http://dx.doi.org/10.7236/JIIBC.2013.13.5.163
JIIBC 2013-5-20

최소비용 우선선택 방법에 기반한 할당 문제 알고리즘

Assignment Problem Algorithm Based on the First Selection Method of the Minimum Cost

이상운*

Sang-Un, Lee

요 약 본 논문은 할당 문제의 최적해를 간단히 찾을 수 있는 알고리즘을 제안하였다. 일반적으로 할당 문제의 최적해는 Hungarian 알고리즘으로 구한다. 제안된 알고리즘은 Hungarian 알고리즘의 4단계 수행 과정을 2단계로 단축시켰다. 첫 번째로, 행렬의 최소 비용을 선택하고 행과 열의 값을 삭제하는 과정을 거쳐 초기 할당을 수행하였다. 두 번째로 할당을 조정하는 과정을 수행하였다. 제안된 알고리즘을 27개의 균형 할당 문제와 7개의 불균형 할당 문제에 적용한 결과 Genetic 알고리즘으로 찾지 못한 최적해를 찾는데 성공하였다. 따라서 제안된 알고리즘은 Hungarian 알고리즘을 대체하여 일반적으로 적용할 수 있을 것이다.

Abstract This paper proposes an algorithm that seeks the optimal solution for an assignment problem through a simplified process. Generally it is Hungarian algorithm that is prevalently used to solve a given assignment problem. The proposed algorithm reduces 4 steps Hungarian algorithm into 2 steps. Firstly, the algorithm selects the minimum cost from a matrix and deletes the rest of the rows and columns. Secondly, it improves on the solution through reassignment process. For 27 balanced assignment problems and 7 unbalanced problems, the proposed algorithm has successfully yielded the optimal solution, which Genetic algorithm has failed. This algorithm is thus found to be an appropriate replacement of Hungarian algorithm.

Key Words : Hungarian algorithm, Balanced assignment, Unbalanced assignment, Minimum cost, Optimal solution

Ⅰ. 서 론

할당 문제 (assignment problem)는 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 경우의 수송문제 (transportation problem)의 특수한 경우이다. 또한, 수송비용 (c_{ij})이 모두 다르며, 한 공급처에서 반드시 한 수요처로만 수송이 이루어져야만 한다. 이 경우 총 수송비용 합이 최소인 최적해 $z=\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}x_{ij}$를 찾도록 $x_{ij}=1$을 할당하는 문제이다.[1-3]

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

일반적으로 할당 문제는 Hungarian 알고리즘[1-3]을 적용하고 있으며, 일부는 유전자 알고리즘 (genetic

*정회원, 강릉원주대학교 과학기술대학 멀티미디어공학과
접수일자 : 2013년 03월 11일, 수정완료 : 2013년 9월 20일
게재확정일자 : 2013년 10월 11일

Received: 11 March, 2013 / Revised: 20 September, 2013 / Accepted: 11 October, 2013
*Corresponding Author: sulee@gwnu.ac.kr
Dept. of Multimedia Eng., Gangneung-Wonju National University, Korea

algorithm, GA)[4]을 시도하는 경우도 있다. Hungarian 알고리즘의 수행 복잡도는 $O(n^3)$으로 균형 할당 문제에 대해서는 최적해 (optimal solution)를 항상 찾을 수 있다고 알려져 있다.[1] 그러나 균형 할당문제에 대해서도 항상 최적해를 찾지 못하는 경우도 발생하여 최적해를 얻었는지 검증하는 과정이 필요하다. 또한, 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, Hungarian 알고리즘은 마지막으로 얻은 0의 비용들 중에서 각 열에서 중복되지 않게 1개씩만 선택하여야 하는 불편함이 따르며, 0을 포함하는 최소한의 선을 m개를 긋는 과정이 반복적으로 수행되어야만 한다.

본 논문은 비용 행렬의 크기와 무관하며, 균형과 불균형 할당 문제와 무관한 모든 경우에 대해 항상 최적해를 찾는 단순하면서도 일반적으로 적용할 수 있는 알고리즘을 제안한다.

2장에서는 할당 문제의 최적 해를 찾는 대표적인 Hungarian 알고리즘을 살펴보고 문제점을 고찰해 본다. 3장에서는 최적해를 간단히 찾는 할당 알고리즘을 제안한다. 4장에서는 제안된 알고리즘을 다양한 균형과 불균형 할당 문제 사례들에 적용하여 최적 해를 찾는지 평가해 본다.

II. 관련 연구와 연구 배경

할당 문제에서 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 최적해를 찾는다.

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

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

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

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

할당 문제의 최적 해를 찾는 Hungarian 알고리즘은 헝가리 수학자인 Harold Kuhn이 1955년에 제안하고, 1957년에 James Munkres가 보완한 4단계를 수행하는 방법이다. 따라서 할당 알고리즘 또는 Kuhn-Munkres 알고리즘이라 부르기도 한다.[2]

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

c_{ij}		Machine				공급량 (s_i)
		1	2	3	4	
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	

그림 1. A_1 할당 문제
Fig. 1. A_1 Assignment problem

Kumar[5]가 그림 1의 A_1 문제에 대해 Hungarian 알고리즘을 적용하여 최적해를 찾은 결과는 그림 2와 같다. 0을 모두 포함하는 최소한의 선을 m개 찾은 결과에 대해 작업을 $x_{11}=x_{23}=x_{32}=x_{44}=1$로 할당하여 최적해 $z=1+10+5+5=21$을 얻었다.

