〈論　文〉

Modified Balanced Assignment Problem in Vector Case

– An Attempt to Apply the Method for 2-dimensional Case to 3-dimensional –

Yuusaku Kamura

Abstract We consider a combinatorial problem that arises from systems' construction. Each system consists of many components, and every component has its own error expressed in a vector. It is required to make the combination that minimizes as small as possible the difference between the maximum error and the minimum one. It means the vector case's balanced optimization problem. In this paper, we deal with the case that each system consists of 2 components and the vector dimension is 3. We apply the approximation algorithm that we have presented for 2-dimensional case to 3-dimensional problem and we show the numerical experiments.

Keywords optimization; approximation algorithm; assignment problem; combinatorial problem

1. Introduction

Suppose we have n suppliers $u_1, u_2, \ldots, u_n$ (to be denoted by a set U) and n customers $v_1, v_2, \ldots, v_n$ (to be denoted by V). If supplier u_i and customer v_j are chosen, the cost is c_{ij}. We are going to determine a one to one correspondence $\pi : U \to V$ under an appropriate objective function. This is the well-known assignment problem.

Since making the one to one correspondence $\pi : U \to V$ is equivalent to permuting the set $\{1, 2, \ldots, n\}$, we hereafter call the one to one correspondence as permutation.

The assignment problem has various versions with respect to the objectives [1]. If we are going to minimize the total sum of the cost $\sum_{i=1}^{n} c_{i\pi(i)}$, the problem is called the *linear sum assignment problem*, and there have been proposed many efficient algorithms [2].

If we are going to minimize the difference of the maximum cost and the minimum one of the corresponded pair, the problem is called the *balanced assignment problem* [3]. The objective is

$$\min_{\pi}\{\max_{1\le i\le n} c_{i\pi(i)} - \min_{1\le i\le n} c_{i\pi(i)}\}\,. \tag{1}$$

The polynomial time algorithms have been proposed for this problem, too.

In this paper we extend the balanced assignment problem to the case that costs are multidimensional, i.e., vectors. Here we assume that cost vector $\boldsymbol{c}_{ij}$ is represented as a sum of the supplier vector $\boldsymbol{a}_i$ and the customer vector $\boldsymbol{b}_j$.

A formal description of our problem is as follows. Let A and B be sets of m dimensional vectors. We denote each element of A by $\boldsymbol{a}_i = (a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(m)})$ and each element of B by $\boldsymbol{b}_i = (b_i^{(1)}, b_i^{(2)}, \ldots, b_i^{(m)})$. We assume that $a_i^{(1)}, a_i^{(2)}, \ldots, a_i^{(m)}$ and $b_i^{(1)}, b_i^{(2)}, \ldots, b_i^{(m)}$ are nonnegative and define the sum of vectors $\boldsymbol{a}_i + \boldsymbol{b}_j$ as

$$\boldsymbol{a}_i + \boldsymbol{b}_j = (a_i^{(1)} + b_j^{(1)}, a_i^{(2)} + b_j^{(2)}, \ldots, a_i^{(m)} + b_j^{(m)}). \tag{2}$$

We denote this vectors' sum as $\boldsymbol{c}_{ij}$, namely $\boldsymbol{c}_{ij} = \boldsymbol{a}_i + \boldsymbol{b}_j$. Let us consider the following problem: find a permutation π of a set $\{1, 2, \ldots, n\}$ such that

$$\begin{aligned} T_m(\pi) = \max\{ \ & \max\nolimits_{1\le i\le n} c_{i\pi(i)}^{(1)} - \min\nolimits_{1\le i\le n} c_{i\pi(i)}^{(1)}, \\ & \max\nolimits_{1\le i\le n} c_{i\pi(i)}^{(2)} - \min\nolimits_{1\le i\le n} c_{i\pi(i)}^{(2)}, \\ & \vdots \\ & \max\nolimits_{1\le i\le n} c_{i\pi(i)}^{(m)} - \min\nolimits_{1\le i\le n} c_{i\pi(i)}^{(m)} \ \} \end{aligned} \tag{3}$$

is the minimum. $T_m(\pi)$ means the maximum of m differences

$$\max_{1\le i\le n} c^{(k)}_{i\pi(i)} - \min_{1\le i\le n} c^{(k)}_{i\pi(i)}, \tag{4}$$

where $1 \le k \le m$.

Such a problem appears when we combine industrial products. Each system consists of many individuals. The lens for a semiconductor manufacturing system is such a product. That includes many lenses (about 30) and the error of each one is given by the high dimensional vector.

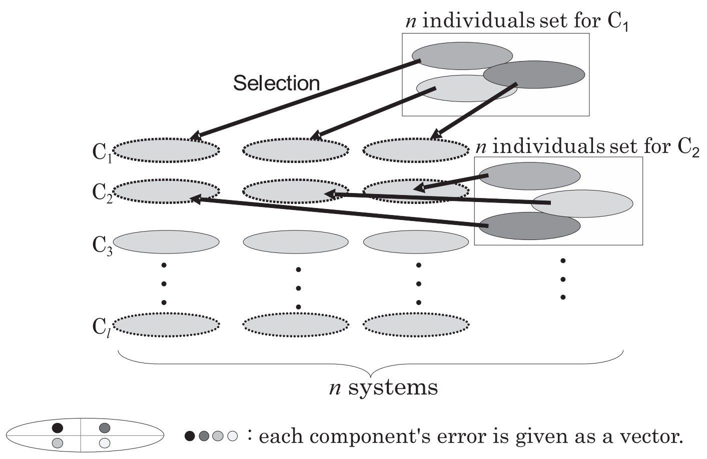
n individuals set for C$_1$
Selection
n individuals set for C$_2$
C$_1$
C$_2$
C$_3$
C$_l$
n systems
●●●○ : each component's error is given as a vector.

Fig. 1. System construction. A system consists of l components C_i $(i = 1, 2, \dots, l)$. Each component is given by the lot.

It is required to minimize the variation range of systems' errors. That problem is generally formulated to a multi-index balanced assignment problem. But it is shown that even a multi-index linear sum assignment in scalar case is NP-complete. Therefore for the vector case's balanced assignment problem, we deal with the problem in a single-index, that is, the combination between 2 sets of n vectors.

From the above assumptions, our problem is formulated as follows:

Problem 1 Given 2 sets of n vectors A and B, find the permuation π that minimizes

$$T_m(\pi) = \max_{1\le k\le m} \{\max_i c^{(k)}_{i\pi(i)} - \min_i c^{(k)}_{i\pi(i)}\}, \tag{5}$$

where $c^{(k)}_{i\pi(i)} = a^{(k)}_i + b^{(k)}_{\pi(i)}$.

In our former research [4], [5], [6], [7] we considered the problem of making n sets of system such that the worst combined error is the minimum. On the other hand, in [8] we dealt with the balanced assignment problem in 2-dimensional vector case. In this paper we attempt to apply our algorithm in [8] to the problem in 3-dimensional vector case.

Remark In general assignment problems, edges' costs are given independently. However, in our problem they are introduced by the vertices' costs. Hence we call this the 'modified' problem.

2. Formulation as an integer programming problem

Problem 1 is formulated to the following 0-1 integer programming problem. Here M is a sufficient big number.

Problem 2

Minimize $f = t$

subject to

$$u^{(k)} - l^{(k)} \le t, \qquad (k = 1, 2, \dots, m);$$

$$\begin{array}{ll}
c_{ij}^{(k)} x_{ij} \leq u^{(k)}, & (i,j=1,2,\ldots,n; k=1,2,\ldots,m);\\
l^{(k)} \leq c_{ij}^{(k)} x_{ij} + M(1-x_{ij}), & (i,j=1,2,\ldots,n; k=1,2,\ldots,m);\\
\displaystyle\sum_{i=1}^{n} x_{ij} = 1, & (j=1,2,\ldots,n);\\
\displaystyle\sum_{j=1}^{n} x_{ij} = 1, & (i=1,2,\ldots,n);\\
l^{(k)} \geq 0, \quad u^{(k)} \geq 0, & (k=1,2,\ldots,m);\\
x_{ij} \in \{0,1\}, & (i,j=1,2,\ldots,n).
\end{array}$$

Unlike the scalar case's assignment problem, no polynomial time algorithm exists to obtain the integer solution of this problem, and unfortunately, the relaxation method is not effective for this type of integer programming problems [9]. So instead of trying to solve Problem 2 directly, we propose an approximation algorithm which gives a quasi optimal solution.

3. Property of the problem

Before describing our algorithm for Problem 1, let us consider the property of our problem.

At first, let us consider the case that $m = 1$, i.e., a_i and b_j are scalars. It is trivial that the total sum of $c_{i\pi(i)}$ does not depend on π, i.e.,

$$\sum_{i=1}^{n} c_{i\pi(i)} = \sum_{i=1}^{n} a_i + \sum_{i=1}^{n} b_{\pi(i)} = \sum_{i=1}^{n} a_i + \sum_{j=1}^{n} b_j = \mathit{const}. \tag{6}$$

Next, let us consider the case that $m = 3$. In this case again the total sum of $c_{i\pi(i)}^{(1)}$ and that of $c_{i\pi(i)}^{(2)}$, $c_{i\pi(i)}^{(3)}$ do not depend on π. We denote $\frac{1}{n}\sum_{i=1}^{n}(a_i^{(1)} + b_i^{(1)})$ by μ_1 and $\frac{1}{n}\sum_{i=1}^{n}(a_i^{(2)} + b_i^{(2)})$ by μ_2, $\frac{1}{n}\sum_{i=1}^{n}(a_i^{(3)} + b_i^{(3)})$ by μ_3. For general m, the situation is unchanged.

