A random search method for finding ‘$K \geq 2$’ number of ranked optimal solution to an assignment problem

Santosh Kumar$^{1,2}$ · Ali Al-Hasani$^{1,3}$ ·
Masar Al-Rabeeah$^{1,3}$ ·
Andrew Eberhard$^1$

Received: date / Accepted: date

Abstract A need for an optimal solution for a given mathematical model is well known and many solution approaches have been developed to identify efficiently an optimal solution in a given situation. For example, one such class of mathematical models with industrial applications have been classified as mathematical programming models (MPM). The main idea behind these models is to find the optimal solution described by those models. However, the same is not true for a ‘$K$’number of ranked optimal solutions, where $K \geq 2$. Mathematically, the $K^{th}$ best solution, $K \geq 2$, deals with determination of the 2$^{nd}$, 3$^{rd}$, 4$^{th}$ or in general the $K^{th}$ best solution. This $K^{th}$ best solution $K \geq 2$, suddenly becomes much more demanding with respect to computational requirements, which increases with the increase in the value of $K$. This paper first identifies difficulties associated with determination of ranked solutions and later develops a random search method to find ranked optimal solutions in the case of an assignment problem. We test the efficiency of the proposed approach by executing the random search method on a number of different size assignment problems.

Keywords Assignment model; $K^{th}$ best solution, $K \geq 2$; Random search method.

Mathematics Subject Classification (2010) 90C10 · 90C27 · 65K05

Santosh Kumar
Department of Mathematics and Statistics, University of Melbourne, Melbourne, Australia.
Tel.: +61411136612
E-mail: santosh.kumarau@gmail.com

Ali Al-Hasani
School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, Australia.

$^1$School of Mathematical and Geospatial Sciences, RMIT University, Melbourne, Australia.
$^2$Department of Mathematics and Statistics, University of Melbourne, Melbourne, Australia.
$^3$Department of Mathematics, College of Sciences, University of Basrah, Al- Basrah, Iraq.
1 Introduction

Usually, determination of the best solution is relatively an easy problem when one can identify optimality of a solution with the help of a few associated conditions. These conditions establish the optimality of the given solution in a given environment. However, for a $K$ ranked solution, $K \geq 2$ the same is not true as their ranked optimality can be linked with the previous $(K - 1)$ ranked optimal solutions. Thus computational requirement suddenly increases. A need for ranked optimal solution for a linear programming model was identified for solving an extreme point mathematical programming model [3, 9, 10, 15].

A large literature exist in OR books, where real life situations have been described by mathematical models and methods for their solution procedures have been described see Taha [23] and Kumar at el. [17]. Mathematical background, related difficulties and a characteristic equation for integer programming models have been presented in Section 2. In the case of an assignment problem, the characteristic equation that was established by Kumar, Munapo and Jones [12] for a pure integer programming model to find ranked optimal integer solutions is no longer of help as assignment models are free of fractional values. In Section 3 we develop a random search method for the assignment problem and find $K \geq 2$ number of ranked optimal solutions. Numerical illustrations and computational experiments have been presented in Section 4 and finally the paper is concluded in Section 5.

2 Mathematical background and difficulties associated with determination of ranked optimal solutions

2.1 What is a ranked optimal solution and why it is difficult to find?

Consider a typical linear programming (LP) model given by:

Maximize $Z = CX$

Subject to $AX \leq b; \quad X \geq 0.$

Many approaches exist to find an optimal solution to a given LP, which is an extreme point of the convex set generated by the given constraint set $AX \leq b,$ $X \geq 0.$ Let such an optimal solution be denoted by $X_0 \geq 0.$ However, if the solution vector $X_0 \geq 0$ is not acceptable for some other practical reasons, one may be interested to search for the 2nd best extreme point of the convex set $AX \leq b, X \geq 0,$ thus the 2nd best solution becomes of interest, which may be defined as the best solution among all possible solutions after excluding the best one. In general, the $(K + 1)^{th}$ best extreme point solution to a given LP model will be the best solution after excluding $K$ best extreme point solutions, $K = 1, 2, \ldots$ This seemingly simple problem can be computationally demanding and difficult. For example, if we are required to find the 2nd best extreme point of the convex set generated by set $AX \leq b, X \geq 0,$ we have
A random search method for finding ‘$K \geq 2$’ number of ranked optimal solutions will be the point which leads to minimum reduction in the total value of the objective compared to the best. The order of difficulty starts to increase when we are required to find the $K^{th}$ best, $K = 3, 4, \ldots$ as the $K^{th}$ best extreme point can be an extreme point that can be reached from the best (optimal) or the 2nd best and in general from the $(K - 1)^{th}$ best extreme point solutions of that given LP. Thus optimality of a solution to be graded as best is relatively an easy problem, as the required solution can be identified with the help from some local conditions of optimality. For example, for the LP model local conditions are non-negative requirement on values in the objective raw, when simplex method is used. However, ranked solution does not have any such local properties that can guarantee its ranked optimality. In other words, we may be at the required point but we have no clear means to identify and recognize its ranked optimality. Thus computational complexity start to rise when one is required to find the $K^{th}$ best, $K \geq 3$.