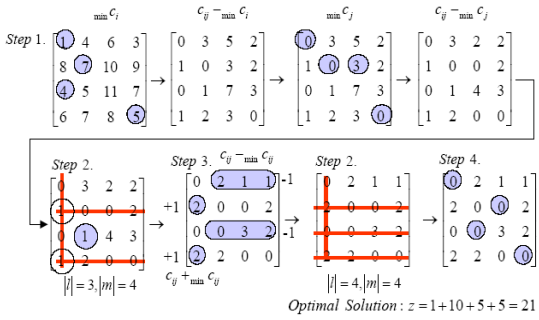


그림 2. A_1 할당문제에 대한 Hungarian 알고리즘
Fig. 2. Hungarian algorithm for A_1 assignment problem

Hungarian 알고리즘은 다음과 같은 문제점이 존재한다.

(1) 불균형 할당문제인 경우 부정확한 결과를 얻을 수 있어 가상 행이나 열을 추가하여 $m\times n(m=n)$인 균형 할당을 만드는 과정이 필요하다.

(2) 비용 행렬이 크고 0을 다수 포함하는 경우 0을 모두 포함하는 최소한의 라인을 긋는 방법이 어렵다.
(3) 최종적으로 얻은 0 값들에 대해 각 열을 기준으로 중첩되지 않게 1개씩만 선택하는 어려움이 있다.
(4) Hungarian 알고리즘의 수행 복잡도는 $O(n^3)$으로 큰 비용행렬인 경우 최적 해법을 도출하기 위해서는 많은 시간이 소요된다.

Ⅲ. 최소비용 우선선택 할당 알고리즘

본 장에서는 Hungarian 알고리즘의 수행 복잡도 $O(n^3)$을 $O(n)$으로 획기적으로 향상시키면서, 균형과 불균형 할당 문제 모두에 즉시 적용할 수 있는 할당 알고리즘을 제안한다.

제안된 알고리즘은 $m \times n$ 행렬의 최소 비용 $_{\min}c_{ij}$를 우선적으로 반복하여 선택하는 방법이다. 일단 첫 번째 $_{\min}c_{ij}$가 선택되면 해당 i행과 j열의 비용을 삭제한다. 나머지 행렬의 $i=0 \cap j=0$가 될 때까지 이 과정을 반복적으로 수행한다. 다음으로 검증 과정에서는 첫 번째 $_{\min}c_{ij}$에 대해 가능한 i행의 $_{\min}\{c_i - {}_{\min}c_{ij}\}$로 이동 또는 $_{\max}c_{ij}$를 가능한 i행의 $_{\max}\{{}_{\max}c_{ij} - c_i\}$로 이동시 비용을 감소시킬 수 있으면 할당을 변경한다. 이 알고리즘을 "최소비용 우선 선택 (minimum cost first selection, MCFS) 알고리즘"이라 하자. MCFS 할당 알고리즘은 그림 3에 제시되어 있다.

$m \times n$ 행렬의 최소 비용: $_{\min}c_{ij}$, 최대 비용: $_{\max}c_{ij}$.
[Step 1: 초기 할당]
while $i=0 \cap j=0$
  if j열에 $|_{\min}c_{ij}| \geq 2$ 존재 then
    $_{\min}\{2^{nd}{}_{\min}c_i - {}_{\min}c_{ij}\}$ 선택, i행 삭제, $i=i-1$, j열 삭제, $j=j-1$
  else if j열에 $|_{\min}c_{ij}|=1$ 존재 then $_{\min}c_{ij}$ 선택, i행 삭제, $i=i-1$, j열 삭제, $j=j-1$.
end
[Step 2: 검증]
$_{\min}c_{ij}(c_a)$를 가능한 $_{\min}\{c_i - {}_{\min}c_{ij}\}$인 c_b로 이동 or $_{\max}c_{ij}(c_a)$를 가능한 $_{\max}\{{}_{\max}c_{ij} - c_i\}$인 c_b로 이동.
c_b의 j열에 선택된 c_c를 다시 c_d로 이동.
if $c_d(j) \neq c_a(j)$ then c_d의 j열에 선택된 c_e를 다시 c_f로 이동
  if $c_f(j)=c_a(j)$ then
    if $\Sigma(-) > \Sigma(+)$ then $c_b \leftarrow c_a, c_d \leftarrow c_c$
else if $c_d(j)=c_a(j)$ then
  if $\Sigma(-) > \Sigma(+)$ then $c_b \leftarrow c_a, c_d \leftarrow c_c$.