Thus, we have the following Property 1.

Property 1 For each k, the total sum of $c_{i\pi(i)}^{(k)}$ does not depend on the selection of permutation π and is constant.

4. Algorithm for Problem 1

Hereafter we consider the case that the dimension of error vector is 3. So we rewrite our proposed algorithm for 2-dimensional case in [8] to 3-dimensional case.

Firstly, we assume the following two properties.

Assumption 1

(i) Each vector's dimension is 3, that is, $m = 3$.

(ii) $\mu_1 = \mu_2 = \mu_3$. We denote it μ.

Our proposed algorithm uses the searching method of scalar case's in [3]. It is the procedure that narrows gradually the range of edges' weight which restricts edges to be used to construct a complete matching between A and B.

We describe the essence of the searching method.

Searching method

$C = (c_{ij})_{i,j=1,2,\ldots,n}$: the cost matrix for an $n \times n$ assignment problem.
$v_1 < v_2 < \cdots < v_k$: the values appearing in C.
l : the smallest value's number of v_i. u : the largest one.
e_{ij} : the edge corresponds to c_{ij}.

```
while $u < k$ do
begin
```

```
  **if** $e_{ij}$s satisfy $v_l \leq w(e_{ij})(= c_{ij}) \leq v_u$ can create a complete matching
  **then**
    $l \leftarrow l + 1$
  **else**
    $u \leftarrow u + 1$;
**end;**
```

For the scalar case, the complexity of checking whether a complete matching exists is $O(n^{2.5})$ [10], and the above search step is iterated n^2 times in the worst case, hence it is easy to see that the total time complexity of the algorithm is $O(n^{4.5})$. Furthermore, there has been proposed an $O(n^4)$ algorithm [3].

A. Basis and idea for the 3-dimensional case

From Property 1, the best π is combining each $c^{(k)}_{i\pi(i)}(= a^{(k)}_i + b^{(k)}_{\pi(i)})$ as closer to the mean μ as possible.

We normalize each $a^{(1)}_i$ and represent it as $\tilde{a}^{(1)}_i$, that is,

$$\tilde{a}^{(1)}_i = \frac{a^{(1)}_i - \mu_{a_1}}{\sigma_{a_1}}, \tag{7}$$

where σ_{a_1} is the variance of $a^{(1)}_i$s and μ_{a_1} is the mean of them. By the normalization $a^{(1)}_i$ is converted to the probabilistic variable that the mean and the variance is 0 and 1, respectively. We define $\tilde{a}^{(2)}_i, \tilde{a}^{(3)}_i, \tilde{b}^{(1)}_j, \tilde{b}^{(2)}_j, \tilde{b}^{(3)}_j$ similarly.

So we can say that $c^{(k)}_{i\pi(i)} = a^{(k)}_i + b^{(k)}_{\pi(i)} = \mu$ is equivalent to $\tilde{a}^{(k)}_i + \tilde{b}^{(k)}_{\pi(i)} = 0$. Our problem is converted to find the permutation $\pi : \{1, 2, \ldots, n\} \rightarrow \{1, 2, \ldots, n\}$ that all $3n$ values $\tilde{a}^{(1)}_i + \tilde{b}^{(1)}_{\pi(i)}$ and $\tilde{a}^{(2)}_i + \tilde{b}^{(2)}_{\pi(i)}, \tilde{a}^{(3)}_i + \tilde{b}^{(3)}_{\pi(i)}$ are as close to origin as possible (Fig.2. Illustrate 2-dimensional case).

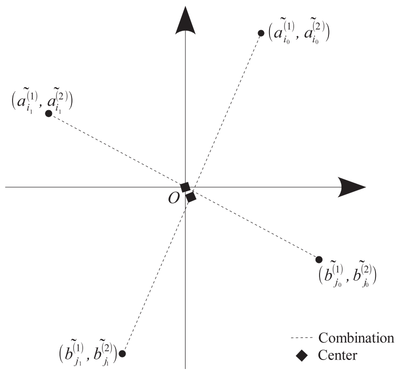
$(\tilde{a}^{(1)}_{i_0}, \tilde{a}^{(2)}_{i_0})$
$(\tilde{a}^{(1)}_{i_1}, \tilde{a}^{(2)}_{i_1})$
O
$(\tilde{b}^{(1)}_{j_0}, \tilde{b}^{(2)}_{j_0})$
$(\tilde{b}^{(1)}_{j_1}, \tilde{b}^{(2)}_{j_1})$
Combination
◆ Center

Fig. 2. 2-dimensional case : find a permutation π that combines each $(\tilde{a}^{(1)}_i, \tilde{a}^{(2)}_i)$ to $(\tilde{b}^{(1)}_{\pi(i)}, \tilde{b}^{(2)}_{\pi(i)})$ such that those of center to be as close to origin as possible. Similar for the 3-dimensional case.

From this fact, we consider the assignment problem whose objective is

$$g = \sum_{i=1}^{n} \max\left\{|\tilde{a}^{(1)}_i + \tilde{b}^{(1)}_{\pi(i)}|, |\tilde{a}^{(2)}_i + \tilde{b}^{(2)}_{\pi(i)}|, |\tilde{a}^{(3)}_i + \tilde{b}^{(3)}_{\pi(i)}|\right\}. \tag{8}$$

To minimize g, it is sufficient to minimize

$$|\tilde{a}^{(1)}_i + \tilde{b}^{(1)}_{\pi(i)}| \quad \text{and} \quad |\tilde{a}^{(2)}_i + \tilde{b}^{(2)}_{\pi(i)}|, \quad |\tilde{a}^{(3)}_i + \tilde{b}^{(3)}_{\pi(i)}|. \tag{9}$$

Therefore we can expect π that minimizes g gives an approximate solution to Problem 1. The assignment problem for g is a linear sum assignment problem, which is easy to solve.

And we construct a cost matrix $\tilde{C}$ for searching, where the element of $\tilde{C}$ is the maximum value of $|\tilde{a}_i^{(1)} + \tilde{b}_j^{(1)}|$,$|\tilde{a}_i^{(2)} + \tilde{b}_j^{(2)}|$, $|\tilde{a}_i^{(3)} + \tilde{b}_j^{(3)}|$. That is,

$$\tilde{c}_{ij} = \max\{|\tilde{a}_i^{(1)} + \tilde{b}_j^{(1)}|, |\tilde{a}_i^{(2)} + \tilde{b}_j^{(2)}|, |\tilde{a}_i^{(3)} + \tilde{b}_j^{(3)}|\}. \tag{10}$$

B. Algorithm

Outline of our proposed algorithm is as follows:

(i) According to the searching method of scalar case's algorithm, for $v_l \le \tilde{c}_{ij} \le v_u$, in order to get the permutation π, solve the linear sum assignment problem for g.

(ii) For Problem 1 if π gives better solution than the previous one, adopt π as the provisional best solution π^*.

(iii) Narrow the range $[v_l, v_u]$, then go to (i).

Now in the following Algorithm 1 we show in detail our proposed algorithm.

Algorithm 1

Preparation(Create a cost matrix)

Define the cost matrix $\tilde{C} = (\tilde{c}_{ij})$ by

$$\tilde{c}_{ij} = \max\{|\tilde{a}_i^{(1)} + \tilde{b}_j^{(1)}|, |\tilde{a}_i^{(2)} + \tilde{b}_j^{(2)}|, |\tilde{a}_i^{(3)} + \tilde{b}_j^{(3)}|\}. \tag{11}$$

$v_1 < v_2 < \cdots < v_k$: the values appear in $\tilde{C}$.

$\tilde{C}(l,u)$: the set of $\tilde{c}_{ij}$ satisfies $v_l \le \tilde{c}_{ij} \le v_u$.

$T_3(\pi)$: defined in (3) where $m = 3$.

Step 0 (Initialization)

$l \leftarrow 1, u \leftarrow 1, t \leftarrow \infty$.

Step 1 (Set edges' weights w_{ij} and solve the linear assignment problem)

```
for $i \leftarrow 1$ to $n$ do
  begin
  for $j \leftarrow 1$ to $n$ do
    begin
      if $v_l \le \tilde{c}_{ij} \le v_u$ then
        $w_{ij} \leftarrow \max\{\tilde{a}_i^{(1)} + \tilde{b}_j^{(1)}, \tilde{a}_i^{(2)} + \tilde{b}_j^{(2)}, \tilde{a}_i^{(3)} + \tilde{b}_j^{(3)}\}$
      else $w_{ij} \leftarrow \infty$;
    end;
  end;
```

Solve the linear assignment problem for w_{ij} :

$g = \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_{ij}$, then

if $g \ne \infty$ **then goto** Step 2 **else goto** Step 3;

Step 2 ($\tilde{C}(l,u)$ contains a complete matching)

```
if $l = u$ then
  begin
    $T \leftarrow 0$; $l^* \leftarrow u^*$; $u^* \leftarrow u$; stop
  end
else
  begin
    if $T > T_3(\pi)$ then
      begin
        $T \leftarrow T_3(\pi)$; $\pi^* \leftarrow \pi$; $l^* \leftarrow l$; $u^* \leftarrow u$
      end;
    $l \leftarrow l+1$;  **goto** Step 1
  **end;**
```

Step 3 ($\tilde{C}(l,u)$ does not contain a complete matching)

```
if $u = k$ then stop
  else
  begin
    $u \leftarrow u+1$;  goto Step 1
  end;
```

When Algorithm 1 terminated, π^* gives the quasi optimal solution to Problem 1.

