow{c_{HT}(3)} c_{23}(10)\right\}$으로, $c_{31}(4) \rightarrow c_{32}(5)$ 와 $c_{22}(7) \rightarrow c_{23}(10)$으로 이동되어 $z = 1 + 10 + 5 + 5 = 21$의 최적 해를 얻는다.

HA는 행과 열의 ${}_{\min}c_i, {}_{\min}c_j$인 최소값 선택, $c_{ij} = c_{ij} - {}_{\min}c_i$와 $c_{ij} = c_{ij} - {}_{\min}c_j$ 계산, 선을 긋고 $c_{ij} > 0$ 인 셀에 대한 $c_{ij} = c_{ij} - {}_{\min}c_{ij}$와 행과 열의 선이 교차된 c_{ij}에 대한 $c_{ij} = c_{ij} + c_{ij}$를 수행하는 복잡함을 갖고 있다. 또한, Lee[6,7, 8,9]가 제안한 다양한 기법들은 중복 선택시 어떤 셀을 이동시키며, 최적화를 어떻게 시킬 것인지를 추가적으로 고려해야만 한다. 반면에, 제안된 MCM 알고리즘은 단순히 행에 대해 최소값을 선택하고, 다중 선택 열에서 선택되지 않은 열로 최소 비용 증가분을 가진 비용을 이동시키는 단순화된 방법으로 최적 해를 구하였다. 따라서, 할당문제에 대해 제안된 MCM 알고리즘이 가장 간단하면서도 최적 해를 구할 수 있음을 알 수 있다.

IV. Experiments and Evaluation

본 장에서는 그림 4의 26개 균형 할당과 7개 불균형 할당 문제를 대상으로 MCM 알고리즘의 적용성을 평가해 본다. 평가 결과분석에는 최적 해를 얻었는지 유무, 알고리즘 수행 복잡도와 더불어 메모리 사용 공간 복잡도 등 다양한 요인들을 분석해야만 한다. 그러나, 기존 알고리즘들은 단지 최적 해를 얻는지 유무만으로 알고리즘의 성능을 평가하였으며, 복잡도는 평가 대상에서 고려하지 않아 비교 대상 데이터가 없는 관계로, 부득이 최적 해 비교로만 알고리즘의 성능을 평가한다.

$$\begin{bmatrix}1&2&3&4\\2&4&6&8\\3&6&9&12\\4&8&12&16\end{bmatrix}$$
B_1 [10]

$$\begin{bmatrix}5&2&3&4\\7&8&4&5\\6&3&5&6\\2&2&3&5\end{bmatrix}$$
B_2 [11]

$$\begin{bmatrix}5&7&4&3\\5&5&4&3\\6&5&4&3\\5&5&6&6\end{bmatrix}$$
B_3 [12]

$$\begin{bmatrix}7&5&8&2\\7&8&9&4\\3&5&7&9\\5&5&6&7\end{bmatrix}$$
B_4 [13]

$$\begin{bmatrix}8&6&5&7\\6&5&3&4\\7&8&4&6\\6&7&5&6\end{bmatrix}$$
B_5 [14]

$$\begin{bmatrix}8&2&3&3\\2&7&5&8\\0&9&8&4\\2&5&6&3\end{bmatrix}$$
B_6 [15]

$$\begin{bmatrix}13&4&7&6\\1&11&5&4\\6&7&2&8\\1&3&5&9\end{bmatrix}$$
B_7 [16]

$$\begin{bmatrix}6&12&3&7\\13&10&12&8\\2&5&15&20\\2&7&8&13\end{bmatrix}$$
B_8 [10]

$$\begin{bmatrix}90&75&75&80\\35&85&55&65\\125&95&90&105\\45&110&95&115\end{bmatrix}$$
B_9 [17]

$$\begin{bmatrix}14&5&8&7\\2&12&6&5\\7&8&3&9\\2&4&6&10\end{bmatrix}$$
B_{10} [3,18]

$$\begin{bmatrix}15&29&35&20\\21&27&33&17\\17&25&37&15\\14&31&39&21\end{bmatrix}$$
B_{11} [10]

$$\begin{bmatrix}18&26&17&11\\13&28&14&26\\38&19&18&15\\19&26&24&10\end{bmatrix}$$
B_{12} [10]

$$\begin{bmatrix}20&10&15&22\\13&11&7&15\\16&12&10&15\\29&10&14&18\end{bmatrix}$$
B_{13} [10]