그림 3. MCFS 할당 알고리즘
Fig. 3. MCFS assignment algorithm

그림 1의 A_1 할당문제에 MCFS 알고리즘을 적용한 과정은 그림 4에 제시되어 있다. A_1에서 $_{\min}c_{ij}$는 $(1,1)=1$로 나머지 1행과 1열을 모두 삭제한다. 다음으로 $_{\min}c_{ij}$을 만족하는 값은 $(3,2)=5$와 $(4,4)=5$가 차례대로 선택되면서 3행 4열과 4행 4열을 삭제하면 최종적으로 $(2,3)=10$이 선택된다. 검증단계에서는 첫 번째 $_{\min}c_{ij}$인 $(1,1)=1$을 2^{nd} $_{\min}c_i$인 $(1,4)=3$으로 (+3) 증가시키면 $(4,4)=5$는 2^{nd} $_{\min}c_i$인 $(4,1)=6$으로 +1이 증가되어 재조정이 불가하다. 결국, $z=1+10+5+5=21$을 간단하게 얻는데 성공하였다.

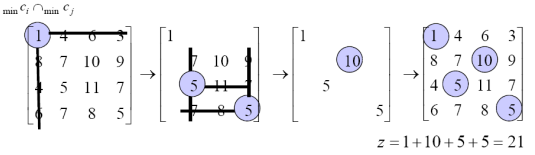


그림 4. A_1 문제의 MCFS 알고리즘 적용
Fig. 4. MCFS algorithm for A_1 assignment problem

제안된 알고리즘은 Hungarian 알고리즘의 4단계를 2단계로 단순화시켰다.

Ⅳ. 알고리즘 적용성 평가

본 장에서는 26개의 균형 할당 문제와 7개의 불균형 할당 문제를 대상으로 최적 할당 알고리즘의 적용성을 평가해 본다. 먼저, 그림 5에 제시된 26개의 균형 할당 문제에 대해 제안된 알고리즘을 적용하여 본다.

4×4 비용행렬은 B_1에서 B_{15}까지 15개 데이터이며, 5×5비용행렬은 B_{16}에서 B_{23}까지 8개 데이터, 6×6 비용행렬은 B_{24}에서 B_{25}까지 2개 데이터, 12×12비용행렬은 B_{26}의 1개 데이터로 구성되어 있다. B_1은 [6], B_2는 [7], B_3는 [8], B_4는 [9], B_5는 [10], B_6는 [11], B_7은 [12], B_8은 [6], B_9는 [13], B_{10}은 [3,14], $B_{11} \sim B_{15}$는 [6], B_{16},B_{17}은 [10], B_{18}은 [15], B_{19}는 [16], B_{20}은 [17], B_{21},B_{22}는 [6], B_{23}은 [18], B_{24}는 [19], B_{25}는 [6], B_{26}은 [20]에서 인용되었다.

$4\times4, 5\times5, 6\times6$과 12×12비용행렬에 대한 MCFS 알고리즘을 적용한 결과는 각각 그림 6, 그림 7, 그림 8과 그림 9에 제시되어 있다.

$$\begin{bmatrix}1&2&3&4\\2&4&6&8\\3&6&9&12\\4&8&12&16\end{bmatrix}(B_1)\quad \begin{bmatrix}5&2&3&4\\7&8&4&5\\6&3&5&6\\2&2&3&5\end{bmatrix}(B_2)\quad \begin{bmatrix}5&7&4&3\\5&5&4&3\\6&5&4&3\\5&5&6&6\end{bmatrix}(B_3)\quad \begin{bmatrix}7&5&8&2\\7&8&9&4\\3&5&7&9\\5&5&6&7\end{bmatrix}(B_4)\quad \begin{bmatrix}8&6&5&7\\6&5&3&4\\7&8&4&6\\6&7&5&6\end{bmatrix}(B_5)$$

$$\begin{bmatrix}8&2&3&3\\2&7&5&8\\0&9&8&4\\2&5&6&3\end{bmatrix}(B_6)\quad \begin{bmatrix}13&4&7&6\\1&11&5&4\\6&7&2&8\\1&3&5&9\end{bmatrix}(B_7)\quad \begin{bmatrix}6&12&3&7\\13&10&12&8\\2&5&15&20\\2&7&8&13\end{bmatrix}(B_8)\quad \begin{bmatrix}90&75&75&80\\35&85&55&65\\125&95&90&105\\45&110&95&115\end{bmatrix}(B_9)\quad \begin{bmatrix}14&5&8&7\\2&12&6&5\\7&8&3&9\\2&4&6&10\end{bmatrix}(B_{10})$$

$$\begin{bmatrix}15&29&35&20\\21&27&33&17\\17&25&37&15\\14&31&39&21\end{bmatrix}(B_{11})\quad \begin{bmatrix}18&26&17&11\\13&28&14&26\\38&19&18&15\\19&26&24&10\end{bmatrix}(B_{12})\quad \begin{bmatrix}20&10&15&22\\13&11&7&15\\16&12&10&15\\29&10&14&18\end{bmatrix}(B_{13})\quad \begin{bmatrix}20&25&22&28\\15&18&23&17\\19&17&21&24\\25&23&24&24\end{bmatrix}(B_{14})\quad \begin{bmatrix}42&35&28&21\\30&25&20&15\\30&25&20&15\\24&20&16&12\end{bmatrix}(B_{15})$$