### 2.2 Need for a ranked optimal solution

A need for a ranked optimal solution for a linear programming can be realized by the extreme point mathematical programming model discussed by Kirby et al. [9, 10], Puri and Swarup [21] and Sherali and Dickey [22]. A mathematical model of an extreme point mathematical programming is given by:

Maximize $Z = CX$  \hspace{1cm} (2)

subject to

$AX \leq b; \quad b \geq 0, \quad X \geq 0$

and $X$ must be an extreme point of another set of linear constraints

$DX \leq d, \quad d \geq 0$.

For an application of the extreme point mathematical programming model (2), see, Chandrasakeran, Kumar and Wagner [3], and for linear programming based solution procedures, see Kumar and Wagner [15] and random search method was applied by Huynh and Kumar [7].

### 2.3 The $k^{th}$ best integer solution for an integer programming model

The model (1) becomes a pure integer programming model when the condition $X \geq 0$ is replaced by $X \geq 0$ and integer. In the case of such a pure integer programming model, even the optimal solution does not satisfy local optimality conditions similar to the LP model. Thus alternative approaches were developed to create an environment similar to a LP, i.e. the local optimality conditions become applicable even when all variables are restricted as $X \geq 0$ and integer. This object was achieved with the help of cuts, which
were developed to refine the LP convex space in such a way that the modified convex space resulted in an integer extreme point, where the local optimality conditions can once again establish the optimality of the solution. The convex space generated by the set of given constraints set \( AX \leq b, X \geq 0 \) was modified by the additional cut constraints to reduce the feasible space to converge to an integer extreme point to be identified by the LP local conditions. The process to deal with cuts was identified as slow and cuts also increase the size of the original model. Every cut adds an extra row and an extra variable to the original LP model. To overcome this difficulty, an alternative approach for a pure integer programming model (PIP) was suggested by Kumar, Munapo and Jones [12]. They developed a linear relation and called it a **descending hyper-plane**. Later it was renamed as a **characteristic equation** by Kumar, Munapo and Jones [13] and Kumar and Munapo [11]. The linear relation was derived from the LP output of the final simplex tableau. The characteristic equation they developed is a necessary but not a sufficient condition for the ranked integer solutions of a given integer program.

Murty [20] developed a solution method for finding the \( K \)th best solution of assignment problem \( K \geq 2 \). Murty’s approach was based on blocking, one at a time, the optimal assignment that is if \( i \)th job was assigned to \( j \)th person, i.e., he set \( c_{ij} = \infty \). The \( K \)th best solution has been discussed in context of the shortest route by Lawler [18]. An application of the kth best path in a graph has been attempted by Ishii [8]. Chen, McPhee and Kumar [4] have approach the \( K \)th best solution of assignment problem by developing some bounds with the help of the optimal solution. The \( K \)th best solution for a one-dimensional knapsack model has been presented by Yanasse, Soma and Maculan [24]. Lu and Rosenbaum [19] developed the method for ranking assignment solutions.

### 2.4 The characteristic equation of a pure integer program

Consider a pure integer programming (PIP) model given by relations (3).

\[
\begin{align*}
\text{Maximize } Z &= CX \\
\text{Subject to } AX &\leq b; \\
X &\geq 0 \quad \text{and integer.}
\end{align*}
\]

Assume (3) is a ‘\( m \)’ constraints and ‘\( n \)’ variable problem with all constraints inequalities of the form \( \leq \) type and all RHS elements of the vector \( b \geq 0 \). Ignoring the integer restriction on the variables, if the LP relaxation of (3) is solved using any available LP package by the usual simplex method, the final simplex tableau giving the optimal solution will look like, as illustrated in Table 1. The number of basic variables will be ‘\( m \)’and the number of non-basic will be ‘\( n \)’, after introducing ‘\( m \)’number of slack variables. Here without any loss of generality we have displayed basic variables in the first ‘\( m \)’columns and the non-basic variables as the remaining ‘\( n \)’columns.

Since this is an optimal solution, all elements in the RHS column and the objective row are non-negative and may be fractional or integer values, hence
A random search method for finding \( K \geq 2 \) number of ranked optimal

Table 1 A typical LP optimal solution output

| Variables | \( x_{B1} \) | \( x_{B2} \) | \ldots | \( x_{Bm} \) | \( x_{BN1} \) | \( x_{BN2} \) | \ldots | \( x_{BNn} \) | RHS |
|-----------|--------------|--------------|--------|----------------|----------------|----------------|--------|----------------|------|
| \( x_{B1} \) | 1 | 0 | \ldots | 0 | \( a_{b1,NB1} \) | \( a_{b1,NB2} \) | \ldots | \( a_{b1,NNn} \) | \( \beta_1 \) |
| \( x_{B2} \) | 0 | 1 | \ldots | 0 | \( a_{b2,NB1} \) | \( a_{b2,NB2} \) | \ldots | \( a_{b2,NNn} \) | \( \beta_2 \) |
| \ldots | \ldots | \ldots | \ldots | \ldots | \ldots | \ldots | \ldots | \ldots | \ldots |
| \( x_{Bm} \) | 0 | 0 | \ldots | 0 | \( a_{bm,NB1} \) | \( a_{bm,NB2} \) | \ldots | \( a_{bm,NNn} \) | \( \beta_m \) |
| \( Z_j - C_j \) | 0 | 0 | \ldots | 0 | \( Y_{NB1} \) | \( Y_{NB2} \) | \ldots | \( Y_{NNn} \) | \( Y_{LP opt} \) |

may not form an acceptable solution to the PIP given by the model (3). Since the integer feasible points are within the feasible region, the non-basic variables values for LP solution, which are zero, have to be increased. This is the basic concept behind the characteristic equation and how to control the increase in values of these non-basic values is the second important concept. These two guiding factors are displayed by the characteristic equation obtained by using the elements in the last row of Table 1. One can develop the following equation:

\[
Z + Y_{NB1}x_{NB1} + Y_{NB2}x_{NB2} + \cdots + Y_{NNn}x_{NNn} = Y_{LP opt} \quad (4)
\]