$$\begin{bmatrix}20&25&22&28\\15&18&23&17\\19&17&21&24\\25&23&24&24\end{bmatrix}$$
B_{14} [10]

$$\begin{bmatrix}42&35&28&21\\30&25&20&15\\30&25&20&15\\24&20&16&12\end{bmatrix}$$
B_{15} [10]

$$\begin{bmatrix}3&9&2&3&7\\6&1&5&6&6\\9&4&7&10&3\\2&5&4&2&1\\9&6&2&4&5\end{bmatrix}$$
B_{16} [14]

$$\begin{bmatrix}3&8&2&10&3\\8&7&2&9&7\\6&4&2&7&5\\8&4&2&3&5\\9&10&6&9&10\end{bmatrix}$$
B_{17} [14]

$$\begin{bmatrix}3&8&9&15&10\\4&10&7&16&14\\9&13&11&19&10\\8&13&12&20&13\\1&7&5&11&9\end{bmatrix}$$
B_{18} [19]

$$\begin{bmatrix}5&4&6&4&1\\2&5&4&10&5\\10&12&10&6&8\\1&3&4&2&6\\2&5&8&11&7\end{bmatrix}$$
B_{19} [20]

$$\begin{bmatrix}8&16&15&91&64\\83&42&93&75&27\\76&95&75&81&50\\20&42&96&90&24\\38&28&2&15&81\end{bmatrix}$$
B_{20} [21]

$$\begin{bmatrix}30&37&40&28&40\\40&24&27&21&36\\40&32&33&30&35\\25&38&40&36&36\\29&62&41&34&39\end{bmatrix}$$
B_{21} [10]

$$\begin{bmatrix}40&40&35&25&50\\42&30&16&25&27\\50&48&40&60&50\\20&19&20&18&25\\58&60&59&55&53\end{bmatrix}$$
B_{22} [10]

$$\begin{bmatrix}22&30&26&16&25\\27&29&28&20&32\\33&25&21&29&23\\24&24&30&19&26\\30&33&32&37&31\end{bmatrix}$$
B_{23} [22]

$$\begin{bmatrix}4&4&8&8&3&9\\4&6&4&1&4&6\\10&4&5&1&8&3\\2&7&2&6&7&2\\2&1&5&5&2&5\\2&8&3&1&2&1\end{bmatrix}$$
B_{24} [23]

$$\begin{bmatrix}10&12&8&10&8&12\\9&10&8&7&8&9\\8&7&8&8&8&6\\12&13&14&14&15&14\\9&9&9&8&8&10\\7&8&9&9&9&8\end{bmatrix}$$
B_{25} [10]

$$\begin{bmatrix}79&24&13&53&47&66&85&17&92&47&46&13\\43&59&33&95&55&97&34&55&84&94&26&56\\29&52&26&27&13&33&70&11&71&86&6&76\\88&83&64&72&90&67&27&47&83&62&35&38\\65&90&56&62&53&91&48&23&6&89&49&33\\44&79&86&93&71&7&86&59&17&56&45&59\\35&51&9&91&39&32&3&12&79&25&79&81\\50&12&59&32&23&64&20&94&97&14&11&97\\25&17&39&0&38&63&87&14&4&18&11&45\\68&45&99&0&94&44&99&59&37&18&38&74\\93&36&91&30&44&69&68&67&81&62&66&37\\19&36&5&50&49&94&95&17&63&41&84&1\end{bmatrix}$$
B_{26} [24]

(a) Balanced assignment

$$\begin{bmatrix}7&5&8&4\\5&6&7&4\\8&7&9&8\end{bmatrix}$$
UB_1 [10]

$$\begin{bmatrix}13&16&12&11\\15&0&13&20\\5&7&10&6\end{bmatrix}$$
UB_2 [9,14,21]

$$\begin{bmatrix}20&25&22&28\\15&18&23&17\\19&17&21&24\end{bmatrix}$$
UB_3 [6]

$$\begin{bmatrix}60&80&50\\50&30&60\\70&90&40\\80&50&70\end{bmatrix}$$
UB_4 [6]

$$\begin{bmatrix}9&14&19&15\\7&17&20&19\\9&18&21&18\\10&12&18&19\\10&15&21&16\end{bmatrix}$$
UB_5 [6]

$$\begin{bmatrix}37.7&43.4&33.3&29.2\\32.9&33.1&28.5&26.4\\33.8&42.2&38.9&29.6\\37.0&34.7&30.4&28.5\\35.4&41.8&33.6&31.1\end{bmatrix}$$
UB_6 [22]

