Advanced Sensitivity Analysis of the Semi-Assignment Problem

Shih-Ting Zeng and Ue-Pyng Wen
Department of Industrial Engineering and Engineering Management
National Tsing Hua University, 101, Section 2, Kuang-Fu Rd., Hsinchu, 300, Taiwan

Chi-Jen Lin
Department of Industrial Engineering and Management
Ta Hwa Institute of Technology
No.1, Dahua Rd., Qionglin Township, Hsinchu County 307, Taiwan

Abstract
Sensitivity analysis of the semi-assignment problem (semi-AP) is concerned with obtaining the perturbed range of the cost that can be perturbed without the current positive variable set changing. Due to the high degeneracy of the semi-AP, the traditional sensitivity analysis, which decides the perturbed range where the current optimal basis remains optimal, is impractical. By using our proposed algorithm, the degenerate basic variable can be pivoted out, and the chosen basic variable can be pivoted in without changing the assigned cell. Concerning the high degeneracy of the semi-AP, we can broaden the perturbed range by allowing this basis changing, as long as maintaining the same positive variable set of the current problem. In this paper, we propose an algorithm for determining the perturbed range of the assigned and unassigned cell, and computational result is also provide to demonstrate the efficient of the proposed algorithm.

Keywords
Semi-assignment problem, sensitivity analysis, degeneracy

1. Introduction
The key assumption of the classic assignment problem is that the tasks and the agents to which they are to be assigned are unique; however, Pentico (2007) mentioned that although all the agents are unique, it is a common condition that some of the tasks are identical in a company or factory. Therefore, that kind of problem is more practical than the classic assignment problem. The problem for that situation is called the semi-AP, and the model for the situation where there are \( n \) agents should be assigned to \( m \) task groups, with \( d_i \) tasks in group \( i \). Considering the feasibility in the model, we assume that
\[
n = \sum_{i=1}^{m} d_i
\]

The model of the semi-AP is shown as model (1) (Barr 1976):

\[
\begin{align*}
\text{Min} & \quad \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}x_{ij} \\
\text{S.t} & \quad \sum_{j=1}^{n} x_{ij} = d_i \quad i = 1, ..., m, \\
& \quad \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, ..., n, \\
& \quad x_{ij} \geq 0,
\end{align*}
\]

where \( c_{ij} \) is the cost of assigning agent \( j \) to task group \( i \). Each group should be assigned to \( d_i \) agents, and each agent should be assigned to only one task. Or we can consider the problem as a production line worker allocation problem. Assume there are \( i \) production lines and each line should be assigned \( d_i \) workers, and each work should be assigned
to only one production line. Their objective functions are all to determine how all the assignment should be allocated to minimize the total cost.

Due to a semi-AP, which is a totally unimodular property of constraint matrix, it can be treated both as a linear programing, and also as a 0–1 integer programming problem or a network problem. With that property, some particular algorithms have been developed which can solve a semi-AP very efficiently. Since many methods for obtaining optimal solutions to the semi-AP have been proposed, sensitivity analysis is usually carried out for considering the uncertainty price fluctuation. With our algorithm we can keep the same assignment and still got the minimum cost but the object function change absolutely.

When degeneracy occurs, using the conventional sensitivity analysis may mislead the decision maker. For avoiding that problem, there are three definitions of sensitivity analysis ranges show up. According to Lin and Wen (2003) and Hadigheh and Terlaky (2006), we summarize three types of sensitivity analysis for linear Programming which is based on different propose.

Let $B$ be one of the optimal bases of the assignment problem with matrix $C$. Define $\omega^* = \{(i, j) \mid x_{ij} = 1, (i, j) \in B^*\}$ the corresponding optimal assignment which is also called support set.

- **Type I (Basics Invariancy):**
  Type I sensitivity is the traditional sensitivity analysis, as implemented in all commercial LP packages, and introduced in many operational research related text books. We denote the perturbed range $\Delta c_{pq}$ in (2-1) as $[L_{pq}, U_{pq}]$ such that $B^*$ remains the same. Type I sensitivity analysis uses the simplex method and is based on the nondegeneracy assumption of the optimal solution. It is worthwhile to mention that when a problem has multiple optimal or/and degenerate solutions, then depending on the basics invariancy make no sense on optimality ranges, and thus bewilder the policy-maker.