As all elements in the LHS of the equation (4) are zero as non-basic variables, resulting in the RHS as the value of \( Z \), the objective function value. Rewrite (4) by taking the lowest common factor and remove all fractions, as shown in equation (5). Let the lowest common factor be denoted \( C \), then one can write (4) as:

\[
\left( \frac{C}{C} \right) Z + \left( \frac{\gamma_{NB1}x_{NB1} + \gamma_{NB2}x_{NB2} + \cdots + \gamma_{NNn}x_{NNn}}{C} \right) = R + iC \quad (5)
\]

The characteristic equation is given by:

\[
\gamma_{NB1}x_{NB1} + \gamma_{NB2}x_{NB2} + \cdots + \gamma_{NNn}x_{NNn} = R + iC \quad (6)
\]

Where \( i = 0, 1, 2, \ldots \) Note \( R \) is the remainder and \( C \) is the lowest common factor.

For better understanding, here is a simple numerical illustration of a characteristic equation. Consider the pure integer programming problem:

\[
\text{Max } Z = 8x_1 + 4x_2,
\]

subject to

\[
x_1 + x_2 \leq 5, \quad 9x_1 + 4x_2 \leq 40, \quad x_1, x_2 \geq 0 \text{ and integers.}
\]

The LP relaxed solution for the non-negative \( x_j \) is given in Table 2.
Table 2 LP relaxed solution

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| Z | $x_1$ | $x_2$ | $S_1$ | $S_2$ | RHS |
| 1 | 0   | 0   | 4/5 | 4/5 | 196/5 |
| 0 | 0   | 1   | 9/5 | −1/5 | 21/5 |
| 0 | 1   | 0   | −4/5 | 1/5 | 14/5 |

The LP solution $x_2 = 21/5$, $x_1 = 14/5$ and $Z = 196/5$ is not an acceptable solution to PIP, hence there is a need for the characteristic equation. From Table 2, it is given by:

$$4s_1 + 4s_2 = 1 + 5i, \quad i = 0, 1, 2, \ldots, 39.$$  

Note the residue is 1 and the lowest common factor is 5 and $s_1$, $s_2$ are also integer restricted values for non-basic variables in the PIP model. This characteristic equation does not have integer solution for $s_1$ and $s_2$ for $i = 0, 1$ and 2. However, for $i = 3$, there are many solutions like $s_1 = 4$, $s_2 = 0$ or $s_1 = 2$, $s_2 = 2$ or $s_1 = 1$, $s_2 = 3$ or $s_1 = 3$, $s_2 = 1$. From all these possibilities, only the solution $s_1 = 2$, $s_2 = 2$ results in $x_1 = 4$, $x_2 = 1$ and $Z = 36$.

It may be noted that effectiveness of the characteristic equation is a function of the value of $C$, the lowest common factor, which in the above example was 5 and the remainder value was 1. Thus we investigated the characteristic equation for RHS values of 1 for $i = 0, 6$ for $i = 1, 11$ for $i = 2$ and finally we found an integer solutions for non-basic variables when RHS value is 16 for $i = 3$. Note that the characteristic equation will not be effective when the lowest common factor (LCF) is very small quantity and remainder is zero.

For an assignment problem $R = 0$ and LCF = 1; hence CE approach is no longer going to help for finding the ranked optimal solution for an assigning problem. The assignment is essentially an integer programming problem, which generally has been classified as difficult problem. For more ideas, see Kumar et.al [16].

3 Random search method to find the ranked optimal solution for an assignment model

3.1 The Random search method

The random search method is intended to approximately estimate a near optimal solution for multidimensional function when it is difficult to establish optimality for a solution. It was first applied in the area of experimental design; the motivating factor was to find solutions to problems where traditional approach become intractable. Random search method has been described by Brooks [2] and its characteristics have been studied by Kumar and Scramozzino [14]. Random search method has been applied by Huynh and Kumar [7]. Essential elements for an application of a random search method are:
1. An estimate on the number of possible solutions for estimating the probability of a successful solution in the feasible domain, which in the case of an $n$ by $n$ assignment is given by $n(n-1)$ for finding the second best solution. Here successful solution means that a solution picked up randomly may be the required ranked solution with a given probability $p = \frac{1}{n(n-1)}$. For the $K^{th}$ best this value will be $p = \frac{1}{(k-1)(n(n-1))}$.

2. The value of assurance factor denoted by $(PR)$, is a given probability associated with the obtained solution. Here, it has been taken as 0.80 in Section 4. Computational load will increase for higher value of this assurance factor $(PR)$.