The time complexity of Algorithm 1 is $O(n^5)$. Because Algorithm 1 solves the linear sum assignment problem n^2 times. The linear sum assignment problem can be solved in $O(n^3)$ [2].

5. Numerical experiments

We prepare the test data for the numerical experiments as follows : For vectors $\boldsymbol{a}_i = (a_i^{(1)}, a_i^{(2)}, a_i^{(3)})$ and $\boldsymbol{b}_j = (b_j^{(1)}, b_j^{(2)}, b_j^{(3)})$, $(i, j = 1, 2, \ldots, n)$, let $a_i^{(1)}, a_i^{(2)}, a_i^{(3)}$; $b_j^{(1)}, b_j^{(2)}, b_j^{(3)}$ follow the normal distribution that the mean is 10 and the variance is 1, respectively. Here $\rho_a^{(12)}$ is the correlation coefficient of $a_i^{(1)}$ and $a_i^{(2)}$, $\rho_a^{(31)}$ is that of $a_i^{(3)}$ and $a_i^{(1)}$. Similarly we define $\rho_b^{(12)}$ and $\rho_b^{(31)}$. When the number of vectors n is not so large, that is, at most 1000, $0 \le \rho_a^{(12)} < 0.5$ means the $a_i^{(1)}$ and $a_i^{(2)}$ have little correlative relation. So if $n \le 100$, we can say that it is sufficient to set $\rho_a^{(12)}$ to $0.5, 0.7$, and 0.9. It is the same for $\rho_a^{(31)}$ and $\rho_b^{(12)}$, $\rho_b^{(31)}$.

Each pair of $\rho_a^{(12)}$ and $\rho_a^{(31)}$, $\rho_b^{(12)}$ and $\rho_b^{(31)}$, we prepare several different data sets for $n = 40, 80, 120$. For each data set, we solve the problem by Algorithm 1, then compare to the exact solution. Exact solutions are found by solving Problem 1, that is the mixed integer programming problem(MIP), using the solver SCIP[1]. We denote three results here for $n = 40, 80, 120$, respectively. Table I-III show in the case $n = 40$ and also Table IV-VI show in the case $n = 80$, Table VII-IX show in the case $n = 120$.

In every two rows, the upper is the exact solution and the lower is the approximate one. The column ‘Sol. of bottleneck’ shows the solution to the problem. ‘Rel. Err.’ expresses the relative error of the approximate solution and the exact one, that is,

$$\frac{(\text{Approximate solution by Algorithm 1}) - (\text{Exact solution})}{(\text{Exact solution})}.$$

For the problem that minimizes each sum of elements, from ‘Rel. Err.’ we find our proposed algorithm does not give well approximate solutions to all cases. We think the result depends on the data’s property.It is not only the correlation coefficient.

But if we take $\max(\mathrm{Max}_i); i = 1, 2, 3$ and $\min(\mathrm{Min}_j); j = 1, 2, 3$, and we consider the interval $[\min(\mathrm{Min}_j), \max(\mathrm{Max}_i)]$ for exact solutions and approximate ones respectively, the two intervals almost coincide in every case. This is another bottleneck assignment problem formulated as Problem 3, that is, find the minimum interval includes 3 ones.

Problem 3 Given 2 sets of n vectors A and B, find the permutation π that minimizes

$$\tilde{T}_m(\pi) = \max\{\max_{1\le k\le m} \max_i c_{i\pi(i)}^{(k)} - \min_{1\le k\le m} \min_i c_{i\pi(i)}^{(k)}\}, \tag{12}$$

where $c_{i\pi(i)}^{(k)} = a_i^{(k)} + b_{\pi(i)}^{(k)}$. Here $m = 3$.

We also show the results of Problem 3 in each table. We define relative errors for the maximum value and the minimum one as follows:

$$\begin{aligned}
\mathrm{Gap}_{r1} &= \frac{\mathrm{Approx.}\max(\mathrm{Max}_i) - \mathrm{Exact.}\max(\mathrm{Max}_i)}{\mathrm{Exact.}\max(\mathrm{Max}_i)},\\
\mathrm{Gap}_{r2} &= \frac{\mathrm{Approx.}\min(\mathrm{Min}_i) - \mathrm{Exact.}\min(\mathrm{Min}_i)}{\mathrm{Exact.}\min(\mathrm{Min}_i)}.
\end{aligned}$$

[1]SCIP (Solving Constraint Integer Programs) is a non-commercial MIP solver developed at Zuse Institute Berlin. http://scip.zib.de/

We find $\mathrm{Gap}_{r1}, \mathrm{Gap}_{r2}$ are less than 5% in almost every case. So we can say that our algorithm is efficient for Problem 3.

A. *Considerations*

From numerical experiments, unfortunately our proposed algorithm is not always efficient for the original 3-dimensional bottleneck assignment problem. Why the approximate solutions by our algorithm in each interval's case is not good?

We think the reason is that when we create the cost matrix for searching, we take the representative in $\max\{c_{ij}^{(1)}, c_{ij}^{(2)}, c_{ij}^{(3)}\}$. Therefore the results are probably influenced by the maximum element.

For the original problem whose objective is to minimize each sum, how should we extend our method? One idea is that we do not create only one searching matrix but 3 ones, and narrow the 3 ranges independently. But when we manipulate 3 searching matrices, the other difficulties will occur.

TABLE I. $n = 40$, DATA SET NO.1

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max$_1$, Min$_1$), diff. (Max$_2$, Min$_2$), diff. (Max$_3$, Min$_3$), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap$_{r1}$, Gap$_{r2}$)
0.5, 0.5;	(21.81503217,19.59512506), 2.21990711 (21.31444143,19.09453432), 2.21990711 (21.41705637,19.19714926), 2.21990711	2.21990711	0.1470	(21.81503217,19.09453432)	(0.0000,-0.0117)
0.5, 0.5	(21.81503217,19.34601778), 2.46901439 (21.31444143,19.31780882), 1.99663261 (21.41705637,18.87082011), 2.54623626	2.54623626		(21.81503217,18.87082011)	
0.5, 0.5;	(21.80580840,19.15934005), 2.64646835 (21.66723535,18.99351390), 2.67372145 (21.76497755,19.09835537), 2.66662218	2.67372145	0.000	(21.80580840,18.9935139)	(-0.0064,0.0000)
0.5, 0.7	(21.58861597,19.11716460), 2.47145137 (21.66723535,18.99351390), 2.67372145 (21.13571833,19.13394120), 2.00177713	2.67372145		(21.66723535,18.99351390)	
0.5, 0.5;	(22.15289613,19.40908195), 2.74381418 (21.57488767,18.85891805), 2.71596962 (21.8228943,19.09258263), 2.73031167	2.74381418	0.000	(22.15289613,18.85891805)	(0.0000,0.0000)
0.5, 0.9	(22.15289613,19.40908195), 2.74381418 (21.50968142,18.85891805), 2.65076337 (21.69816874,19.09258263), 2.60558611	2.74381418		(22.15289613,18.85891805)	
0.5, 0.7;	(20.99940919,18.58655879), 2.41285040 (21.28655925,18.87370885), 2.41285040 (21.51519189,19.10234149), 2.41285040	2.41285040	0.0587	(21.51519189,18.58655879)	(-0.0005,0.0070)
0.5, 0.7	(20.99940919,18.71575644), 2.28365275 (21.28655925,18.73206750), 2.55449175 (21.50461106,19.06203809), 2.44257297	2.55449175		(21.50461106,18.71575644)	
0.5, 0.7;	(21.33488368,18.85018275), 2.48470093 (20.97960974,18.46130620), 2.51830354 (20.99078744,18.46176909), 2.52901835	2.52901835	0.0336	(21.33488368,18.46130620)	(-0.0086,0.0000)
0.5, 0.9	(21.15056282,19.10873679), 2.04182603 (20.85064173,18.46130620), 2.38933553 (21.07568411,18.46176909), 2.61391502	2.61391502		(21.15056282,18.46130620)	
0.5, 0.9;	(21.11903327,18.53461421), 2.58441906 (20.70273793,18.11831887), 2.58441906 (21.73600914,19.24136784),2.49464130	2.58441906	0.1060	(21.73600914,18.11831887)	(-0.0095,0.0205)
0.5, 0.9	(20.96915486,18.48923728), 2.47991758 (20.98565970,18.49771031), 2.48794939 (21.52928374,18.67080183), 2.85848191	2.85848191		(21.52928374,18.48923728)	
0.7, 0.7;	(20.81823644,18.62931892), 2.18891752 (20.73692109,18.54272061), 2.19420048 (21.12450000,18.93029952), 2.19420048	2.19420048	0.0668	(21.12450000,18.54272061)	(-0.0039,0.000)
0.7, 0.7	(20.85798795,18.62931892), 2.22866903 (20.83428788,18.54272061), 2.29156727 (21.04280016,18.70198460), 2.34081556	2.34081556		(21.04280016,18.54272061)	
0.7, 0.7;	(21.19064887,19.03932685), 2.15132202 (21.00680180,18.85547978), 2.15132202 (20.84869731,18.69737529), 2.15132202	2.15132202	0.1141	(21.19064887,18.69737529)	(-0.0018,0.0003)
0.7, 0.9	(21.15153278,18.75470127), 2.39683151 (20.73977097,18.70272854), 2.03704243 (21.12155818,18.80347384), 2.31808434	2.39683151		(21.15153278,18.70272854)	
0.7, 0.9;	(21.24138362,19.53866864), 1.70271498 (20.70903836,19.02161027), 1.68742809 (21.24733438,19.5446194), 1.70271498	1.70271498	0.0106	(21.24733438,19.02161027)	(-0.0050,0.0235)
0.7, 0.9	(20.95899401,19.46922302), 1.48977099 (21.06854235,19.34770591), 1.72083644 (21.14175043,19.42691372), 1.71483671	1.72083644		(21.14175043,19.46922302)	
0.9, 0.9;	(20.39858036,18.76555027), 1.63303009 (20.80084426,19.16781417), 1.63303009 (20.61766325,18.98463316),1.63303009	1.63303009	0.0321	(20.80084426,18.98463316)	(-0.0078,-0.0004)
0.9, 0.9	(20.63916611,19.06182560), 1.57734051 (20.55328365,18.98357907), 1.56970458 (20.45937406,18.77391319), 1.68546087	1.68546087		(20.63916611,18.76555027)	