- **Type II (Support Set Invariancy):**
  Taking multiple optimal or(and) degenerate solutions into consideration, Type II sensitivity analysis deals with finding a perturbed range $[L_{pq}, U_{pq}]$ such that maintain the same $\omega^*$ (support set). It means that the perturbed problem should have an optimal solution with the same support set.

- **Type III (Optimal Partition Invariancy):**
  In this type of sensitivity analysis, we want to determine those values of model parameters for which the rate of change of the optimal objective function value is constant.

In this paper, we investigate Type II sensitivity analysis for the semi-AP. Degeneracy of optimal solutions causes considerable difficulties in sensitivity analysis by using traditional sensitivity analysis method. Therefore, we want to develop advanced sensitivity analysis of the semi-AP to solve the problem.

Also, there are many successful cases extend the theory to practical in the semi-AP. According to the study of Duffuaa et al. (1994), he mentioned that in the well-known car company GM, semi-assignment formulations are used to assigned microprocessors to tasks or functions that means a processor can handle more than one task. Besides, the semi-AP can also be applied to manpower planning (assigning personnel to jobs), scheduling (assigning subassemblies to tasks) quoting from Pentico (2007).

For solving those practical problems, there are many different algorithms developed. The first algorithm specially designed for solving the semi-AP was developed by Barr et al (1976) and was called the alternating basis method. The major difference concept between the alternating basis method algorithm and the primal simplex method is that it considers a subset of bases, called alternating path bases, for the purpose to obtain an optimal solution. Then, Duffuaa et al. (1994) claimed the total solution time of the algorithm developed by them called a shortest augmenting path algorithm is faster than alternating path bases method. Shortest augmenting path algorithm is developed to maintain dual feasibility and complementary slackness and works toward satisfying primal feasibility. Volgenant (1996) developed a modify algorithm base on shortest augmenting path procedure and said the code is superior to other codes for the semi-AP. Some assignment problem can be solved by decomposition to the semi-AP such as a vehicle routing problem described by Toth and Vigo (2002). Moreover, some approaches get the ranges which the optimal assignment remains optimal by considering all optimal basis such as (Gal 1986), but the method is troublesome and tedious.
Many of methods investigating the solution procedure, and very few sensitivity analysis methods are proposed.

2. The Advanced Sensitivity Analysis Algorithm

In the section, we propose two theories for simplifying our algorithm procedure and demonstrate our algorithm.

2.1 Type II Sensitivity Analysis of the Semi-AP

Theorem 1 shows that the lower bound of Type II range for an assigned cell is unbounded in the semi-AP, and the Theorem 2 shows the upper bound of Type II Range for an unassigned cell is also unbounded in the semi-AP. Before we start to prove the theorem, we need to know that if the optimal solution of a dual problem is infeasible, then it optimal solution of the primal problem is unbounded proposed by Bazaraa et al. (2010).

**Theorem 1**

Suppose $\omega^*$ is the current optimal assignment of $C$ as model (2), and $\bar{C}$ as model (3) is the perturbed matrix as follows. If $(p, q) \in \omega^*$, then the lower bound of Type II Range $L_{pq}^{II} \to -\infty$.

\[
C = \begin{bmatrix}
    c_{11} & c_{12} & \cdots & c_{1n} \\
    c_{21} & c_{22} & \cdots & c_{2n} \\
    \vdots & \vdots & \ddots & \vdots \\
    c_{n1} & c_{n2} & \cdots & c_{nn}
\end{bmatrix}
\]

(2)

\[
\bar{C} = \begin{bmatrix}
    c_{11} & \cdots & c_{1q} & \cdots & c_{1n} \\
    \vdots & \ddots & \ddots & \vdots & \vdots \\
    c_{p1} & \cdots & c_{pq} + \Delta c_{pq} & \cdots & c_{pn} \\
    \vdots & \ddots & \ddots & \ddots & \vdots \\
    c_{n1} & \cdots & c_{nq} & \cdots & c_{nn}
\end{bmatrix}
\]

(3)