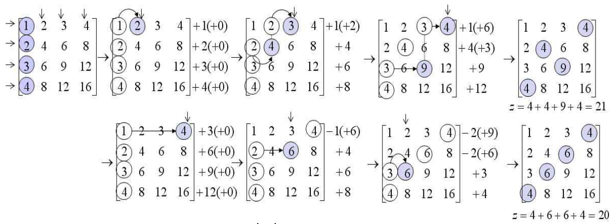
$$\begin{bmatrix}34&31&20&27&24&24&18&33&35&19\\14&14&22&34&26&19&22&29&22&19\\22&16&21&27&35&25&30&22&23&23\\17&21&24&16&31&22&20&27&26&17\\17&29&22&31&18&19&26&24&25&14\\26&29&37&34&37&20&21&25&27&27\\30&28&37&28&29&23&19&33&30&21\\28&21&30&24&35&20&24&24&32&24\\19&18&19&28&28&27&26&32&23&22\\30&22&29&19&30&29&29&21&20&18\\29&25&35&29&27&18&30&28&19&23\\15&19&19&33&22&24&25&31&33&21\\27&32&27&29&29&21&19&25&20&27\end{bmatrix}$$
UB_7 [4]

(b) Unbalanced assignment

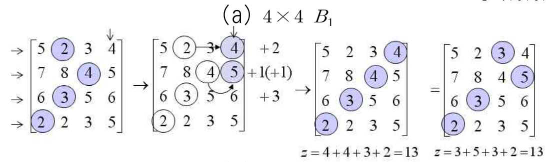
Fig. 4. Experimental Data for Assignment Problems

26개의 균형 할당 문제 [3,10-24]에 대해, 4×4 비용행렬은 15개, 5×5 비용행렬은 8개, 6×6 비용행렬은 2개, 12×12 비용행렬은 1개로 구성되어 있다. 7개의 불균형 할당 문제[4,6,9,10,14,21,22]에 대해, 3×4 비용행렬은 UB_1, UB_2, UB_3, 4×3 비용행렬은 UB_4, 5×4 비용행렬은 UB_5, UB_6, 13×10 비용행렬은 UB_7이다.

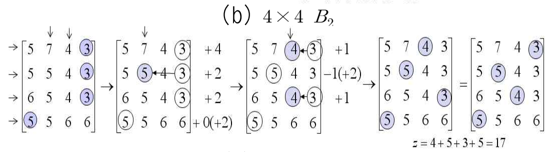
균형 할당 비용행렬에 대한 MCM 알고리즘을 적용한 결과는 그림 5에 제시되어 있다. 26개 균형 할당 문제에 대해 제안된 최적 할당 알고리즘은 모두 최적 해를 찾았다. B_{26}은 Dantzig[24] 원문에서 최적 해를 제시하지 않았다. 반면에 제안된 최적 할당 알고리즘은 최적 해 $z=189$ 를 찾았다.