$$\begin{bmatrix}3&9&2&3&7\\6&1&5&6&6\\9&4&7&10&3\\2&5&4&2&1\\9&6&2&4&5\end{bmatrix}(B_{16})\quad \begin{bmatrix}3&8&2&10&3\\8&7&2&9&7\\6&4&2&7&5\\8&4&2&3&5\\9&10&6&9&10\end{bmatrix}(B_{17})\quad \begin{bmatrix}3&8&9&15&10\\4&10&7&16&14\\9&13&11&19&10\\8&13&12&20&13\\1&7&5&11&9\end{bmatrix}(B_{18})\quad \begin{bmatrix}5&4&6&4&1\\2&5&4&10&5\\10&12&10&6&8\\1&3&4&2&6\\2&5&8&11&7\end{bmatrix}(B_{19})$$

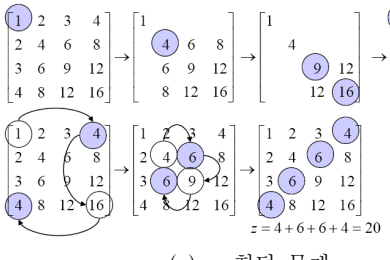
$$\begin{bmatrix}8&16&15&91&64\\83&42&93&75&27\\76&95&75&81&50\\20&42&96&90&24\\38&28&2&15&81\end{bmatrix}(B_{20})\quad \begin{bmatrix}30&37&40&28&40\\40&24&27&21&36\\40&32&33&30&35\\25&38&40&36&36\\29&62&41&34&39\end{bmatrix}(B_{21})\quad \begin{bmatrix}40&40&35&25&50\\42&30&16&25&27\\50&48&40&60&50\\20&19&20&18&25\\58&60&59&55&53\end{bmatrix}(B_{22})\quad \begin{bmatrix}22&30&26&16&25\\27&29&28&20&32\\33&25&21&29&23\\24&24&30&19&26\\30&33&32&37&31\end{bmatrix}(B_{23})$$

$$\begin{bmatrix}4&4&8&8&3&9\\4&6&4&1&4&6\\10&4&5&1&8&3\\2&7&2&6&7&2\\2&1&5&5&2&5\\2&8&3&1&2&1\end{bmatrix}(B_{24})\quad \begin{bmatrix}10&12&8&10&8&12\\9&10&8&7&8&9\\8&7&8&8&8&6\\12&13&14&14&15&14\\9&9&9&8&8&10\\7&8&9&9&9&8\end{bmatrix}(B_{25})$$

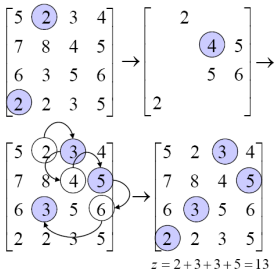
$$\begin{bmatrix}79&24&13&53&47&66&85&17&92&47&46&13\\43&59&33&95&55&97&34&55&84&94&26&56\\29&52&26&27&13&33&70&11&71&86&6&76\\88&83&64&72&90&67&27&47&83&62&35&38\\65&90&56&62&53&91&48&23&6&89&49&33\\44&79&86&93&71&7&86&59&17&56&45&59\\35&51&9&91&39&32&3&12&79&25&79&81\\50&12&59&32&23&64&20&94&97&14&11&97\\25&17&39&0&38&63&87&14&4&18&11&45\\68&45&99&0&94&44&99&5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

그림 5. 균형 할당 실험 데이터

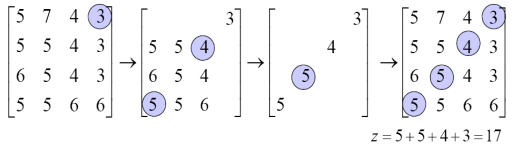
Fig. 5. Experimental data for balanced assignment problem