**Proof:** Theorem 1 is proved with model (4) and model (5), and $L_{pq}^{II}$ is the optimal of model (2)

\[
\begin{align*}
\text{Min} & \quad \Delta c_{pq} \\
\text{s.t.} & \quad u_i + v_j + Q_{ij} = c_{ij} \quad \text{if } (i, j) \neq (p, q) ; \quad i = 1, \ldots, m, j = 1, \ldots, n \\
& \quad u_p + v_q + Q_{pq} - \Delta c_{pq} = c_{pq} \quad ; \quad (p, q) \in \omega^* \\
& \quad Q_{ij} = 0 \quad \text{if } (i, j) \in \omega^* \quad ; \quad i = 1, \ldots, m, j = 1, \ldots, n \\
& \quad Q_{ij} \geq 0 \quad \text{if } (i, j) \notin \omega^* \quad ; \quad i = 1, \ldots, m, j = 1, \ldots, n .
\end{align*}
\]

(4)

Model (5) is the dual problem of model (4)
Max $\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} f_{ij}$

s.t. $\sum_{j=1}^{n} t_{ij} = 0 \quad i=1,2,...,m$

$\sum_{i=1}^{m} t_{ij} = 0 \quad j=1,2,...,n$

$-t_{pq} = 1$

$-t_{ij} \leq 0 \quad \text{if} (i,j) \not\in \Omega^* ; i,j = 1,...,n.$ \hspace{1cm} (5)

Since $t_{pq} = -1$ and $t_{pj} \leq 0, j \neq q$; one can induce $\sum_{j=1}^{n} t_{pj} < 0$. This contradicts with the constraint $\sum_{j=1}^{n} t_{pj} = 0$. Hence, model (5) is infeasible, and we can conclude that model (4) is unbounded. The proof has completed.

**Theorem 2** Suppose $\Omega^*$ is the current optimal assignment of $C$ and $\widetilde{C}$ is the perturbed matrix. If $(p, q) \not\in \Omega^*$, then the upper bound of Type II Range $U_{pq}^{II} \to \infty$.

**Proof:** Theorem 2 is proved with model (6) and model (7), where $U_{pq}^{II}$ is the optimal value of model (6):

Max $\Delta c_{pq}$

s.t. $u_{i} + v_{j} + Q_{ij} = c_{ij} \quad \text{if} (i,j) \neq (p, q) ; i=1,...,m, j=1,...,n$

$u_{i} + v_{j} + Q_{pq} - \Delta c_{pq} = c_{pq} \quad \text{if} (p, q) \notin \Omega^*$

$Q_{ij} = 0 \quad \text{if} (i,j) \in \Omega^* ; i=1,...,m, j=1,...,n$

$Q_{ij} \geq 0 \quad \text{if} (i,j) \notin \Omega^* ; i=1,...,m, j=1,...,n.$ \hspace{1cm} (6)

Model (7) is the dual problem of (6).

Min $\sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} t_{ij}$

s.t. $\sum_{j=1}^{n} t_{ij} = 0 \quad i=1,2,...,m$

$\sum_{i=1}^{m} t_{ij} = 0 \quad j=1,2,...,n$

$-t_{pq} = 1$

$t_{ij} \geq 0 \quad \text{if} \ (i,j) \notin \Omega^* ; i=1,...,m, j=1,...,n.$ \hspace{1cm} (7)

Since equation $t_{pq} = -1$ contradicts with $t_{pq} \geq 0$ if $(p, q) \notin \Omega^*$, problem model (7) is infeasible. Hence, problem model (6) is unbounded.

After proving the two theorems, we can conclude that the lower bound for an assigned cell and the upper bound for an unassigned cell are infinite. Hence, we just need to calculate the upper bound of an assigned cell and the lower bound of an unassigned cell in our algorithm. That can simplify our procedure.
2.2 The Proposed Algorithm
When we want to calculate a Type II range of a semi-AP through the algorithm, we should input a matrix $\mathbf{Q}^{\text{new}}$. The input matrix is modified from $\mathbf{Q}$ which represents current optimal reduce costs ($u_i + v_j - c_{ij}$) of an optimal problem, and it is shown at model (8). After we use $-1$ to replace the reduce cost of an assigned cell $(i, j)$, $\mathbf{Q}$ is transferred to a new matrix $\mathbf{Q}^{\text{new}}$ which is shown as model (9).

$$
\mathbf{Q} = \begin{bmatrix}
2 & 0 & 1 \\
1 & 1 & 0 \\
0 & 0 & 3 \\
0 & 5 & 5 \\
0 & 5 & 2 \\
0 & 3 & 0
\end{bmatrix} \tag{8}
$$

$$
\mathbf{Q}^{\text{new}} = \begin{bmatrix}
2 & -1 & 1 \\
1 & 1 & -1 \\
-1 & 0 & 3 \\
-1 & 5 & 5 \\
-1 & 5 & 2 \\
0 & 3 & -1
\end{bmatrix} \tag{9}
$$

The algorithm consists of three phases: perturbing phase, labeling phase and modify phase, and the procedure is shown on Figure 1. First, we explain the purpose of the four phases. After that, we talk about the detail steps in each phase.