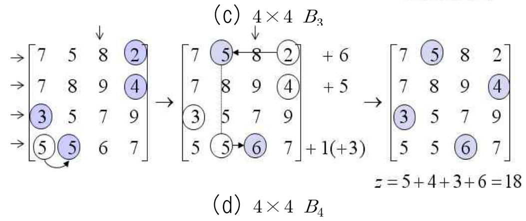
(a) 4×4 B_1



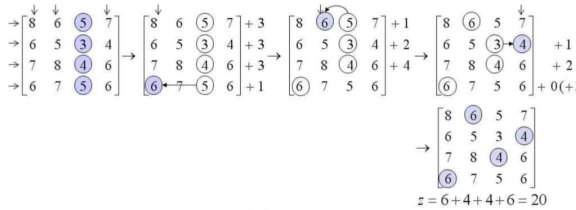
(b) 4×4 B_2



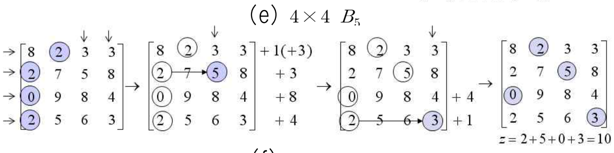
(c) 4×4 B_3



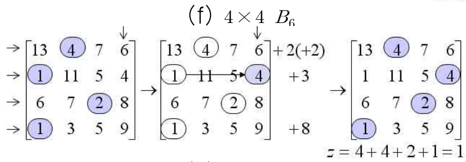
(d) 4×4 B_4



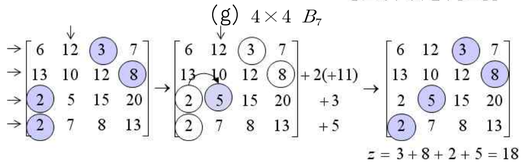
(e) 4×4 B_5



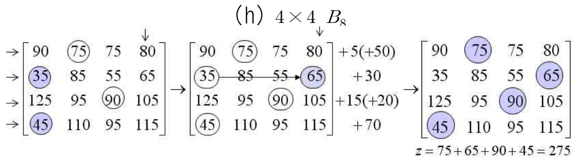
(f) 4×4 B_6



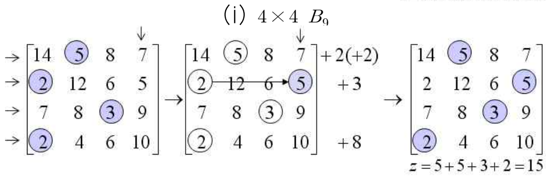
(g) 4×4 B_7



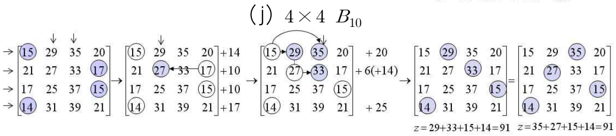
(h) 4×4 B_8



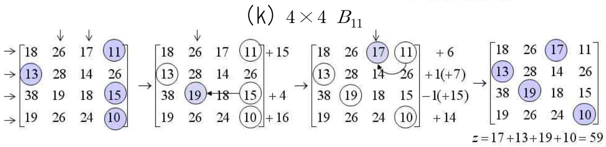
(i) 4×4 B_9



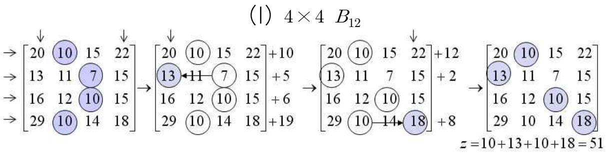