TABLE II. $n = 40$, DATA SET NO.2

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max$_1$, Min$_1$), diff. (Max$_2$, Min$_2$), diff. (Max$_3$, Min$_3$), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap$_{r1}$, Gap$_{r2}$)
0.5, 0.5;	(22.09750979,19.20062100), 2.89688879 (21.60616442,18.73979427), 2.86637015 (21.22794551,18.39312316), 2.83482235	2.89688879		(22.09750979,18.39312316)	
0.5, 0.5	(21.26737858,19.05313340), 2.21424518 (21.55528607,18.65747283), 2.89781324 (21.14288780,18.80585683), 2.33703097	2.89781324	0.0003	(21.55528607,18.65747283)	(-0.0245,0.0144)
0.5, 0.5;	(21.03738737,18.77457847), 2.26280890 (21.37434593,19.10806740), 2.26627853 (21.19947363,18.9331951), 2.26627853	2.26280890		(21.37434593,18.77457847)	
0.5, 0.7	(21.08833355,18.74401128), 2.34432227 (21.37434593,19.10806740), 2.26627853 (21.18393014,19.21093336), 1.97299678	2.34432227	0.0360	(21.37434593,18.74401128)	(0.0000,-0.0016)
0.5, 0.5;	(20.96931193,18.83586267), 2.13344926 (21.38818575,19.22687262), 2.16131313 (21.47203342,19.34808726), 2.12394616	2.16131313		(21.47203342,18.83586267)	
0.5, 0.9	(21.32976249,19.09904579), 2.23071670 (21.38818575,19.18946433), 2.19872142 (21.15845017,19.13591579), 2.02253438	2.23071670	0.0321	(21.38818575,19.09904579)	(-0.0039,0.0140)
0.5, 0.7;	(20.69815993,18.42109219), 2.27706774 (21.21134698,18.93427924), 2.27706774 (21.14513880,18.87414467), 2.27099413	2.27706774		(21.21134698,18.42109219)	
0.5, 0.7	(21.07518118,18.46438901), 2.61079217 (21.14481926,18.74703125), 2.39778801 (20.96088042,18.25907747), 2.70180295	2.70180295	0.1865	(21.14481926,18.25907747)	(-0.0031,-0.0088)
0.5, 0.7;	(20.97288952,18.98403730), 1.98885222 (20.92519820,19.00621409), 1.91898411 (21.00186864,19.04806486), 1.95380378	1.98885222		(21.00186864,18.9840373)	
0.5, 0.9	(21.03779594,18.98403730), 2.05375864 (20.88214597,18.97541896), 1.90672701 (21.02581215,19.04806486), 1.97774729	2.05375864	0.0326	(21.03779594,18.97541896)	(0.0017,-0.0005)
0.5, 0.9;	(20.73191314,19.41438625), 1.31752689 (20.76943915,19.43880701), 1.33063214 (20.77310236,19.44247022), 1.33063214	1.33063214		(20.77310236,19.41438625)	
0.5, 0.9	(20.76542715,19.62560038), 1.13982677 (20.78345538,19.38450069), 1.39895469 (20.63076998,19.24559193), 1.38517805	1.39895469	0.0513	(20.78345538,19.24559193)	(0.0005,-0.0087)
0.7, 0.7;	(21.53089719,19.00433917), 2.52655802 (21.47206851,18.94551049), 2.52655802 (21.46586074,18.93930272), 2.52655802	2.52655802		(21.53089719,18.93930272)	
0.7, 0.7	(21.27777584,19.09739705), 2.18037879 (21.47206851,18.94551049), 2.52655802 (21.37160465,18.93930272), 2.43230193	2.52655802	0.000	(21.47206851,18.93930272)	(-0.0027,0.0000)
0.7, 0.7;	(21.80619566,19.06020889), 2.74598677 (21.37697058,18.55973965), 2.81723093 (20.86524233,18.05877146), 2.80647087	2.81723093		(21.80619566,18.05877146)	
0.7, 0.9	(21.41408845,18.97491660), 2.43917185 (21.59062660,18.55973965), 3.03088695 (20.74017930,18.05877146), 2.68140784	3.03088695	0.0758	(21.59062660,18.05877146)	(-0.0099,0.0000)
0.7, 0.9;	(20.80616599,18.61979915), 2.18636684 (20.61432050,18.46585070), 2.14846980 (20.91061491,18.78552972), 2.12508519	2.18636684		(20.91061491,18.46585070)	
0.7, 0.9	(20.76161400,18.61979915), 2.14181485 (20.76046275,18.46585070), 2.29461205 (20.91061491,18.71882922), 2.19178569	2.29461205	0.0495	(20.91061491,18.46585070)	(0.0000,0.0000)
0.9, 0.9;	(21.14804202,19.14930885), 1.99873317 (20.70316023,18.70442706), 1.99873317 (20.85315407,18.85442090), 1.99873317	1.99873317		(21.14804202,18.70442706)	
0.9, 0.9	(21.11079515,19.20374604), 1.90704911 (20.78132676,18.70442706), 2.07689970 (20.85074985,18.85442090), 1.99632895	2.07689970	0.0391	(21.11079515,18.85442090)	(-0.0018,0.0080)