3. Some rule to stop the search, which is given by the value of number of searches $(NS)$, as a function of $PR$ and $p$. $NS$ can be calculated by

$$NS = \frac{\ln(1 - PR)}{\ln \left(1 - \frac{1}{(k-1)(n(n-1))}\right)}$$

Dantzig[6] has discussed the case of a 70 by 70 assignment model and emphasized that the number of possible solutions for that assignment problem is given by 70!, which is a large number that can easily become beyond capabilities of the modern computers. Thus mathematical methods that can provide to real life problems in realistic time-frame, some real solutions for benefit of the mankind play very important role.

### 3.2 Pseudo-code for the random search method

In this section, we present an algorithm to find ranked optimal solutions for an assignment model by using the random search method. This approach has been labelled as Algorithm[4]
Algorithm 1 A random search method for finding "$K^{th}$" number of ranked optimal solutions

**Step 1.**
1.1 Find the optimal solution $Z_{\text{opt}}$.
1.2 $Z_{\text{ranked}}$ is a set of the value of $K^{th}$ best solution;
   initially $Z_{\text{ranked}} = \{Z_{\text{opt}}\}$.

**Step 2:**
2.1 Number of ranked solution required, say ‘$L \geq 2$’ such that $L$
   is a given integer value.
2.2 Assign an integral value to the number of points in the feasible domain,
   which for the $(n \times n)$ assignment problem is $n(n-1)$ and therefore
   $p = \frac{1}{n(n-1)}$.
2.3 The value of PR, the probability of assurance associated with
   the ranked optimal solutions; $0 < PR < 1$. It is a given value.
2.4 $S = 0$; counter.
2.5 $K = 2$; 2nd best solution

**Step 3.**
Find the $K^{th}$ best solution.

**Step 4.**
$Z^* = \emptyset$ ($Z^*$ is a set of $Z_S$ values).

**Step 5.**
5.1 Find number of random searches (NS) from the equation:
   $$NS = \frac{\ln(1 - PR)}{\ln\left(1 - \frac{p}{(k-1)}\right)};$$
   NS is an integer number.
5.2 Let $X_{\text{rand}} = \emptyset$ ($X_{\text{rand}}$ is a set of $x_{ij}$).

**Step 6.**
Search NS time to find other solution by random search method:
6.1 Do $S = S + 1$
6.2 Chose rand($x_{ij}$)$_S$.
6.3 If $x_{ij} \in X_{\text{rand}}$ then ignore rand($x_{ij}$)$_S$ GoTo Step 6.2, else continue
6.4 Add $x_{ij} \rightarrow X_{\text{rand}}$
6.5 rand($x_{ij}$)$_S \rightarrow x_{ij}^*$; ($x_{ij}^*$) is optimal solution.
6.6 Find $Z_S$; Add $Z_S \rightarrow Z^*$
6.7 If $S = (NS + 1)$ GoTo 7. Else GoTo Step 6.

**Step 7.**
Rank $Z^*$ to find the best $Z^K$; add $Z^K \rightarrow Z_{\text{ranked}}$.

**Step 8.**
If $K = L$ GoTo Step 9. Else $K = K + 1$ Goto Step 3

**Step 9.**
Rank $Z_{\text{ranked}}$

**Step 10.** stop

4 Numerical illustrations and computational study

In this section, a numerical illustration has been discussed and the proposed
algorithm has been implemented to find the $K$-number of ranked optimal
solutions on several assignment problems, which were generated randomly in
different sizes.

4.1 An illustrative example

Consider an assignment problem

$$\min Z = \sum_{i=1}^{4} \sum_{j=1}^{4} C_{ij}x_{ij} \quad (8)$$
A random search method for finding \( K \geq 2 \) number of ranked optimal

Subject to

\[
\sum_{i=1}^{4} x_{ij} = 1; \quad j = 1, 2, \ldots, 4
\]
\[
\sum_{j=1}^{4} x_{ij} = 1; \quad i = 1, 2, \ldots, 4
\]

All \( x_{ij} = 0 \) or 1. \( j = 1, 2, \ldots, 4 \) and \( i = 1, 2, \ldots, 4 \).

Such that:

\[
C_{ij} = \begin{bmatrix}
5 & 2 & 9 & 6 \\
1 & 10 & 8 & 2 \\
2 & 3 & 8 & 9 \\
5 & 7 & 3 & 1
\end{bmatrix}
\]

We arbitrarily take \( PR = 0.80; \ K = 2 \) and find

\[
p = \frac{1}{(K-1)n(n-1)} = \frac{1}{12}.
\]

Then, the number of searches \( NS \) is obtained from the equation:

\[
NS = \left\lfloor \frac{\ln(1 - PR)}{\ln(1 - p)} \right\rfloor = \left\lfloor \frac{\ln(1 - 0.8)}{\ln \left( 1 - \frac{1}{12} \right)} \right\rfloor = 18;
\]

where \( \lfloor \cdot \rfloor \) is nearest integer function.