(j) 4×4 B_{10}



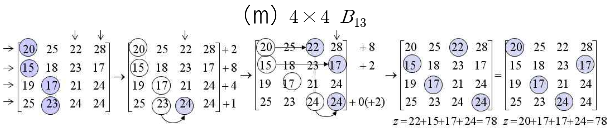
(k) 4×4 B_{11}



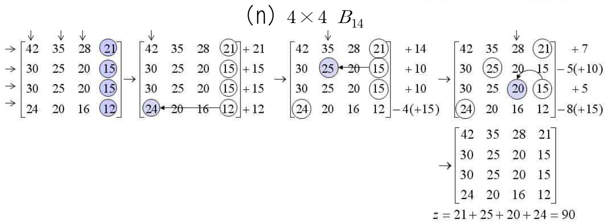
(l) 4×4 B_{12}



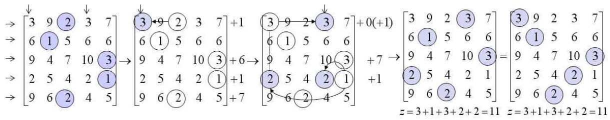
(m) 4×4 B_{13}



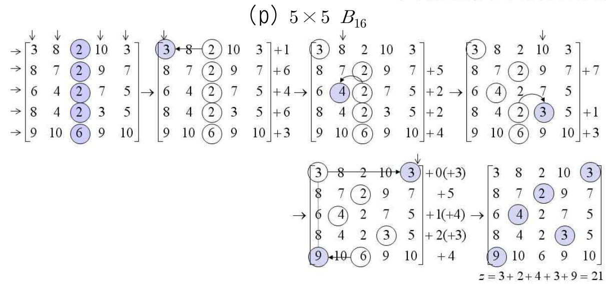
(n) 4×4 B_{14}



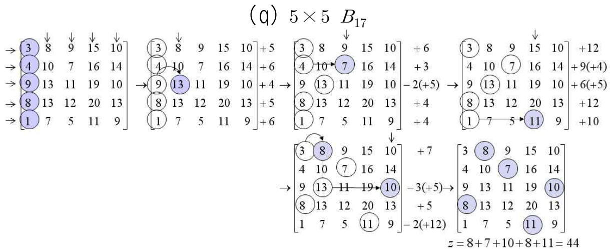
(o) 4×4 B_{15}



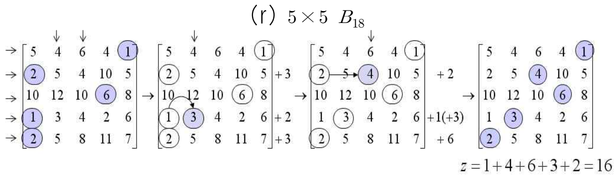
(p) 5×5 B_{16}



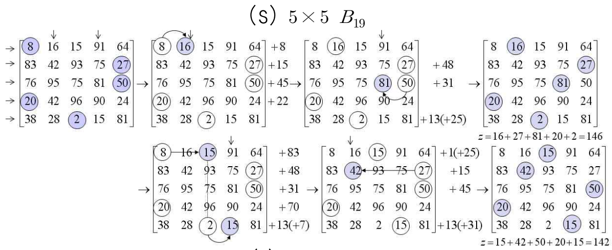
(q) 5×5 B_{17}



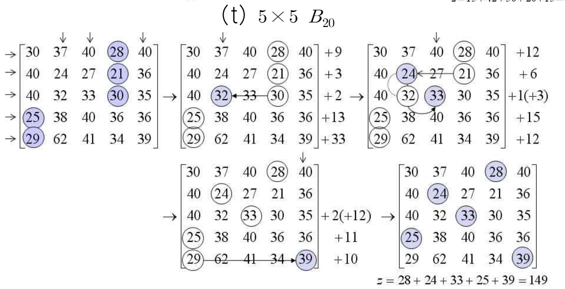
(r) 5×5 B_{18}



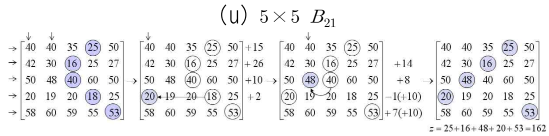
(s) 5×5 B_{19}