TABLE III. $n = 40$, DATA SET NO.3

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	$(\mathrm{Max}_1, \mathrm{Min}_1)$, diff. $(\mathrm{Max}_2, \mathrm{Min}_2)$, diff. $(\mathrm{Max}_3, \mathrm{Min}_3)$, diff.	Sol. of bottleneck	Rel.Err.	$(\max(\mathrm{Max}_i), \min(\mathrm{Min}_i))$	$(\mathrm{Gap}_{r1}, \mathrm{Gap}_{r2})$
0.5,0.5;	(21.02284509,19.04836931), 1.97447578 (21.00727800,19.03280222), 1.97447578 (20.77828235,18.83388115), 1.94440120	1.97447578		(21.02284509,18.83388115)	
0.5,0.5	(21.16035474,19.10355897), 2.05679577 (20.88616680,18.97025033), 1.91591647 (21.01430843,18.85841515), 2.15589328	2.15589328	0.0919	(21.16035474,18.85841515)	(0.0065,0.0013)
0.5,0.5;	(20.99344367,18.59743668), 2.39600699 (20.68509129,18.32778772), 2.35730357 (20.8890702,18.49306321), 2.39600699	2.39600699		(20.99344367,18.32778772)	
0.5,0.7	(21.04958144,18.59743668), 2.45214476 (20.77880688,18.36331077), 2.41549611 (21.16006688,18.57683208), 2.58323480	2.58323480	0.0781	(21.16006688,18.36331077)	(0.0079,0.0019)
0.5,0.5;	(21.39775597,19.61787364), 1.77988233 (20.70932983,18.91808259), 1.79124724 (21.04526517,19.25401793), 1.79124724	1.79124724		(21.39775597,18.91808259)	
0.5,0.9	(21.24956957,19.35023881), 1.89933076 (20.84937801,18.95584005), 1.89353796 (21.04526517,19.11949652), 1.92576865	1.92576865	0.0751	(21.24956957,18.95584005)	(-0.0069,0.0020)
0.5,0.7;	(20.95271556,18.68533232), 2.26738324 (21.11252923,18.84514599), 2.26738324 (21.19036422,18.92298098), 2.26738324	2.26738324		(21.19036422,18.68533232)	
0.5,0.7	(21.31085473,19.32462614), 1.98622859 (21.22299871,18.82727498), 2.39572373 (21.32143879,19.02929500), 2.29214379	2.39572373	0.0566	(21.32143879,18.82727498)	(0.0062,0.0076)
0.5,0.7;	(21.57448776,19.28650629), 2.28798147 (21.49124258,19.29166904), 2.19957354 (21.46037395,19.21338208), 2.24699187	2.28798147		(21.57448776,19.21338208)	
0.5,0.9	(21.57448776,19.28650629), 2.28798147 (21.25378183,19.16222045), 2.09156138 (21.46863962,19.24187835), 2.22676127	2.28798147	0.0000	(21.57448776,19.16222045)	(0.0000,-0.0027)
0.5,0.9;	(21.09474252,19.33949428), 1.75524824 (20.95402973,19.19878149), 1.75524824 (21.05215708¡19.29690884), 1.75524824	1.75524824		(21.09474252,19.19878149)	
0.5,0.9;	(21.09474252,19.33949428), 1.75524824 (20.86917576,19.33598240), 1.53319336 (21.05215708,19.36181191), 1.69034517	1.75524824	0.0000	(21.09474252,19.33598240)	(0.0000,0.0071)
0.7,0.7;	(20.95008187,19.17959390), 1.77048797 (20.77636316,19.00587519), 1.77048797 (21.00705208,19.23656411), 1.77048797	1.77048797		(21.00705208,19.00587519)	
0.7,0.7	(21.04149608,19.17959390), 1.86190218 (20.94536983,19.00587519), 1.93949464 (21.05865687,19.24516721), 1.81348966	1.93949464	0.0955	(21.05865687,19.00587519)	(0.0025,0.0000)
0.7,0.7;	(20.85414749,18.20317671), 2.65097078 (20.58251909,17.93154831), 2.65097078 (20.93378078,18.31635262), 2.61742816	2.65097078		(20.93378078,17.93154831)	
0.7,0.9	(21.03974999,18.20317773), 2.83657226 (20.73775745,17.93154831), 2.80620914 (20.79470078,18.31635262), 2.47834816	2.83657226	0.0700	(21.03974999,17.93154831)	(0.0051,0.0000)
0.7,0.9;	(20.81304371,18.30736174), 2.50568197 (20.87061183,18.2896104), 2.58100143 (20.87554553,18.22893352), 2.64661201	2.64661201		(20.87554553,18.22893352)	
0.7,0.9	(20.78174884,18.36866624), 2.41308260 (20.83034936,18.28961040), 2.54073896 (20.87554553,18.22893352), 2.64661201	2.64661201	0.0000	(20.87554553,18.22893352)	(0.0000,0.0000)
0.9,0.9;	(20.81350832,19.24280819), 1.57070013 (20.75210008,19.08120993), 1.67089015 (20.61067753,18.93978738), 1.67089015	1.67089015		(20.81350832,18.93978738)	
0.9,0.9	(20.81350832,19.16966292), 1.64384540 (20.75210008,19.08120993), 1.67089015 (20.77945040,19.07780830), 1.70164210	1.70164210	0.0834	(20.81350832,19.07780830)	(0.0000,0.0073)

TABLE IV. $n = 80$, DATA SET NO.4

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max$_1$, Min$_1$), diff. (Max$_2$, Min$_2$), diff. (Max$_3$, Min$_3$), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap$_{r1}$, Gap$_{r2}$)
0.5, 0.5;	(21.07003948,18.83632014), 2.23371934 (21.36726131,19.13999145), 2.22726986 (20.94233799,18.71946098), 2.22287701	2.23371934		(21.36726131,18.71946098)	
0.5, 0.5	(21.07003948,18.83632014), 2.23371934 (21.06897364,18.97385619), 2.09511745 (20.92268075,18.70512374), 2.21755701	2.23371934	0.0000	(21.07003948,18.70512374)	(-0.0139,-0.0008)
0.5, 0.5;	(20.95925618,18.75237689), 2.20687929 (21.04891855,18.84203926), 2.20687929 (20.91496969,18.78196712), 2.13300257	2.20687929		(21.04891855,18.66645449)	
0.5, 0.7	(20.82502863,18.66645449), 2.15857414 (21.04839844,18.74665732), 2.30174112 (21.08419051,18.66645449), 2.41773602	2.41773602	0.0955	(21.08419051,18.36266207)	(0.0017,-0.0163)
0.5, 0.5;	(21.20478878,19.06021439), 2.14457439 (21.51033042,19.36529163), 2.14503879 (21.17076697,19.03125857), 2.1395084	2.14503879		(21.51033042,19.03125857)	
0.5, 0.9	(21.25822622,18.93392486), 2.32430136 (21.51033042,18.97881156), 2.53151886 (21.23186635,18.74753846), 2.48432789	2.53151886	0.1802	(21.51033042,18.74753846)	(0.0000,-0.0149)
0.5, 0.7;	(20.88108803,18.66629281), 2.21479522 (20.76124978,18.55733673), 2.20391305 (20.99265937,18.83933001), 2.15332936	2.21479522		(20.99265937,18.55733673)	
0.5, 0.7	(21.03541179,18.78659293), 2.24881886 (20.99508001,18.84843830), 2.14664171 (21.10392946,18.78844938), 2.31548008	2.31548008	0.0455	(21.10392946,18.78659293)	(0.0053,0.0124)
0.5, 0.7;	(21.00522474,18.74083759), 2.26438715 (21.04053673,18.82193916), 2.21859757 (21.15119650,18.88680935), 2.26438715	2.26438715		(21.15119650,18.74083759)	
0.5, 0.9	(21.06678546,18.60213847), 2.46464699 (21.02962917,18.53187921), 2.49774996 (21.09241076,18.57668720), 2.51572356	2.51572356	0.1110	(21.09241076,18.53187921)	(-0.0028,-0.0111)
0.5, 0.9;	(21.35743317,19.24020032), 2.11723285 (21.42042596,19.28508900), 2.13533696 (21.50118161,19.35342377), 2.14775784	2.14775784		(21.50118161,19.24020032)	
0.5, 0.9	(21.35743317,19.22918253), 2.12825064 (21.42042596,19.18238556), 2.23804040 (21.50118161,19.35342377), 2.14775784	2.23804040	0.0571	(21.50118161,19.18238556)	(0.0000,-0.0030)
0.7, 0.7;	(21.06805968,19.24793017), 1.82012951 (20.94917272,19.11687127), 1.83230145 (21.11440227,19.29744328), 1.81695899	1.83230145		(21.11440227,19.11687127)	
0.7, 0.7	(20.90854706,18.94664443), 1.96190263 (20.92977196,18.98942802), 1.94034394 (20.81752821,18.82414100), 1.99338721	1.99338721	0.0879	(20.92977196,18.82414100)	(-0.0087,-0.0153)
0.7, 0.7;	(20.71905965,18.90144090), 1.81761875 (20.99101215,19.19077209), 1.80024006 (21.07914778,19.27126040), 1.80788738	1.817618745		(21.07914778,18.90144090)	
0.7, 0.9	(20.79066468,18.99373593), 1.79692875 (20.72391447,18.97259244), 1.75132203 (20.64391767,18.79775705), 1.84616062	1.84616062	0.0157	(20.79066468,18.79775705)	(-0.0137,-0.0055)
0.7, 0.9;	(20.74856480,19.23895118), 1.50961362 (20.60673606,19.08310400), 1.52363206 (20.85515494,19.33402141), 1.52113353	1.523632060		(20.85515494,19.08310400)	
0.7, 0.9	(20.55215776,18.62594116), 1.92621660 (20.73903995,18.93007925), 1.80896070 (20.83304745,19.00052437), 1.83252308	1.92621660	0.2642	(20.83304745,18.62594116)	(-0.0011,-0.0240)
0.9, 0.9;	(20.99628897,19.67353966), 1.32274931 (20.95604175,19.63329244), 1.32274931 (20.91679193,19.61203199), 1.30475994	1.32274931		(20.99628897,19.61203199)	
0.9, 0.9	(20.99628897,19.66349699), 1.33279198 (20.81254202,19.68492476), 1.12761726 (20.88498574,19.48379896), 1.40118678	1.40118678	0.0593	(20.99628897,19.48379896)	(0.0000,-0.0065)