Step 1. The optimal solution of assignment problem is:

|   | J1 | J2 | J3 | J4 |
|---|----|----|----|----|
| S1 | 4  | 0  | 0  | 4  |
| S2 | 0  | 8  | 4  | 0  |
| S3 | 0  | 0  | 3  | 6  |
| S4 | 5  | 6  | 0  | 0  |

Step 2. The first random search selects a random number which represent the variable \( x_{14} \) excluding the optimal allocations \( x_{12}, x_{24}, x_{31} \) and \( x_{43} \). Shown in red. A feasible allocation in the cell \( x_{14} \) will also require a change in \( x_{24} \). The
new allocation will be \(x_{14}, x_{22}, x_{31}\) and \(x_{43}\) which is shown in next table.

\[
\begin{array}{cccc}
  & J1 & J2 & J3 & J4 \\
 S1 & 4 & 8 & 0 & 0 \\
 S2 & 0 & 0 & 4 & 4 \\
 S3 & 0 & 0 & 3 & 6 \\
 S4 & 5 & 6 & 0 & 0 \\
\end{array}
\]

The solution and \(Z\) from above table is: \(X = x_{14}, x_{22}, x_{31}, x_{43}, Z_{1,2} = 21\)

We continue this process for NS number of times to compute \(Z_{1,3}, Z_{1,4}, \ldots, Z_{1,18}\).

Step 3. Then, we arrange the \(Z_{1,2}, Z_{1,3}, Z_{1,4}, \ldots, Z_{1,18}\) in increasing order and find the 2\(^{nd}\) best solution. Select the solution which corresponds to minimum \(Z_{1,2}, Z_{1,3}, \ldots, Z_{1,18}\) and label that solution as the second best. This solution was \(Z_{13} = 13\) at \(x_{11} = x_{24} = x_{32} = x_{43} = 1\). Thus we have \(Z_{\text{opt}} = 9\) at \(x_{12} = x_{24} = x_{31} = x_{43} = 1\) and second best is \(Z_{13} = 13\) at \(x_{11} = x_{24} = x_{32} = x_{43} = 1\). A summary of these 18 search of random searches is given in Table 3.

**Table 3** Ranked optimal solution for illustrative example

| \(Z\)    | \(X\)          |
|----------|----------------|
| \(Z^* = 9\) | \(x_{12}, x_{24}, x_{31}, x_{43}\) |
| \(Z_{13} = 13\) | \(x_{11}, x_{24}, x_{32}, x_{43}\) |
| \(Z_{15} = 13\) | \(x_{12}, x_{23}, x_{31}, x_{44}\) |
| \(Z_{14} = 15\) | \(x_{12}, x_{24}, x_{31}, x_{43}\) |
| \(Z_{17} = 15\) | \(x_{13}, x_{24}, x_{31}, x_{42}\) |
| \(Z_{16} = 17\) | \(x_{12}, x_{24}, x_{33}, x_{41}\) |
| \(Z_{12} = 21\) | \(x_{14}, x_{22}, x_{31}, x_{43}\) |

Step 4. We chose \(Z_{13}\) as a second best, then return to steps 2 and 3 again to find the 3\(^{rd}\) best optimal solution. Thus we continue and find five ranked solution as given in Table 4.
A random search method for finding \( K \geq 2 \) number of ranked optimal solutions

Table 4: Optimal solution for the illustrative example

| \( Z^K_{th} \) | \( X \) |
|----------------|------------------|
| \( Z^9 \) = 9  | \( x_{12}, x_{24}, x_{31}, x_{43} \) |
| \( Z^{12} \) = 12 | \( x_{12}, x_{21}, x_{33}, x_{44} \) |
| \( Z^{13} \) = 13 | \( x_{14}, x_{21}, x_{31}, x_{43} \) or \( x_{12}, x_{23}, x_{33}, x_{44} \) or \( x_{11}, x_{24}, x_{32}, x_{43} \) |
| \( Z^{14} \) = 14 | \( x_{13}, x_{21}, x_{32}, x_{44} \) |
| \( Z^{15} \) = 15 | \( x_{12}, x_{21}, x_{34}, x_{43} \) |

4.2 Computational study

As an effective algorithm, we generated 60 assignment problems of different sizes: 4 \( \times \) 4, 5 \( \times \) 5, 6 \( \times \) 6, 7 \( \times \) 7, 8 \( \times \) 8, 9 \( \times \) 9, 10 \( \times \) 10, 15 \( \times \) 15, 20 \( \times \) 20, 30 \( \times \) 30, 40 \( \times \) 40 and 50 \( \times \) 50, all values of the coefficient of the objective function were randomly generated between [20, 2500]. We assumed the assurance probability as 0.80. The proposed algorithm was implemented in C, using Cplex 12.6 to read lp files, which represents the assignment models, and run on HP laptop Via VMware Workstation 14 player (Lubuntu 16.04 LTS x64) with 1 GB of RAM memory. A summary of these experiments is presented in Tables 5, 6, 7, 8.