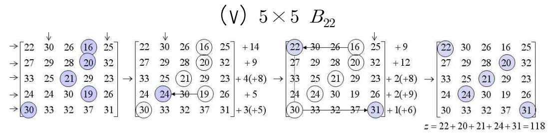
(t) 5×5 B_{20}



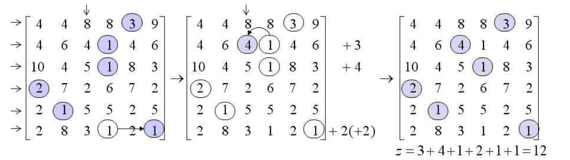
(u) 5×5 B_{21}



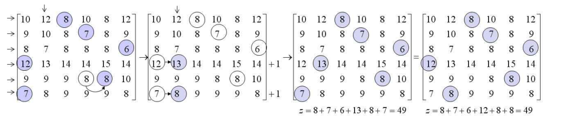
(v) 5×5 B_{22}



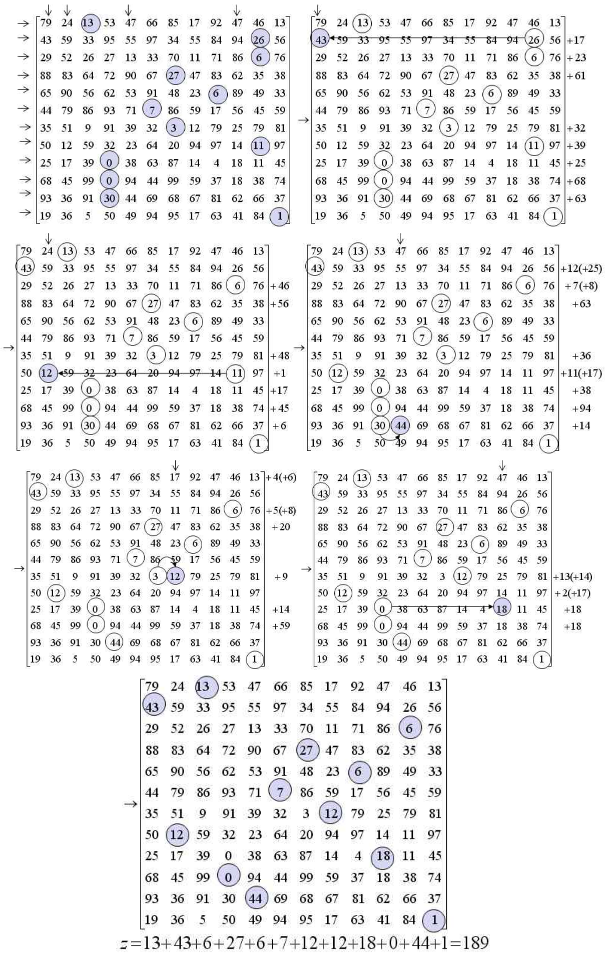
(w) 5×5 B_{23}



(x) 6×6 B_{24}



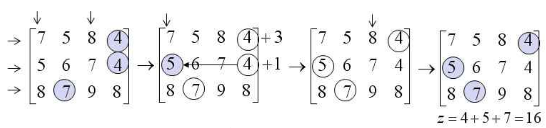
(y) 6×6 B_{25}



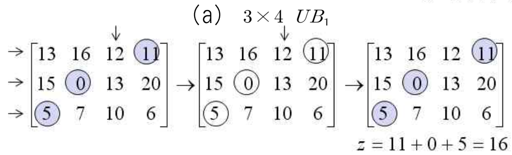
(z) 12×12 B_{26}

Fig. 5. MCM Algorithm for Balanced Assignment Problem

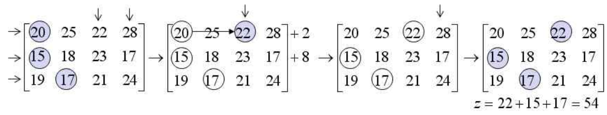
7개 불균형 할당 문제의 3×4, 4×3, 5×4와 13×10 비용 행렬에 대해 MCM 알고리즘을 적용한 결과는 그림 6에 제시하였다.