TABLE V. $n = 80$, DATA SET NO.5

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max_1, Min_1), diff. (Max_2, Min_2), diff. (Max_3, Min_3), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap_{r1}, Gap_{r2})
0.5, 0.5;	(21.74178207,18.81694241), 2.92483966 (21.17724374,18.16831008), 3.00893366 (20.61515986, 17.59686588), 3.01829398	3.13154685		(21.74178207,17.59686588)	
0.5, 0.5	(21.25202809,19.06626294), 2.18576515 (20.67918158,18.87987362), 1.79930796 (20.77891875,17.59686588), 3.18205287	3.18205287	0.0161	(21.25202809,17.59686588)	(-0.0225,0.0000)
0.5, 0.5;	(22.02306676,18.94812514), 3.07494162 (21.66738032,18.55402802), 3.11335230 (20.60744244,17.49409014), 3.1133523	3.11335230		(22.02306676,17.49409014)	
0.5, 0.7	(21.23507550,19.31553863), 1.91953687 (21.09691642,18.43749224), 2.65942418 (21.13571487,18.01025106), 3.12546381	3.12546381	0.0039	(21.23507550,18.01025106)	(-0.0358,0.0295)
0.5, 0.5;	(21.71289780,19.17870770), 2.53419010 (20.50596236,17.94174483), 2.56421753 (21.60557592,19.04200334), 2.56357258	2.56421753		(21.71289780,17.94174483)	
0.5, 0.9	(21.71289780,18.92084654), 2.79205126 (21.16249256,18.33881515), 2.82367741 (21.49608665,18.32876249), 3.16732416	3.16732416	0.2352	(21.71289780,18.32876249)	(0.0000,0.0216)
0.5, 0.7;	(20.69666457,18.48045838), 2.21620619 (21.07742458,18.87045354), 2.20697104 (20.68209117,18.45932601), 2.22276516	2.22276516		(21.07742458,18.45932601)	
0.5, 0.7	(20.88222434,18.48045838), 2.40176596 (21.03349116,18.87045354), 2.16303762 (20.86459888,18.45932601), 2.40527287	2.40527287	0.0821	(21.03349116,18.45932601)	(-0.0021,0.0000)
0.5, 0.7;	(20.97316509,19.23450044), 1.73866465 (20.84853274,19.14284395), 1.70568879 (21.13952192,19.39739238), 1.74212954	1.74212954		(21.13952192,19.14284395)	
0.5, 0.9	(21.21224915,19.31362299), 1.89862616 (21.25387332,19.23824067), 2.01563265 (21.13547581,19.07732271), 2.05815310	2.05815310	0.1814	(21.25387332,19.07732271)	(0.0054,-0.0034)
0.5,0.9;	(21.55005596,19.68707973), 1.86297623 (20.98033239,19.12373253), 1.85659986 (21.48932172,19.63985898), 1.84946274	1.86297623		(21.55005596,19.12373253)	
0.5,0.9	(21.33527223,19.41769712), 1.91757511 (20.85710369,18.97161384), 1.88548985 (21.47490259,19.49692706), 1.97797553	1.97797553	0.0617	(21.47490259,18.97161384)	(0.0159,-0.0089)
0.7,0.7;	(21.08381599,18.47915837), 2.60465762 (21.33675939,18.73210177), 2.60465762 (21.21513262,18.61581084), 2.59932178	2.60465762		(21.33675939,18.47915837)	
0.7,0.7	(21.08381599,18.47915837), 2.60465762 (21.31706597,18.84414855), 2.47291742 (21.19981311,18.75374931), 2.44606380	2.60465762	0.0000	(21.31706597,18.47915837)	(-0.0009,0.0000)
0.7,0.7;	(20.91407099,19.08071040), 1.83336059 (20.61490978,18.82007557), 1.79483421 (20.87488031,19.05398022), 1.82090009	1.83336059		(20.91407099,18.82007557)	
0.7,0.9	(20.91407099,19.08071040), 1.83336059 (20.61490978,18.80104596), 1.81386382 (20.87488031,19.05398022), 1.82090009	1.83336059	0.0000	(20.91407099,18.80104596)	(0.0000,-0.0010)
0.7,0.9;	(21.17845736,19.47826510), 1.70019226 (20.70818495,19.05771622), 1.65046873 (20.85249302,19.15230076), 1.70019226	1.70019226		(21.17845736,19.05771622)	
0.7,0.9	(21.17845736,19.36027234), 1.81818502 (20.69101041,19.15347878), 1.53753163 (20.85249302,19.09315126), 1.75934176	1.81818502	-0.0649	(21.17845736,19.09315126)	(0.0000,0.0019)
0.9,0.9;	(20.51262240,18.96580027), 1.54682213 (20.40953286,18.86423010), 1.54530276 (21.02407464,19.48195895), 1.54211569	1.54682213		(21.02407464,18.86423010)	
0.9,0.9	(20.58944900,18.96580027), 1.62364873 (20.49099608,18.86423010), 1.62676598 (20.53844305,19.27509898), 1.26334407	1.62676598	-0.0491	(20.58944900,18.86423010)	(-0.0207,0.0000)

TABLE VI. $n = 80$, DATA SET NO.6

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max_1, Min_1), diff. (Max_2, Min_2), diff. (Max_3, Min_3), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap_{r1}, Gap_{r2})
0.5, 0.5;	(20.69863151,18.78886779), 1.90976372 (20.31997282,18.41715832), 1.90281450 (20.95385849,19.04409477), 1.90976372	1.90976372		(20.95385849,18.41715832)	
0.5, 0.5	(20.97190060,18.78886779), 2.18303281 (20.70481423,18.49628295), 2.20853128 (20.92886967,18.85564251), 2.07322716	2.20853128	0.1564	(20.97190060,18.49628295)	(0.0009,0.0043)
0.5, 0.5;	(20.89316346,18.83871926), 2.05444420 (20.84694979,18.86566187), 1.98128792 (20.96511696,18.92708617), 2.03803079	2.05444420		(20.96511696,18.83871926)	
0.5, 0.7	(20.90174809,19.04183134), 1.85991675 (21.08490022,18.93150224), 2.15339798 (20.96511696,18.93867767), 2.02643929	2.15339798	0.0482	(21.08490022,18.93150224)	(0.0057,0.0049)
0.5, 0.5;	(20.80468638,18.78838553), 2.01630085 (20.85843410,18.83090357), 2.02753053 (20.91416675,18.88641427), 2.02775248	2.02775248		(20.91416675,18.78838553)	
0.5, 0.9	(20.85570393,18.78838553), 2.06731840 (20.73972884,19.02240712), 1.71732172 (21.02193361,18.99441290), 2.02752071	2.06731840	0.0195	(20.85570393,18.78838553)	(-0.0028,0.0000)
0.5,0.7;	(21.53761477,19.48622068), 2.05139409 (21.12016869,19.05338610), 2.06678259 (20.84537246,18.87970605), 1.96566641	2.06678259		(21.53761477,18.87970605)	
0.5,0.7	(21.53155685,18.98532114), 2.54623571 (21.57007132,19.08809538), 2.48197594 (20.89970485,18.88468514), 2.01501971	2.54623571	0.2320	(21.57007132,18.88468514)	(0.0015,0.003)
0.5,0.7;	(20.94915516,18.89398239), 2.05517277 (20.82109907,18.78376890), 2.03733017 (21.24342115,19.15869294), 2.08472821	2.08472821		(21.24342115,18.78376890)	
0.5,0.9	(20.87016583,18.89398239), 1.97618344 (20.81820742,18.87078591), 1.94742151 (21.24342115,19.03934089), 2.20408026	2.20408026	0.0573	(21.24342115,18.87078591)	(0.0000,0.0046)
0.5,0.9;	(21.12027682,18.92247065), 2.19780617 (20.61524493,18.43367251), 2.18157242 (20.64937654,18.45536562), 2.19401092	2.19780617		(21.12027682,18.43367251)	
0.5,0.9	(21.32925897,19.19036511), 2.13889386 (21.29077782,18.83272926), 2.45804856 (21.43635104,18.94796031), 2.48839073	2.48839073	0.1322	(21.43635104,18.83272926)	(0.0119,-0.0038)
0.7,0.7;	(21.18435567,19.05349628), 2.13085939 (21.05090920,18.91481068), 2.13609852 (21.02704626,18.90418760), 2.12285866	2.13609852		(21.18435567,18.90418760)	
0.7,0.7	(21.14075280,19.05349628), 2.08725652 (21.05090920,18.91481068), 2.13609852 (20.92641108,18.91061913), 2.01579195	2.13609852	0.0000	(21.14075280,18.91061913)	(0.0010,0.0259)
0.7,0.7;	(21.01588735,19.11747928), 1.89840807 (21.17969829,19.28541034), 1.89428795 (21.15211452,19.23830318), 1.91381134	1.91381134		(21.17969829,19.11747928)	
0.7,0.9	(21.07459385,19.11747928), 1.95711457 (20.96761585,19.01330708), 1.95430877 (21.00590296,19.20613789), 1.79976507	1.95711457	0.0226	(21.07459385,19.01330708)	(-0.0050,-0.0054)
0.7,0.9;	(20.84292595,19.07528132), 1.76764463 (21.09828783,19.32357113), 1.77471670 (21.09204875,19.32010546), 1.77194329	1.77471670		(21.09828783,19.07528132)	
0.7,0.9	(20.84292595,19.40385793), 1.43906802 (21.09828783,19.22053056), 1.87775727 (20.82817884,19.34091073), 1.48726811	1.87775727	0.0581	(21.09828783,19.22053056)	(0.000,0.0076)
0.9,0.9;	(20.98259231,18.88928437), 2.09330794 (20.78605120,18.59361267), 2.19243853 (20.43456709,18.24212856), 2.19243853	2.19243853		(20.98259231,18.24212856)	
0.9,0.9	(20.72947385,18.67104237), 2.05843148 (20.49790195,18.07203414), 2.42586781 (20.71071967,18.27190017), 2.43881950	2.43881950	0.1124	(20.72947385,18.07203414)	(-0.0121,-0.0093)