Table 5: Test instances size 16, 25 and 36

| Class | \( Z_{opt} \) | 2nd best | NS. | 3rd best | NS. | 4th best | NS. | 5th best | NS. | CPU time (Sec.) |
|-------|-------------|----------|-----|----------|-----|----------|-----|----------|-----|-----------------|
| ASSP4-01 | 2093 | 2595 | 18 | 3574 | 37 | 3588 | 57 | 3893 | 76 | 0.0077 |
| ASSP4-02 | 2521 | 3550 | 18 | 3711 | 37 | 3736 | 57 | 4521 | 76 | 0.0071 |
| ASSP4-03 | 3216 | 3715 | 18 | 3817 | 37 | 3864 | 57 | 3951 | 76 | 0.0094 |
| ASSP4-04 | 4101 | 4566 | 18 | 4621 | 37 | 4637 | 57 | 4916 | 76 | 0.0091 |
| ASSP4-05 | 1332 | 1844 | 18 | 1961 | 37 | 2330 | 57 | 2400 | 76 | 0.0068 |
| ASSP5-01 | 3895 | 4177 | 31 | 4235 | 63 | 4406 | 95 | 4555 | 127 | 0.0104 |
| ASSP5-02 | 2944 | 3222 | 31 | 4198 | 63 | 4312 | 95 | 4368 | 127 | 0.0081 |
| ASSP5-03 | 3324 | 4011 | 31 | 4248 | 63 | 4338 | 95 | 4660 | 127 | 0.0100 |
| ASSP5-04 | 1881 | 1920 | 31 | 2523 | 63 | 2858 | 95 | 2888 | 127 | 0.0123 |
| ASSP5-05 | 2931 | 3152 | 31 | 3472 | 63 | 3799 | 95 | 3824 | 127 | 0.0119 |
| ASSP6-01 | 2127 | 2493 | 47 | 2510 | 95 | 3192 | 144 | 3447 | 192 | 0.0135 |
| ASSP6-02 | 2679 | 3079 | 47 | 3223 | 95 | 3747 | 144 | 3872 | 192 | 0.0128 |
| ASSP6-03 | 2874 | 3467 | 47 | 3597 | 95 | 3599 | 144 | 3726 | 192 | 0.0070 |
| ASSP6-04 | 3110 | 3514 | 47 | 3514 | 95 | 3848 | 144 | 3907 | 192 | 0.0193 |
| ASSP6-05 | 3706 | 3783 | 47 | 3800 | 95 | 4003 | 144 | 4017 | 192 | 0.0087 |
Table 6  Test instances size 49, 64 and 81

| Class   | $Z_{Opt}$ | 2nd best | NS. | 3rd best | NS. | 4th best | NS. | 5th best | NS. | CPU time (Sec.) |
|---------|-----------|----------|-----|----------|-----|----------|-----|----------|-----|-----------------|
| ASSP7_7-1 | 1407      | 1632     | 66  | 1711     | 134 | 1755     | 201 | 1913     | 269 | 0.0091         |
| ASSP7_7-2 | 2538      | 2674     | 66  | 2835     | 134 | 2960     | 201 | 3257     | 269 | 0.0117         |
| ASSP7_7-3 | 3026      | 3289     | 66  | 3464     | 134 | 3700     | 201 | 3704     | 269 | 0.0108         |
| ASSP7_7-4 | 2698      | 3180     | 66  | 3371     | 134 | 3428     | 201 | 3849     | 269 | 0.0068         |
| ASSP7_7-5 | 4030      | 4314     | 66  | 4366     | 134 | 4479     | 201 | 4815     | 269 | 0.0057         |
| ASSP8_7-1 | 2261      | 2267     | 89  | 2280     | 179 | 2286     | 269 | 2387     | 359 | 0.0096         |
| ASSP8_7-2 | 2297      | 2485     | 89  | 2516     | 179 | 2653     | 269 | 2684     | 359 | 0.0114         |
| ASSP8_7-3 | 2754      | 2760     | 89  | 2962     | 179 | 3208     | 269 | 3253     | 359 | 0.0079         |
| ASSP8_7-4 | 2983      | 3193     | 89  | 3113     | 179 | 3475     | 269 | 3542     | 359 | 0.0077         |
| ASSP8_7-5 | 2983      | 3193     | 89  | 3113     | 179 | 3475     | 269 | 3542     | 359 | 0.0116         |
| ASSP9_7-1 | 2822      | 3158     | 115 | 3214     | 236 | 3299     | 346 | 3317     | 462 | 0.0112         |
| ASSP9_7-2 | 4565      | 4653     | 115 | 4675     | 236 | 4729     | 346 | 4735     | 462 | 0.0125         |
| ASSP9_7-3 | 2967      | 3056     | 115 | 3172     | 179 | 3207     | 346 | 3456     | 462 | 0.0114         |
| ASSP9_7-4 | 2983      | 3193     | 115 | 3113     | 179 | 3475     | 269 | 3542     | 359 | 0.0077         |
| ASSP9_7-5 | 3608      | 3943     | 115 | 3970     | 236 | 3977     | 346 | 3978     | 462 | 0.0120         |