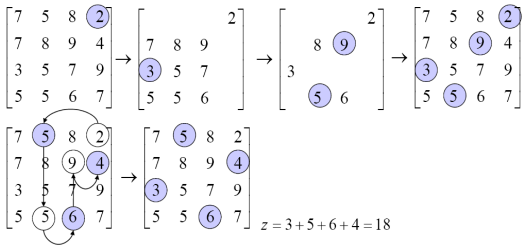
(a) B_1 할당 문제



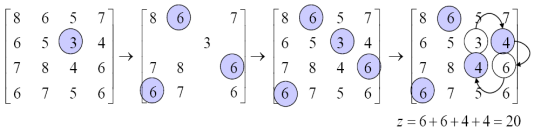
(b) B_2 할당 문제



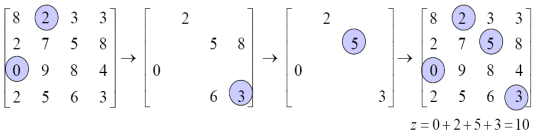
(c) B_3 할당 문제



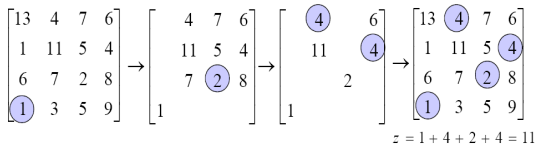
(d) B_4 할당 문제



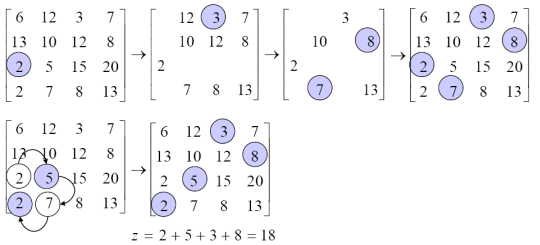
(e) B_5 할당 문제



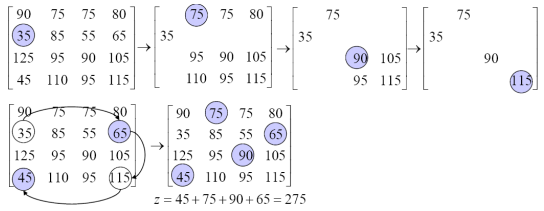
(f) B_6 할당 문제



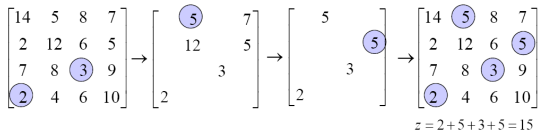
(g) B_7 할당 문제



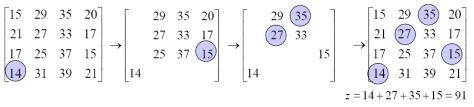
(h) B_8 할당 문제



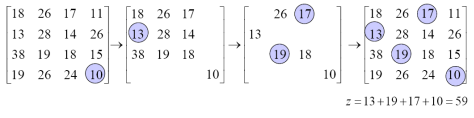
(i) B_9 할당 문제



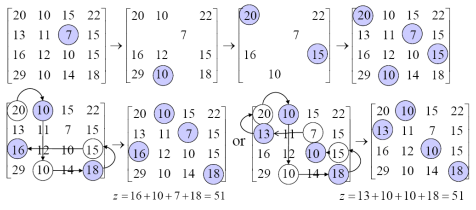
(j) B_{10} 할당 문제



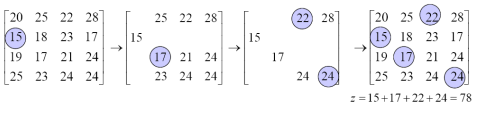
(k) B_{11} 할당 문제



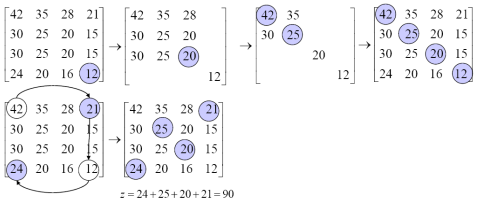
(l) B_{12} 할당 문제



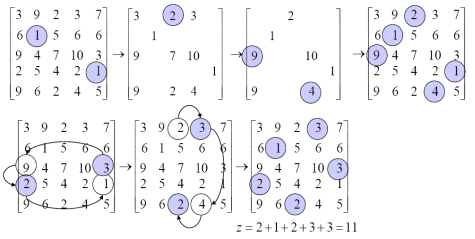
(m) B_{13} 할당 문제



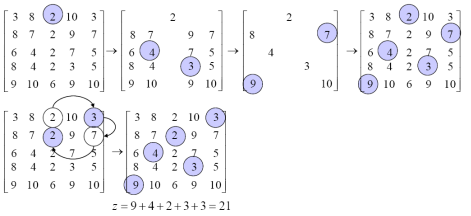
(n) B_{14} 할당 문제



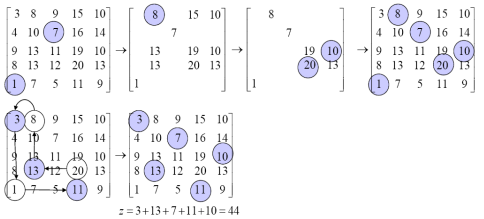
(o) B_{15} 할당 문제



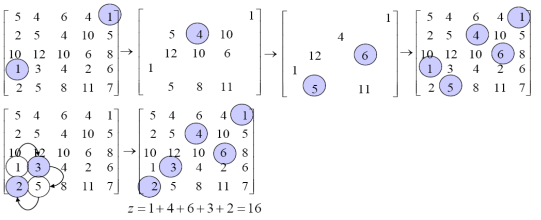
(a) B_{16} 할당 문제