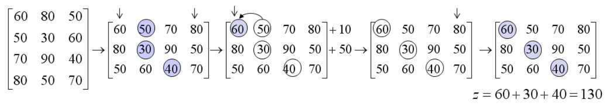
(a) 3×4 UB_1



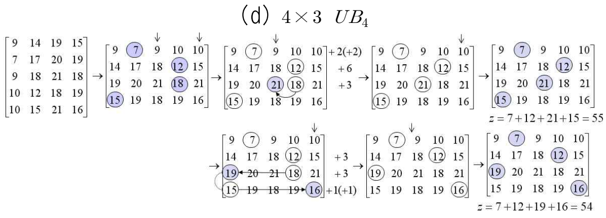
(b) 3×4 UB_2



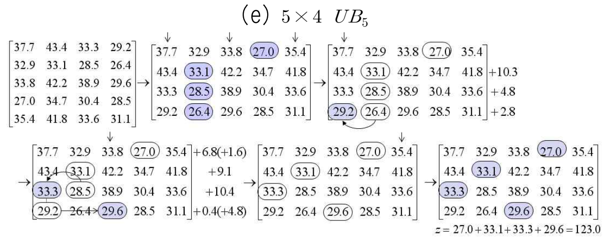
(c) 3×4 UB_3



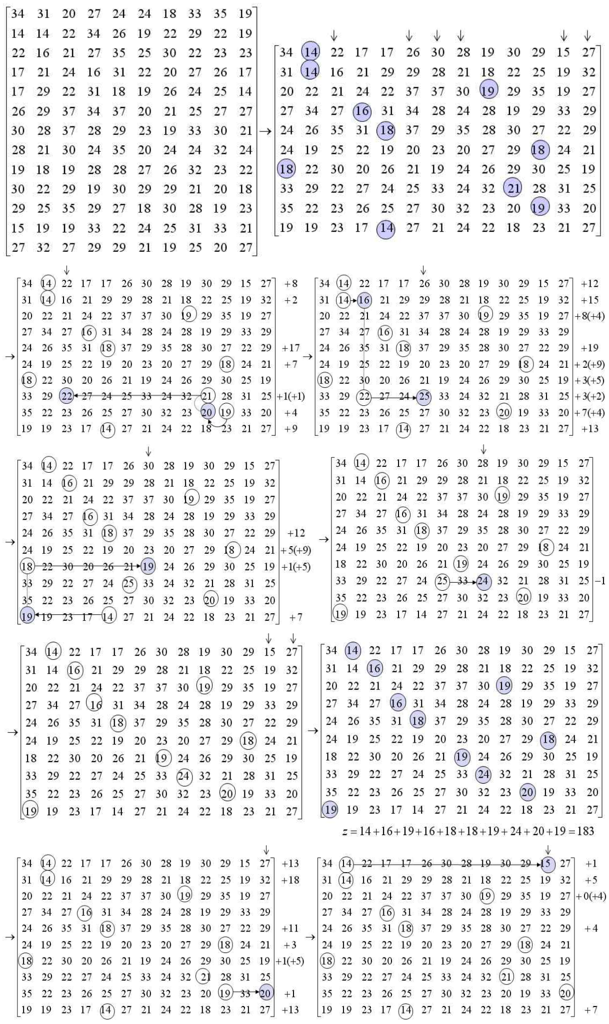
(d) 4×3 UB_4

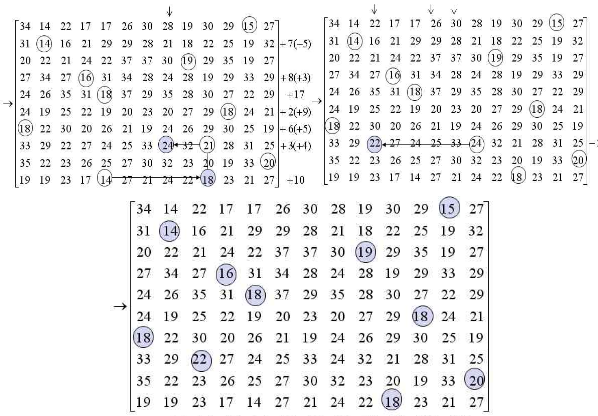


(e) 5×4 UB_5



(f) 5×4 UB_6





$z = 15+14+19+16+18+18+18+22+20+18 = 178$

(g) 13×10 UB_7

Fig. 6. MCM Algorithm for Unbalanced Assignment Problem

7개의 불균형 할당 문제에 대해 제안된 최적 할당 알고리즘은 모두 최적 해를 쉽게 찾는데 성공하였다. UB_6 는 Optimalon Software[25] 원문에서 $z=126.2$ 라 제시하였으나 제안된 알고리즘은 최적해 $z=123.0$ 을 얻었다. 또한, UB_7 은 Kinahan 와 Pryor[4] 원문에서 유전자 알고리즘을 적용한 결과 최적의 해법으로 $z=323$ 이라 제시하고 있다. 반면에 제안된 최적 할당 알고리즘은 최적해 $z=178$ 을 찾는데 성공하였다. 결국, 유전자 알고리즘으로 할당문제를 해결하려고 시도하였으나 실패하였음을 알 수 있다.

V. Conclusions

본 논문에서는 할당 문제의 최적 해를 쉽게 찾는 MCM 알고리즘을 제안하였다. 제안된 알고리즘은 각 행에서 최소 비용을 선택하고, 2개 이상 선택된 열에서 0개가 선택된 열로 최소비용으로 이동시킬 수 있는 경우 이를 이동시키는 단순한 방법을 적용하였다. 이 알고리즘은 배정과 조정의 2단계 수행 방법에서 조정 과정을 수행하지 않고도 최적 해를 얻을 수 있는 방법으로, 기존의 방법들과 차별성이 있다.

제안된 MCM 알고리즘을 균형 할당 27 문제와 불균형 할당 7 문제에 적용한 결과 모든 문제에 대해 최적 해를 찾는데 성공하였다.

제안된 알고리즘은 대표적인 헝가리안 알고리즘의 수행 복잡도 $O(n^3)$을 $O(n^2)$으로 개선하였으며, 균형 할당과 불균형 할당 문제에서 최적 해를 찾지 못하는 경우에 대해서도 최적 해를 쉽게 찾는 장점을 갖고 있다.

제안된 알고리즘은 다양한 균형과 불균형 할당문제들에 대해 행의 최소 비용 이동만으로 중복 선택된 열이 없이 n 쌍의 최적 해를 얻었다. 여기서, 중복 선택된 열을 제거하는 방법으로 열을 S, H, T의 3개 집합으로 분할하여 해결하였다. 추후 중복 선택된 열을 제거하는 보다 간단한 방법을 연구할 계획이다.

REFERENCES

[1] R. Burkard, M. Dell'Amico, and S. Martello, "Assignment Problems," SIAM., ISBN: 978-1-61197-222-1, 2012.

[2] H. W. Kuhn, “The Hungarian Method for the Assignment Problem," Naval Research Logistics Quarterly, Vol. 2, No. 1-2, pp. 83-97, Mar. 1955.