TABLE VII.　$n = 120$, DATA SET NO.7

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max$_1$, Min$_1$), diff. (Max$_2$, Min$_2$), diff. (Max$_3$, Min$_3$), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap$_{r1}$, Gap$_{r2}$)
0.5,0.5;	(20.72407964,18.77534199), 1.94873765 (20.45131572,18.50257807), 1.94873765 (20.90048454,18.95174689), 1.94873765	1.94873765		(20.90048454,18.50257807)	
0.5,0.5	(20.95418859,18.77534199), 2.17884660 (20.68282720,18.50257807), 2.18024913 (20.85806928,19.17398956), 1.68407972	2.18024913	0.1188	(20.95418859,18.50257807)	(0.0026,0.0000)
0.5,0.5;	(20.23356847,17.84578251), 2.38778596 (20.72861653 18.32776055), 2.40085598 (21.25361118,18.85250480), 2.40110638	2.40110638		(21.25361118,17.84578251)	
0.5,0.7	(20.69277822,18.63046706), 2.06231116 (21.02244697,18.52776990), 2.49467707 (20.76873334,18.62621691), 2.14251643	2.49467707	0.0390	(21.02244697,18.52776990)	(-0.0109,0.0382)
0.5,0.5;	(20.41596509,18.43985259), 1.97611250 (20.95603843,19.03235573), 1.92368270 (20.59174689,18.62058718), 1.97115971	1.97115971		(20.95603843,18.43985259)	
0.5,0.9	(20.76512955,18.62397343), 2.14115612 (21.00278644,18.88293189), 2.11985455 (20.91654415,18.77445132), 2.14209283	2.14209283	0.0867	(20.91654415,18.62397343)	(-0.0019,0.0100)
0;5,0.7;	(20.64811581,18.84647091), 1.80164490 (20.98623646,19.19181409), 1.79442237 (20.68554374,18.88389884), 1.80164490	1.80164490		(20.98623646,18.84647091)	
0.5,0.7	(20.83152669,18.87959374), 1.95193295 (20.91246417,18.94636952), 1.96609465 (20.90015105,18.98139752), 1.91875353	1.96609465	0.0913	(20.91246417,18.87959374)	(0.0110,0.0018)
0.5,0.7;	(20.81435396,18.80751412), 2.00683984 (20.74187776,18.72908819), 2.01278957 (20.81590736,18.86134473), 1.95456263	2.01278957		(20.81590736,18.72908819)	
0.5,0.9	(20.79258411,18.83791720), 1.95466691 (20.94716527,19.01181902), 1.93534625 (20.89020518,18.86134473), 2.02886045	2.02886045	0.0080	(20.94716527,18.83791720)	(0.0063,0.0058)
0.5,0.9;	(20.53782064,18.66389448), 1.87392616 (20.51907592,18.63200854), 1.88706738 (20.8112493,18.93540579), 1.87584351	1.88706738		(20.81124930,18.63200854)	
0.5,0.9	(20.62540706,18.76566121), 1.85974585 (20.93197608,18.81464194), 2.11733414 (20.81124930,19.02803905), 1.78321025	2.11733414	0.1220	(20.93197608,18.81464194)	(0.0058,0.0098)
0.7,0.7;	(20.66874738,19.09449076), 1.57425662 (20.49053062,18.97451679), 1.51601383 (20.61486575,19.05041674), 1.56444901	1.57425662		(20.66874738,18.97451679)	
0.7,0.7	(20.66874738,19.09449076), 1.57425662 (20.56909398,19.10571455), 1.46337943 (20.53633197,19.03834980), 1.49798217	1.57425662	0.0000	(20.66874738,19.03834980)	(0.0000,0.0034)
0.7,0.7;	(20.74165563,18.78473399), 1.95692164 (20.83319916,18.86364303), 1.96955613 (20.66873962,18.69918349),1.96955613	1.96955613		(20.83319916,18.69918349)	
0.7,0.9	(20.67080573,18.83090762), 1.83989811 (20.82519202,18.86364303), 1.96154899 (20.66873962,18.69918349), 1.96955613	1.96955613	0.0000	(20.82519202,18.69918349)	(-0.0004,0.0000)
0.7,0.9;	(21.08668758,19.17664021), 1.91004737 (21.01646363,19.10112857), 1.91533506 (21.04900733,19.14511487), 1.90389246	1.91533506		(21.08668758,19.10112857)	
0.7,0.9	(21.04081593,19.21034158), 1.83047435 (21.22184312,19.10112857), 2.12071455 (21.08110331,19.23098858), 1.85011473	2.12071455	0.1072	(21.22184312,19.10112857)	(0.0064,0.0000)
0.9,0.9;	(20.59177787,18.55953448), 2.03224339 (21.22730148,19.17404021), 2.05326127 (20.79103748, 18.74824274), 2.04279474	2.05326127		(21.22730148,18.55953448)	
0.9,0.9	(20.76546223,18.55953448), 2.20592775 (20.77703897,19.17404021), 1.60299876 (20.63851362,18.75495132), 1.88356230	2.20592775	0.0744	(20.77703897,18.55953448)	(-0.0212,0.0000)

TABLE VIII. $n = 120$, DATA SET NO.8

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max_1, Min_1), diff. (Max_2, Min_2), diff. (Max_3, Min_3), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap_{r1}, Gap_{r2})
0.5,0.5;	(20.91106513,18.75632677), 2.15473836 (20.85690691,18.69239589), 2.16451102 (21.04556103,18.88115346), 2.16440757	2.16451102		(21.04556103,18.69239589)	
0.5,0.5	(20.90300541,18.78267932), 2.12032609 (20.98199683,19.04955811), 1.93243872 (21.19668576,18.88749015), 2.30919561	2.30919561	0.00668	(21.19668576,18.78267932)	(0.0072,0.0048)
0.5,0.5;	(21.73538467,19.20803488), 2.52734979 (20.64889425,18.12014934), 2.52874491 (21.38768701,18.88599304), 2.50169397	2.52874491		(21.73538467,18.12014934)	
0.5,0.7	(21.22248607,19.14008521), 2.08240086 (20.84053703,18.12014934), 2.72038769 (21.48245634,19.18710763), 2.29534871	2.72038769	0.0758	(21.48245634,18.12014934)	(-0.0116,0.0000)
0.5,0.5;	(21.98072921,19.81490130), 2.16582791 (21.02696993,18.88057075), 2.14639918 (21.58660417,19.42123892), 2.16536525	2.16582791		(21.98072921,18.88057075)	
0.5,0.9	(21.51024933,19.13195940), 2.37828993 (20.87567008,18.71913331), 2.15653677 (21.34351756,18.78838741), 2.55513015	2.55513015	0.1797	(21.51024933,18.71913331)	(-0.0214,-0.0086)
0.5,0.7;	(21.15793897,18.88003275), 2.27790622 (21.60274561,19.32841691), 2.27432870 (21.13286131,18.86118731), 2.27167400	2.27790622		(21.60274561,18.86118731)	
0.5,0.7	(21.41850264,18.88003275), 2.53846989 (21.32744723,19.22229193), 2.10515530 (21.06825632,19.09632951), 1.97192681	2.53846989	0.1144	(21.41850264,18.88003275)	(-0.0085,0.0010)
0.5,0.7;	(21.51243914,19.34849795), 2.16394119 (22.04657254,19.87996750), 2.16660504 (21.30373626,19.13713122), 2.16660504	2.16660504		(22.04657254,19.13713122)	
0.5,0.9	(21.41959125,18.98924254), 2.43034871 (21.33103891,19.43629448), 1.89474443 (21.08701262,18.61416990), 2.47284272	2.47284272	0.1413	(21.41959125,18.61416990)	(-0.0284,-0.0273)
0.5,0.9;	(20.97947401,19.34862815), 1.63084586 (20.83403626,19.20210373), 1.63193253 (21.07789891,19.45559065), 1.62230826	1.63193253		(21.07789891,19.20210373)	
0.5,0.9	(21.02769107,19.55695127), 1.47073980 (20.83403626,19.20210373), 1.63193253 (20.91902365,19.44547796), 1.47354569	1.63193253	0.0000	(21.02769107,19.20210373)	(-0.0024,0.0000)
0.7,0.7;	(21.04960113,19.09578156), 1.95381957 (20.96497158,19.01115201), 1.95381957 (20.84989857,18.89607900), 1.95381957	1.95381957		(21.04960113,18.89607900)	
0.7,0.7	(20.82333615,19.03962376), 1.78371239 (20.95633016,19.02259712), 1.93373304 (20.98864083,18.89607900), 2.09256183	2.09256183	0.0710	(20.98864083,18.89607900)	(-0.0029,0.0000)
0.7,0.7;	(20.84988501,19.25974413), 1.59014088 (20.81627517,19.22521871), 1.59105646 (20.94624556,19.35279200), 1.59345356	1.59345356		(20.94624556,19.22521871)	
0.7,0.9	(20.84909915,19.27057607), 1.57852308 (20.69186107,19.22621662), 1.46564445 (20.94624556,19.35279200), 1.59345356	1.59345356	0.0000	(20.94624556,19.22621662)	(0.0000,0.0001)
0.7,0.9;	(20.80409267,19.10862122), 1.69547145 (20.55020133,18.85550697), 1.69469436 (20.91381839,19.22850692), 1.68531147	1.69547145		(20.91381839,18.85550697)	
0.7,0.9	(20.57102923,19.45399280), 1.11703643 (20.86253474,19.26343628), 1.59909846 (21.14000327,19.35343648), 1.78656679	1.78656679	0.0537	(21.14000327,19.26343628)	(0.0108,0.0216)
0.9,0.9;	(20.70549222,19.18213490), 1.52335732 (20.68495863,19.15573555), 1.52922308 (21.20528617,19.67292954), 1.53235663	1.53235663		(21.20528617,19.15573555)	
0.9,0.9	(20.53461341,19.30825561), 1.22635780 (20.68495863,19.25419726), 1.43076137 (21.20528617,19.49171398), 1.71357219	1.71357219	0.1183	(21.20528617,19.25419726)	(0.0000,0.0051)