Table 7  Test instances size 100, 225 and 400

| Class   | $Z_{Opt}$ | 2nd best | NS. | 3rd best | NS. | 4th best | NS. | 5th best | NS. | CPU time (Sec.) |
|---------|-----------|----------|-----|----------|-----|----------|-----|----------|-----|-----------------|
| ASSP10_10-1 | 3351      | 3408     | 144 | 3431     | 288 | 3452     | 433 | 3509     | 578 | 0.0079         |
| ASSP10_10-2 | 4754      | 4960     | 144 | 4998     | 288 | 5048     | 433 | 5155     | 578 | 0.0078         |
| ASSP10_10-3 | 3912      | 3949     | 144 | 4106     | 288 | 4251     | 433 | 4292     | 578 | 0.0226         |
| ASSP10_10-4 | 3212      | 3214     | 144 | 3317     | 288 | 3349     | 433 | 3380     | 578 | 0.0076         |
| ASSP10_10-5 | 4887      | 5243     | 144 | 5363     | 288 | 5370     | 433 | 5523     | 578 | 0.0093         |
| ASSP15_15-1 | 4583      | 4666     | 337 | 4675     | 675 | 4693     | 1013| 4696     | 1351| 0.0095        |
| ASSP15_15-2 | 4086      | 4369     | 337 | 4414     | 675 | 4510     | 1013| 4538     | 1351| 0.0144        |
| ASSP15_15-3 | 4624      | 4884     | 337 | 4911     | 675 | 5002     | 1013| 5016     | 1351| 0.0165        |
| ASSP15_15-4 | 4165      | 4241     | 337 | 4322     | 675 | 4398     | 1013| 4516     | 1351| 0.0208        |
| ASSP15_15-5 | 4562      | 4565     | 337 | 4639     | 675 | 4662     | 1013| 4712     | 1351| 0.0150        |
| ASSP20_20-1 | 4500      | 4553     | 610 | 4556     | 1222| 4609     | 1833| 4631     | 2445| 0.0256        |
| ASSP20_20-2 | 3280      | 3386     | 610 | 3394     | 1222| 3423     | 1833| 3439     | 2445| 0.0219        |
| ASSP20_20-3 | 4459      | 4561     | 610 | 4624     | 1222| 4664     | 1833| 4732     | 2445| 0.0143        |
| ASSP20_20-4 | 4862      | 4863     | 610 | 4960     | 1222| 5011     | 1833| 5012     | 2445| 0.0290        |
| ASSP20_20-5 | 4924      | 4976     | 610 | 5103     | 1222| 5149     | 1833| 5189     | 2445| 0.0164        |
A random search method for finding \( K \geq 2 \) number of ranked optimal

Table 8 Test instances size 900, 1600 and 2500

| Class  | \( Z_{opt} \) | 2nd best | NS. | 3rd best | NS. | 4th best | NS. | 5th best | NS. | CPU time (Sec.) |
|--------|-------------|----------|-----|----------|-----|----------|-----|----------|-----|-----------------|
| ASSP30_30-1 | 3276 | 3319 | 1399 | 3353 | 2799 | 3363 | 4199 | 3396 | 5600 | 0.0311 |
| ASSP30_30-2 | 4610 | 4680 | 1399 | 4684 | 2799 | 4685 | 4199 | 4754 | 5600 | 0.0377 |
| ASSP30_30-3 | 4117 | 4161 | 1399 | 4203 | 2799 | 4213 | 4199 | 4216 | 5600 | 0.0493 |
| ASSP30_30-4 | 4947 | 5027 | 1399 | 5050 | 2799 | 5070 | 4199 | 5130 | 5600 | 0.0248 |
| ASSP30_30-5 | 4680 | 4747 | 1399 | 4752 | 2799 | 4813 | 4199 | 4880 | 5600 | 0.0238 |
| ASSP40_40-1 | 4852 | 4901 | 2509 | 4904 | 5020 | 4940 | 7531 | 4983 | 10042 | 0.0470 |
| ASSP40_40-2 | 4621 | 4692 | 2509 | 4699 | 5020 | 4711 | 7531 | 4724 | 10042 | 0.0476 |
| ASSP40_40-3 | 3744 | 3757 | 2509 | 3770 | 5020 | 3803 | 7531 | 3804 | 10042 | 0.0418 |
| ASSP40_40-4 | 4682 | 4688 | 2509 | 4726 | 5020 | 4732 | 7531 | 4749 | 10042 | 0.0457 |
| ASSP40_40-5 | 4601 | 4668 | 2509 | 4732 | 5020 | 4766 | 7531 | 4788 | 10042 | 0.0534 |
| ASSP50_50-1 | 4984 | 4997 | 3942 | 5005 | 7885 | 5018 | 11828 | 5055 | 15771 | 0.0509 |
| ASSP50_50-2 | 5168 | 5206 | 3942 | 5294 | 7885 | 5297 | 11828 | 5326 | 15771 | 0.0556 |
| ASSP50_50-3 | 5203 | 5213 | 3942 | 5226 | 7885 | 5236 | 11828 | 5251 | 15771 | 0.0513 |
| ASSP50_50-4 | 4402 | 4488 | 3942 | 4541 | 7885 | 4558 | 11828 | 4625 | 15771 | 0.0512 |
| ASSP50_50-5 | 5033 | 5099 | 3942 | 5111 | 7885 | 5116 | 11828 | 5133 | 15771 | 0.0666 |

5 Concluding remarks

In this paper, a random search methods to find ranked optimal solution for assignment problem has been established. Necessary code has been developed and implemented to test the proposed method on several instances. It may be noted that:

1. The characteristic equation for the general PIP is not applicable for this uni-model assignment problem.
2. Computational complexity of the proposed approach with existing methods [4, 20] is not possible as these authors have not provided computational results.
3. The CPU time of proposed algorithm is insignificantly low for identifying the ranked solutions.