[3] L. Ntaimo, "Introduction to Mathematical Programming: Operations Research: Transportation and Assignment Problems", Vol. 1, 4th edition, by W. L. Winston and M. Venkataramanan, 2005.

[4] K. Kinahan and J. Pryor, "Algorithm Animations for Practical Optimization: A Gentle Introduction," http://optlab-server.sce.carleton.ca/POAnimations2007/Default.html, 2007.

[5] D. N. Kumar, "Optimization Methods," http://www.nptel.iitm.ac.in/Courses/Webcourse-contents/IISc-BANG/OPTIMIZATIONMETHODS/pdf/Module_4/M4L3_ LN.pdf, IISc, Bangalore, 2008.

[6] S. U. Lee, “Assignment Problem Algorithm Based on the First Selection Method of the Minimum Cost,” Journal of the IIBC, Vol. 13, No. 5, pp. 163-171, Oct. 2013.

[7] S. U. Lee, “A Reverse-delete Algorithm for Assignment Problem,” Journal of the KIIT, Vol. 10, No. 8, pp. 117-126, Aug. 2012.

[8] S. U. Lee, “The Optimal Algorithm for Assignment Problem,” Journal of the KSCI, Vol. 17, No. 9, pp. 139-147, Sep. 2012.

[9] S. U. Lee, “The Simplified Solution for Assignment Problem,” Journal of the IIBC, Vol. 12, No. 5, pp. 141-151, Oct. 2012.

[10] Rai Foundation Colleges, "Information Research," Bachelor of Business Administration, Business Administration, 2008.

[11] S. Noble, "Lectures 15: The Assignment Problem," Department of Mathematical Sciences, Brunel University, 2000.

[12] A. Dimitrios, P. Konstantinos, S. Nikolaos, and S. Angelo, "Applications of a New Network-enabled Solver for the Assignment Problem in Computer-aided Education," Journal of Computer Science, Vol. 1, No. 1, pp. 19-23, 2005.

[13] R. M. Berka, "A Tutorial on Network Optimization," http://home.eunet.cz/berka/o/English/networks/node

8.html, 1997.
[14] M. S. Radhakrishnan, "AAOC C222: Optimization," Birla Institute of Technology & Science, 2006.
[15] R. Burkard, M. D. Amico, and S. Martello, "Assignment Problems, SIAM Monographs on Discrete Mathematics and Applications, 2006.
[16] S. C. Niu, "Introduction to Operations Research," School of Management, The University of Texas at Dallas, 2004.
[17] W. Snyder, "The Linear Assignment Problem," Department of Electrical and Computer Engineering, North Carolina State University, 2005.
[18] M. E. Salassi, "AGEC 7123: Operations Research Methods in Agricultural Economics: Standard LP Form of the Generalized Assignment Problem," Department of Agricultural Economics and Agribusiness, Louisiana State University, 2005.
[19] K. Wayne, "Algorithm Design," http://www.cs.princeton.edu/~wayne/kleinberg-tardos/07assignment.pdf, 2005.
[20] J. Havlicek, “Introduction to Management Science and Operation Research,” http://orms.czu.cz/text/trans problem. html, 2007.
[21] R. Sedgewick and K. Wayne, " Computer Science 226: Data Structures and Algorithms, Princeton University, 2002.
[22] J. E. Beasley, "Operations Research and Management Science: OR-Notes," Department of Mathematical Sciences, Brunel University, West London, 2004.
[23] D. Doty, "Munkres' Assignment Algorithm: Modified for Rectangular Matrices," KCVU, Murray State University, Dept. of Computer Science and Information Systems, 2008.
[24] G. B. Dantzig, "Linear Programming and Extensions," USAF Project RAND, R-366-PR, The RAND Corporation, Santa Monica, California, U.S., 1963.
[25] Optimalon Software, "Transportation Problem (Minimal Cost)," http://www.optimalon.com/examples/transport.htm, 2008.

Authors

Sang Un Lee received the B. Sc. degree in avionics from the Korea Aerospace University in 1997. He received the M. Sc. and Ph. D. degrees in Computer Science from Gyeongsang National University, Korea, in 1997 and 2001, respectively. He is currently Professor with the Department of Multimedia Science, Gangneung-Wonju National University, Korea. He is interested in software quality assurance and reliability modeling, software engineering, software project management, neural networks, and algorithm.