TABLE IX. $n = 120$, DATA SET NO.9

$\rho_a^{(12)}, \rho_a^{(31)}$; $\rho_b^{(12)}, \rho_b^{(31)}$	(Max_1, Min_1), diff. (Max_2, Min_2), diff. (Max_3, Min_3), diff.	Sol. of bottleneck	Rel.Err.	$(\max(\text{Max}_i), \min(\text{Min}_i))$	(Gap_{r1}, Gap_{r2})
0.5,0.5;	(21.20447939,19.08526061), 2.11921878 (21.12036121,19.00114243), 2.11921878 (21.26386944,19.16327445), 2.10059499	2.11921878		(21.26386944,19.00114243)	
0.5,0.5	(21.20447939,19.04216535), 2.16231404 (21.12036121,18.95368447), 2.16667674 (20.95892361,18.88339973), 2.07552388	2.16667674	0.0224	(21.20447939,18.88339973)	(-0.0028,-0.0062)
0.5,0.5;	(20.50777157,18.13755257), 2.3702190 (21.24047608,18.87554525), 2.36493083 (21.33237918,18.98007636), 2.35230282	2.3702190		(21.33237918,18.13755257)	
0.5,0.7	(21.02585512,18.75805694), 2.26779818 (21.20807660,18.74610300), 2.46197360 (21.17374725,18.83492247), 2.33882478	2.46197360	0.0387	(21.20807660,18.74610300)	(-0.0058,0.0336)
0.5,0.5;	(20.49902469,18.26124651), 2.23777818 (20.96481681,18.73664022), 2.22817659 (20.88349184,18.6578795), 2.22561234	2.23777818		(20.96481681,18.26124651)	
0.5,0.9	(20.73003285,18.51522959), 2.21480326 (20.79102334,19.06184593), 1.72917741 (21.10611995,18.77582444), 2.33029551	2.33029551	0.0413	(21.10611995,18.51522959)	(0.0067,0.0139)
0.5,0.7;	(21.07272903,18.92358475), 2.14914428 (21.22263672,19.07290849), 2.14972823 (21.94809422,19.8124786), 2.13561562	2.14972823		(21.94809422,18.92358475)	
0.5,0.7	(21.20766460,19.04395475), 2.16370985 (21.67527243,19.32803827), 2.34723416 (21.24808362,19.23194686), 2.01613676	2.34723416	0.0919	(21.67527243,19.04395475)	(-0.0124,0.0064)
0.5,0.7;	(21.06299926,18.96166580), 2.10133346 (20.99612540,18.90940402), 2.08672138 (20.89190797,18.79462632), 2.09728165	2.10133346		(21.06299926,18.79462632)	
0.5,0.9	(21.06299926,18.96166580), 2.10133346 (21.14486023,19.03383207), 2.11102816 (20.97081131,18.94443618), 2.02637513	2.11102816	0.0046	(21.14486023,18.94443618)	(0.0039,0.0080)
0.5,0.9;	(21.07187149,19.05530223), 2.01656926 (21.47144309,19.46776927), 2.00367382 (21.53538586,19.5188166), 2.01656926	2.01656926		(21.53538586,19.05530223)	
0.5,0.9	(21.07187149,19.05530223), 2.01656926 (20.96386272,19.05676141), 1.90710131 (20.98121078,19.20919165), 1.77201913	2.01656926	0.000	(21.07187149,19.05530223)	(-0.0215,0.0000)
0.7,0.7;	(20.68911924,18.95978904), 1.72933020 (21.19261996,19.46275372), 1.72986624 (20.51473098,18.79119533), 1.72353565	1.72986624		(21.19261996,18.79119533)	
0.7,0.7	(20.40966135,18.82095110), 1.58871025 (20.94576076,19.16611058), 1.77965018 (20.81125866,18.79119533), 2.02006333	2.02006333	0.1678	(20.94576076,18.79119533)	(-0.0116,0.000)
0.7,0.7;	(20.81877137,18.87730388), 1.94146749 (21.16380888,19.22234139), 1.94146749 (21.15122336,19.21927655), 1.93194681	1.94146749		(21.16380888,18.87730388)	
0.7,0.9	(20.86338810,19.00905608), 1.85433202 (21.16380888,19.05970411), 2.10410477 (21.15034435,19.20734157), 1.94300278	2.10410477	0.0838	(21.16380888,19.00905608)	(0.0000,0.0070)
0.7,0.9;	(20.80739498,18.99871886), 1.80867612 (20.98980513,19.18305413), 1.80675100 (20.92202393,19.11519444), 1.80682949	1.80867612		(20.98980513,18.99871886)	
0.7,0.9	(21.05526368,19.10781918), 1.94744450 (21.18774940,19.18667070), 2.00107870 (21.04001795,19.14722973), 1.89278822	2.00107870	0.1064	(21.18774940,19.10781918)	(0.0094,0.0057)
0.9,0.9;	(20.91204704,19.48778477), 1.42426227 (21.02515636,19.59467160), 1.43048476 (20.73734536,19.30922966), 1.42811570	1.43048476		(21.02515636,19.30922966)	
0.9,0.9	(20.84498386,19.53269971), 1.31228415 (21.02515636,19.52891907), 1.49623729 (20.72131942,19.35763076), 1.36368866	1.49623729	0.0460	(20.84498386,19.35763076)	(-0.0086,0.0025)

6. Conclusions

In this paper, we considered a permutation that minimizes the maximum difference between the maximum value and the minimum one in each sum of components given by 3-dimensional vectors. We first formulated this problem as an integer programming problem. Then based on the method for the scalar case's balanced assignment problem we proposed an $O(n^5)$ approximation algorithm for the problem. Thirdly we presented the results from computational experiments using our algorithm and compared to the exact solutions. For each sum's problem, our algorithm did not give good results for all. But for another type problem, that is, to minimize the difference of the maximum value in $3n$ sums and the minimum one, it gave sufficient good approximate solutions. Finally we consider the reason our algorithm was not sufficient for each sum's problem.

For further research, we will propose more efficient algorithm for higher dimensional vector and consider vector case's multi-index assignment problem.

Acknowledgement

Dr. Mario Nakamori, emeritus professor of Tokyo University of Agriculture and Technology, has supported my studies. The problems extended to vector cases are according to his suggestion. I express deeply appreciation for his advice here.

References

[1] Pentico, D.W.: Assignment problems : A golden anniversary survey. European J. Oper. Res. **176**, 774–793 (2007)

[2] Du, D.-Z., Pardalos, P.M.(eds.): Handbook of Combinatorial Optimization. Kluwer (1999)

[3] Martello, S., Pulleyblank, W.R., Toth, P., de Werra, D. : Balanced Optimization Problems. Oper. Res. Lett. **3**, 275–278 (1984)

[4] Kamura, Y., Nakamori, M. : Combining Imperfect Components to Minimize the System's Error. Proc. PDPTA'01. 1277–1283 (2001)

[5] Kamura, Y., Nakamori, M., Shinano, Y. : Combining Imperfect Components (II) — The Case of Multidimensional Error. Proc. PDPTA'02. 228–232 (2002)

[6] Kamura, Y., Nakamori, M. : Combining Imperfect Components (III) — Minimax Optimization of Multidimensional Cost Error. Proc. PDPTA'04. 311–316 (2004)

[7] Kamura, Y., Nakamori, M. : A Simple Approximation Algorithm for the Modified Bottleneck Assignment Problem in Vector Case. International J. of Computer Theory and Engineering. **8**(2), 145–149 (2016)

[8] Kamura, Y., Nakamori, M. : Modified Balanced Assignment Problem in Vector Case: System Construction Problem. Proc. of The 2014 International Conf. on Comput. Sci. & Comput. Intelligence (CSCI ' 2014). **II**, 52–56, IEEE Comput. Soc. (2014)

[9] Mori, M., Matsui, T., : Operations Research (in Japanese). Asakura publishing (2004)

[10] Hopcroft, J.E., Karp, R.M.: An $n^{5/2}$ Algorithm for Maximum Matchings in Bipartite Graphs. SIAM J. Comput. **2**, 225–231 (1973)

[11] Burkard, R.E. : Selected topics on assignment problems. Discrete Appl. Math. **123**, 257–302 (2002)