Let $q_{ij}$ denotes elements of $\mathbf{Q}^{\text{new}}$ matrix in row $i$ and column $j$, and let a parameter $t=1$ to denote cell $(i, j)$ is an assigned cell and using $t=0$ to denote cell $(i, j)$ is an unassigned cell to make the algorithm be coded easily.

Here, we explain the following phase by let cell $(p, q)$ as a perturbed cell.

- **The perturb phase:** It occurs when a cell’s category is equal to one. Suppose $t$ is equal to one and $\Delta c_{pq}$ is a perturbed variation we’ve mentioned before. After some modify of reduce cost for feasibility, we set all $q_{iq} - \Delta c_{pq} \geq 0 \ (i \notin p)$ for the optimal condition, and we have equations $\Delta c_{pq} \leq q_{iq} \ (i \notin p)$. Derive intersection of these inequalities, we got a upper bound occurs at cell $(i, q) = \text{Min} \{q_{iq} | i \notin p\}$ Then we let $p \leftarrow i$, and decide it as the pivot cell $(p, q)$. Hence, we can get the same pivot cell by choosing the minimum reduce cost of all unassigned cells of column $j$ directly.

- **The labeling phase:** It is used to find the leaving or entering loop, and the detail is at below. Let $\mathbf{R}$ matrix to denote if the row is marked or not, and $r_i$ is the elements of $\mathbf{R}$ matrix in row $i$. $C$ and $c_j$ are used for the column in the same purpose.
  
  - Let $r_p = 1$
  - If $r_i = 1 \ (i=1, \ldots, m)$, let $b \leftarrow i$. Then, let $c_j = 1$ if $q_{bj} = -1 \ (j=1, \ldots, n)$. Breaking here, if there are no more rows can be labeled
  - If $c_j = 1 \ (j=1, \ldots, n)$, let $a \leftarrow j$. Then, let $r_i = 1$ if $q_{ia} = 0 \ (i=1, \ldots, m)$. Breaking here, if there are no more columns can be labeled.
The modify phase: When we cannot find a loop after performing labeling phase, the modify phase occurs.

A matrix $M$ is given to denote how many times a cell has been labeled, and modifying the $Q^{\text{new}}$ as the idea of Hungarium algorithm. $m_{ij}$ represent elements of $M$ matrix in row $i$ column $j$. The procedure is shown below.

- If $r_i = 1$ ($i=1,...,m$), let $b \leftarrow i$ and $m_{bj} \leftarrow m_{bj} + 1$.
- If $c_j = 0$ ($j=1,...,n$), let $a \leftarrow j$ and $m_{ia} \leftarrow m_{ia} + 1$.
- $s = \min \{q_{ij}^{\text{new}} \mid m_{ij} = 0\}$ ($s$ is a parameter used for record the minimum number of all $q_{ij}$ under the condition the matrix $M$ of a cell $(i, j)$ is equal to zero.)
- If $m_{ij} = 0$, let $q_{ij}^{\text{new}} \leftarrow q_{ij}^{\text{new}} - s$. ($i=1-m, j=1-n$) If $m_{ij} = 2$, let $q_{ij} \leftarrow q_{ij} + s$. ($i=1-m, j=1-n$)

After we describe the specific content of the four phases, we start to explain the procedure of the algorithm which is given in Figure 1. There are four main steps included in the procedure as the following
• **Labeling procedure for determine the type II range of the semi-assignment problem**
  
  o  **Step1:** We input the data $Q^{\text{new}}$, and distinguishing their $t$ of each cell $(i, j)$. If $t$ is equal to one, go to step2. If $t$ is equal to zero go to step3.
  
  o  **Step2:** Perform the perturb phase and got the pivot cell $(p, q)$, then go to step3.
  
  o  **Step3:** Perform the algorithm phase. If $c_q$ is equal to one, we got the Type II range here, and the range is $[-\infty, q_{pq}]$, when $t$ is equal to one. While the range is $[q_{pq}, \infty]$ for $t$ is equal to zero. If $c_q$ is not equal to one, go to step4.
  
  o  **Step4:** Perform Modify phase and go to Step2 if $t$ is equal to one. If $t$ is equal to zero, go to step3.