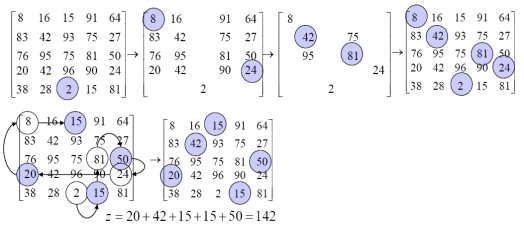
(b) B_{17} 할당 문제



(c) B_{18} 할당 문제

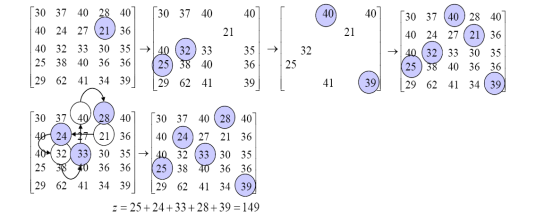


(d) B_{19} 할당 문제

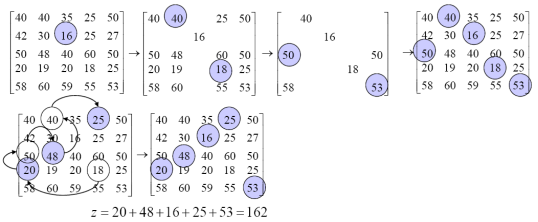


(e) B_{20} 할당 문제

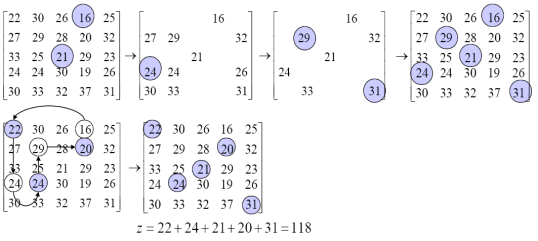
그림 6. 4×4 **할당문제의 MCFS 알고리즘 적용**
Fig. 6. MCFS algorithm for 4×4 **assignment problems**



(f) B_{21} 할당 문제



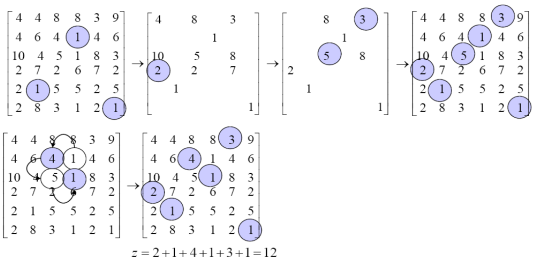
(g) B_{22} 할당 문제



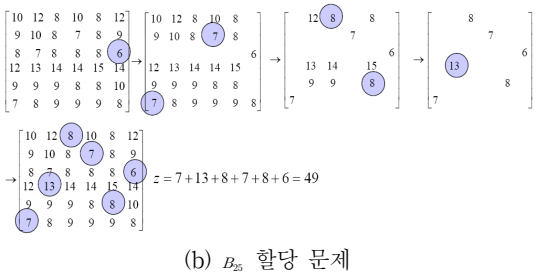
(h) B_{23} 할당 문제

그림 7. 5×5 **할당문제의 MCFS 알고리즘 적용**

Fig. 7. MCFS algorithm for 5×5 assignment problems



(a) B_{24} 할당 문제



(b) B_{25} 할당 문제

그림 8. 6×6 **할당문제의 MCFS 알고리즘 적용**

Fig. 8. MCFS algorithm for 6×6 assignment problems

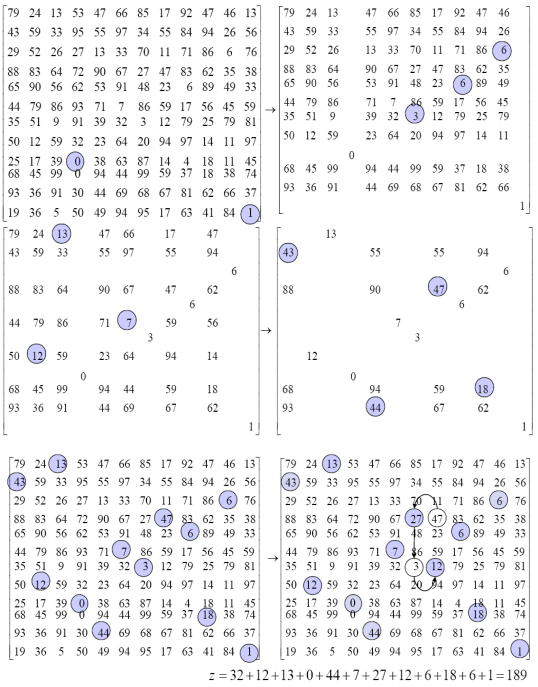


그림 9. 12×12 **할당문제의 MCFS 알고리즘 적용**

Fig. 9. MCFS algorithm for 12×12 assignment problem

다음으로 그림 10의 7개 불균형 할당 문제를 대상으로 MCFS 알고리즘의 적용성을 평가해 본다. UB_1은 [10]에서, UB_2는 [9,14,21]에서, UB_3, UB_4, UB_5는 [6]에서 UB_6은 [22]에서, UB_7은 [4]에서 인용되었다.