Future studies will be to extend the random search method to find all non-dominated points for bi-objective integer programming problems [1], and also apply the random search method for solving other uni-modular problems.

References

[1] Al-Hasani, Ali, Al-Rabeeah, Masar, Kumar, Santosh, and Eberhard, Andrew. “Finding all non-dominated points for a bi-objective generalized assignment problem”. In: The International Conference on Mathematics: Pure, Applied and Computation (ICoMPAC 2018). 2018.
[2] Brooks, Samuel H. “A discussion of random methods for seeking maxima”. In: Operations Research 6.2 (1958), pp. 244–251.

[3] Chandrasekaran, R, Kumar, S, and Wagner, D. “Critical Path Problem under Assignment Constraint: An Application of an Extreme Point Mathematical Programming Problem”. In: Journal of Information and Optimization Sciences 1.1 (1980), pp. 41–51.

[4] Chen, S, McPhee, J, and Kumar, S. “Optimization Techniques and Applications”. In: Optimization Techniques and Applications. Ed. by K.H. Phua, C.M. Wang, W.Y. Yeong, T.Y. Leong, H.T. Loh, K.C. Tan, and F.S. Chou. World Scientific, 1992. Chap. Determination of kth best in an assignment problem, pp. 1110–1118.

[5] CPLEX, IBM ILOG. “12.6”. In: CPLEX Users Manual (2014).

[6] Dantzig, George B. “Linear programming”. In: Operations Research 50.1 (2002), pp. 42–47.

[7] Huynh, H-N and Kumar, Santosh. “A random search method for extreme point mathematical programming.” In: Asia-Pacific Journal of Operational Research 7.1 (1990), pp. 30–45.

[8] Ishii, Hiroaki. “A new method finding the k-th best path in a graph”. In: Journal of the Operations Research Society of Japan 21.4 (1978), pp. 469–476. doi: 10.15807/jorsj.21.469.

[9] Kirby, MJL, Love, HR, and Swarup, Kanti. “Extreme point mathematical programming”. In: Management Science 18.9 (1972), pp. 540–549.

[10] Kirby, MJL, Love, HR, and Swarup, Kanti. “Extreme point programming with nonlinear constraints”. In: Discrete Mathematics 5.4 (1973), pp. 345–365.

[11] Kumar, S and Munapo, E. “Some Lateral Ideas and their applications for developing new solution procedures for a pure integer programming model, Keynote address”. In: Proc. of Herbal International Conference on Applications of Mathematics and Statistics for Intelligent solutions through Mathematics and Statistics, Edited by Marriappan, Srinivasan and Amritraj, Excell India Publisher. 2012, pp. 13–21.

[12] Kumar, S, Munapo, E, and Jones, BC. “An integer equation controlled descending path to a protean pure integer program”. In: Indian Journal of Mathematics 49.2 (2007), pp. 211–237.

[13] Kumar, S, Munapo, E, and Jones, BC. “Descending Hyper-Plane Approach for Solving A Binary Integer Program”. In: International Journal of Mathematical Modeling, Simulation and Applications 2.1 (2009), pp. 53–62.

[14] Kumar, S. and Scarmozzino, F. “Characteristics and applications of random search method for mathematical programming”. In: Investigação Operacional 18 (1998), pp. 167–181.

[15] Kumar, S and Wagner, D. “Some algorithms for solving extreme point mathematical programming problems”. In: New Zealand Journal of Operational Research 7.2 (1979), pp. 127–149.
A random search method for finding \( K \geq 2 \) number of ranked optimal

[16] Kumar, S, Luhandjula, M, Munapo, E, and Jones, B. “Fifty years of integer programming: a review of the solution approaches”. In: Asia Pacific Business Review 6.3 (2010), pp. 5–15.

[17] Kumar, Santosh, Munapo, E, Lesaoana, M, and P, Nyamugure. Some Innovations in OR Methodology: linear Optimization. Lambert Academic Publishing, 2018.

[18] Lawler, Eugene L. “A procedure for computing the k best solutions to discrete optimization problems and its application to the shortest path problem”. In: Management science 18.7 (1972), pp. 401–405.

[19] Lu, B and Rosenbaum, PR. “An Algorithm for Ranking all the Assignments in Order of Increasing Cost”. In: Journal of Computational and Graphical Statistics 13.2 (2004), pp. 422–434.

[20] Murty, Katta G. “Letter to the editor: An algorithm for ranking all the assignments in order of increasing cost”. In: Operations research 16.3 (1968), pp. 682–687.

[21] Puri, MC and Swarup, Kanti. “Strong-cut enumerative procedure for extreme point mathematical programming problems”. In: Mathematical Methods of Operations Research 17.3 (1973), pp. 97–105.

[22] Sherali, Hanif D and Dickey, S Elizabeth. “An extreme-point-ranking algorithm for the extreme-point mathematical programming problem”. In: Computers & operations research 13.4 (1986), pp. 465–475.

[23] Taha, Hamdy A. Operations research: An introduction (for VTU). Pearson Education India, 2005.

[24] Yanasse, Horacio Hideki, Soma, Nei Yoshihiro, and Maculan, Nelson. “An algorithm for determining the k-best solutions of the one-dimensional knapsack problem”. In: Pesquisa Operacional 20.1 (2000), pp. 117–134.