3. **Computational Results**

In this chapter, the difference perturbed range of Type I and Type II is demonstrated and running time of 12 cases by using the algorithm is displayed.

3.1 **The Comparison between the Type I and Type II Ranges**

Type II range can be larger or equal than Type I range as we mentioned before, and the two different ranges is compared for support our view. Here, Type I ranges are obtained by using Lingo package software, and Type II ranges are acquire by using the proposed algorithm with the same cost coefficients value and RHS values. The Type I and Type II ranges is shown at Table 1.

<table>
<thead>
<tr>
<th>Table 1: Type I and Type II Ranges</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type I</td>
</tr>
<tr>
<td>--------</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Regarding to the result, all Type II ranges are all bigger or equal to Type I ranges without exception, and the upper bound and the lower bound is infinite in unassigned cell and assigned cell separately as the two theories we proposed.

3.2 **The Result of Test Problem**

Twelve cases are generated with the combination of four different problem size and three cost intervals. Results are given for problem sizes $50 \times 200, 100 \times 200, 150 \times 200$ and $200 \times 200$, and cost coefficients is decided randomly between cost intervals 1-50, 1-100 and 1-1000. The average times, and standard deviations are average of ten problems in a single case. The results are given as Table 2.

Table 2 shows the each average time of three problems in the same problem size are substantially no different; however, we still can find the average time in 1-50 cost ranges is relatively smaller than the wilder one in almost cases.

The reason is the smaller cost ranges is more likely to cause alternative solutions which result in more zero value reduce costs; therefore the average time of a case in smaller cost ranges is more likely faster than others, because the cycle can be found more easily. Figure 2 is given as follows to display the trend of the average time more clearly.
Table 2: The average time (min) of test problems

<table>
<thead>
<tr>
<th>Problem Size (number of rows)</th>
<th>Cost Ranges</th>
<th>1-50</th>
<th>1-100</th>
<th>1-1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td></td>
<td>1.18</td>
<td>1.18</td>
<td>1.33</td>
</tr>
<tr>
<td>100</td>
<td></td>
<td>4.16</td>
<td>4.19</td>
<td>4.80</td>
</tr>
<tr>
<td>150</td>
<td></td>
<td>6.64</td>
<td>6.81</td>
<td>6.42</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>8.22</td>
<td>8.39</td>
<td>9.33</td>
</tr>
</tbody>
</table>

Then, the standard deviation is given in Table 3 to show the dispersion exists from the average. The trend of standard deviation from the smaller problem size to the bigger problem size becomes higher than the prior one in almost cases. That means there are more uncertainty in the bigger size problem such as the size of the cycle we find and the number of modifying times needed for finding a cycle.

Table 3: The standard deviation of test problems

<table>
<thead>
<tr>
<th>Problem Size (number of rows)</th>
<th>Cost Ranges</th>
<th>1-50</th>
<th>1-100</th>
<th>1-1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td></td>
<td>0.13</td>
<td>0.08</td>
<td>0.12</td>
</tr>
<tr>
<td>100</td>
<td></td>
<td>0.11</td>
<td>0.06</td>
<td>0.21</td>
</tr>
<tr>
<td>150</td>
<td></td>
<td>0.21</td>
<td>0.25</td>
<td>0.34</td>
</tr>
<tr>
<td>200</td>
<td></td>
<td>0.20</td>
<td>0.28</td>
<td>0.55</td>
</tr>
</tbody>
</table>

Figure 2: The trend of average time in each problem size
4. Conclusions
This paper concentrates on the sensitivity analysis of the semi-AP. Considering the special structure of the semi-AP, only the cost coefficients perturbed ranged are concerned. Moreover, the semi-AP is a highly degeneracy problem as all the other network problem, and hence, the traditional sensitivity analysis cannot be applied to the problem. The algorithm we proposed here can attain the Type II range without the need of reduce costs of all basis by using the revised labeling method.

In this study, we perturb only one cost coefficient at a time that is suitable when the change of a cost coefficient is about the agent or workers. however, the cost changing of a production line may occurs by the renovation or totally machine update of one production line that may result in cost coefficients changing in the same degree in one production line. Therefore, the algorithm can be extended by considering how to perturb one row at a time.

References