$$\begin{bmatrix}7&5&8&4\\5&6&7&4\\8&7&9&8\end{bmatrix}\ (UB_1)\qquad \begin{bmatrix}13&16&12&11\\15&0&13&20\\5&7&10&6\end{bmatrix}\ (UB_2)$$

$$\begin{bmatrix}20&25&22&28\\15&18&23&17\\19&17&21&24\end{bmatrix}\ (UB_3)\qquad \begin{bmatrix}60&80&50\\50&30&60\\70&90&40\\80&50&70\end{bmatrix}\ (UB_4)$$

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

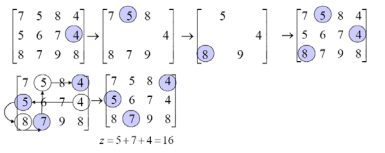
$$\begin{bmatrix}37.7&43.4&33.3&29.2\\32.9&33.1&28.5&26.4\\33.8&42.2&38.9&29.6\\37.0&34.7&30.4&28.5\\35.4&41.8&33.6&31.1\end{bmatrix}\ (UB_6)$$

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

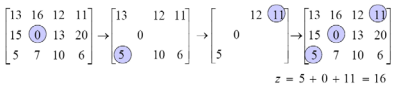
그림 10. 불균형 할당 실험 데이터

Fig. 10. Experimental Data for Unbalanced Assignment Problem

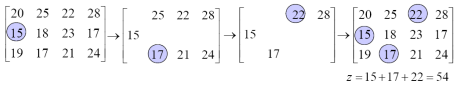
7개 불균형 할당 문제에 대해 최적 할당 알고리즘을 적용한 결과 3×4 비용행렬은 그림 11에, 4×3 비용행렬은 그림 12에, 5×4 비용행렬은 그림 13에, 13×10 비용행렬은 그림 14에 제시하였다.



(a) UB_1 할당 문제



(b) UB_2 할당 문제



(c) UB_3 할당 문제

그림 11. 3×4 할당문제의 MCFS 알고리즘 적용
Fig. 11. MCFS algorithm for 3×4 assignment problems

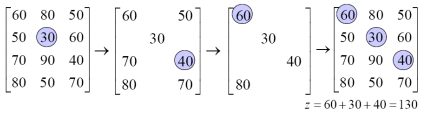
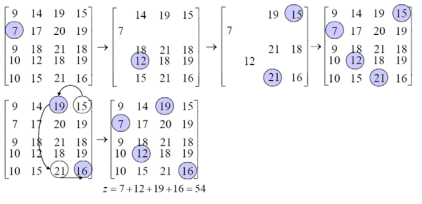
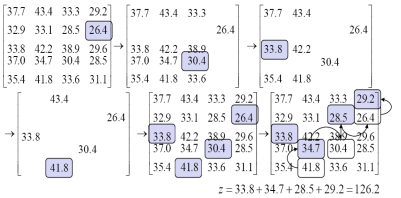


그림 12. 4×3 UB_4 할당문제의 MCFS 알고리즘 적용
Fig. 12. MCFS algorithm for 4×3 UB_4 assignment problem



(a) UB_5 할당 문제



(b) UB_6 할당 문제

그림 13. 5×4 할당문제의 MCFS 알고리즘 적용
Fig. 13. MCFS algorithm for 5×4 assignment problems

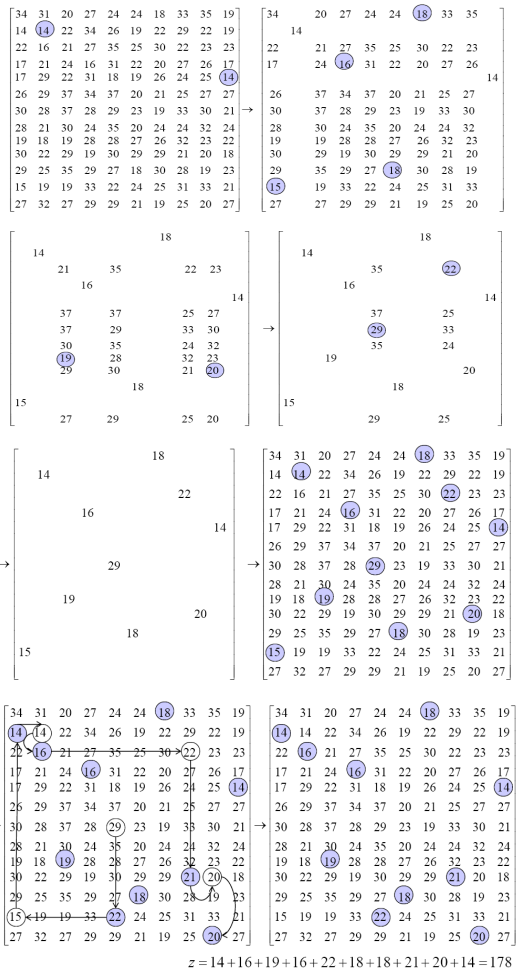


그림 14. 13×10 UB_7 할당문제의 MCFS 알고리즘 적용
Fig. 14. MCFS algorithm for 13×10 UB_7 assignment problem

26개 균형 할당 문제에 대해 제안된 MCFS 알고리즘은 모두 최적해를 찾았다. B_{26}은 Dantzig[20] 원문에서 최적의 해를 제시하지 않았다. 반면에 제안된 최적 할당 알고리즘은 최적 해 $z=189$를 찾을 수 있었다.

7개의 불균형 할당 문제에 대해 제안된 최적 할당 알고리즘은 모두 최적해를 쉽게 찾는데 성공하였다. UB_7은 Kinahan와 Pryor[4] 원문에서 유전자 알고리즘을 적용한 결과 최적의 해법으로 $z=323$이라 제시하고 있다. 반면에 제안된 최적 할당 알고리즘은 최적 해 $z=178$ 을 찾는데 성공하였다. 결국, 유전자 알고리즘으로 할당문제를 해결하려고 시도하였으나 실패하였음을 알 수 있다.

V. 결론

본 논문에서는 최소비용 우선 선택 방법을 적용한 할당 알고리즘을 제안하였다. 제안된 알고리즘은 행렬의 최소 비용을 선택하는 방법을 적용하여 초기 해를 구하고, 할당 값을 재조정하는 2단계를 수행하였다. 따라서 Hungarian 알고리즘의 4단계를 2단계로 단순화시켰으며, 불균형 할당 문제에 대해서도 가상의 행과 열을 추가하여 균형 할당으로 만드는 사전 작업이 필요하지 않다.

제안된 알고리즘을 균형 할당 26 문제와 불균형 할당 7 문제에 적용한 결과 모든 할당 문제에 대해 최적해를 찾는데 성공하였다. 따라서 주어진 문제에 대해 단순히 최소비용 선택 방법으로 최적해를 간단히 찾을 수 있어 Hungarian 알고리즘을 대체하여활용될 수 있을 것이다.

References

[1] Wikipedia, "Assignment Problem," http://en.wikipedia. org/wiki/Assignment_problem, Wikimedia Foundation Inc., 2013.

[2] Wikipedia, "Hungarian Algorithm," http://en.wikipedia. org/wiki/Hungarian_algorithm, Wikimedia Foundation Inc., 2013.

[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/POAnimations 2007 /Default.html, 2007.

[5] D. N. Kumar, "Optimization Methods," IISc, Bangalore, 2008.

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

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

[8] 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.

[9] R. M. Berka, "A Tutorial on Network Optimization," http://home.eunet.cz/berka/o/English/networks/node8.html, 1997.

[10] M. S. Radhakrishnan, "AAOC C222: Optimization," Birla Institute of Technology & Science, 2006.

[11] R. Burkard, M. D. Amico, and S. Martello, *"Assignment Problems*, SIAM Monographs on Discrete Mathematics and Applications, 2006.

[12] S. C. Niu, "Introduction to Operations Research," School of Management, The University of Texas at Dallas, 2004.

[13] W. Snyder, "The Linear Assignment Problem," Department of Electrical and Computer Engineering, North Carolina State University, 2005.

[14] 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.

[15] K. Wayne, "Algorithm Design," http://www.cs.princeton.edu/~wayne/kleinberg-tardos/07assignment.pdf, 2005.

[16] J. Havlicek, "Introduction to Management Science and Operation Research," http://orms.czu.cz/text/transproblem. html, 2007.

[17] R. Sedgewick and K. Wayne, " Computer Science 226: Data Structures and Algorithms, Princeton University, 2002.

[18] J. E. Beasley, "Operations Research and Management Science: OR-Notes," Department of Mathematical Sciences, Brunel University, West London, 2004.

[19] D. Doty, "Munkres' Assignment Algorithm: Modified for Rectangular Matrices," KCVU, Murray State University, Dept. of Computer Science and Information Systems, 2008.

[20] G. B. Dantzig, "Linear Programming and

Extensions," USAF Project RAND, R-366-PR, The RAND Corporation, Santa Monica, California, U.S., 1963.

[21] M. A. Trick, "Network Optimizations for Consultants, "http://mat.gsia.cmu.edu/mstc/networks/networks.html, 1996.

[22] Optimalon Software, "Transportation Problem (Minimal Cost)," http://www.optimalon.com/examples/transport.htm, 2008.

저자 소개

이 상 운(정회원)

- 1987년 : 한국항공대학교 항공전자공학과 (학사)
- 1997년: 경상대학교 컴퓨터과학과 (석사)
- 2001년 : 경상대학교 컴퓨터과학과 (박사)
- 2003년 : 강원도립대학 컴퓨터응용과 전임강사
- 2004년 ~ 2007년 2월 : 국립 원주대학 여성교양과 조교수
- 2007년 3월 ~ 현재 : 강릉원주대학교 멀티미디어공학과 부교수

<관심분야 :소프트웨어 프로젝트 관리, 개발 방법론, 소프트웨어 척도, 분석과 설계 방법론, 시험 및 품질보증, 소프트웨어 신뢰성, 신경망, 뉴로-퍼지, 그래프 알고리즘>

- e-mail : sulee@gwnu.ac.kr