Optimization of fuzzy multi-company workers assignment problem with penalty using genetic algorithm

N. Shahsavari Pour

Department of Industrial Engineering,

Science and Research Branch, Islamic Azad University, Kerman, Iran

Email: shahsavari_n@alum.sharif.edu

*M. Esmaeili

Department of Mathematics and

Computer Science, Shahid Bahonar University , Kerman, Iran.

E-mail: m_esmaeili90@yahoo.com

*Corresponding Author

R. Esmaeili

Department of Industrial Engineering,

Science and Research Branch, Islamic Azad University, Kerman, Iran

***Abstract*— In this paper, we proposed the fuzzy multi-job and multi-company workers assignment problem with penalty. Our purpose is obtaining the optimal solution the assignment problem, where *n* jobs are assigned to *m* workers (*m*>*n*), each job must be assigned to one and only one worker and each worker could be received one job or do not receive any job. Furthermore, there are *k* company where each worker belong a special company. For finding the optimal assignment, we must optimize total cost this problem assignment. This problem has three types of costs, direct cost company cost and penalty. In this paper, first the proposed assignment problem is formulated to the crisp model by using a suitable fuzzy ranking and fuzzy arithmetic operators. Finally, a heuristic genetic algorithm is designed for solving the proposed problem and an example is given to verify the efficiency of the algorithm.**

Keywords: Fuzzy set; Assignment problem; Genetic algorithm; fuzzy ranking.

I. INTRODUCTION

One important problem in the industrial and service systems is assignment problem. So, optimization the total cost of the assignment problem, where *n* jobs are assigned to *m* workers is an important purpose in the industrial.

Furthermore, for each assignment problem (AP) there are various extensions. In this research the assignment problem extended to the fuzzy multi-job and multi-company workers assignment problem with penalty. In the proposed assignment problem there are *n* jobs and *m* workers, $m > n$ and each job must be assigned to one and only one worker, but each worker could be received one job or do not receive any job. Furthermore, there are *k* various company where each worker could be selected from the *sth* company ($s = 1,2,..,k$). Another extension of assignment problem is the AP with penalty (APP) where each job or worker are associated with a cost when the job is not assigned to any worker or the worker do not receive any job. Such a cost is called as the penalty with respect to the job or the worker (In The paper for the workers). The proposed problem has three types of costs, direct cost, company cost and penalty. The *direct cost* (c_{ij}) is related to the assigning the job j to the worker i. The *company cost* (d_s) is related to the selecting a worker from the *sth* company and the *penalty* (g_i) to each worker. This proposed problem called FCAPP.

In the real word possible all of the assignment problem costs aren't crisp. In this problem (FCAPP), some costs of problem are characterized by uncertain information such as fuzzy variables [2,5].

In this researcher, first the fuzzy multi-job and multi-company workers assignment problem is formulated to the crisp model then using the fuzzy ranking and Zadeh's extension principle. Furthermore, the crisp equivalents costs are characterized by trapezoidal fuzzy numbers finally, we designed a heuristic genetic algorithm for obtaining the proposed fuzzy programming.

This paper is organized as follows. Firstly, the concepts of the fuzzy multi-job and multi-company workers assignment problem and the minimum costs are introduced and then the mathematical model of the proposed problem is formulated in Section 2, the proposed problem is formulated to the crisp model in Section 3. We designed a heuristic genetic algorithm for solving the proposed programming models in Section 4, and given an example to show the result of the heuristic algorithm in Section 5. Finally, the remarking conclusion is given in Section 6.

II. PRELIMINARIES

A. Fuzzy arithmetic operators and ranking

The fuzzy set theory was initialized by Zadeh [15]. We give some basic concepts and results of fuzzy numbers, fuzzy arithmetic and ranking of fuzzy numbers which are needed in the rest of the paper (taken from [6]). A fuzzy set is defined as a subset $\tilde{a}$ of universal set $X \subseteq \mathbb{R}$ by its membership function $\mu_{\tilde{a}}(.)$, which assigns to each element, $X \in \mathbb{R}$, a real number $\mu_{\tilde{a}}(.)$ in the interval $[0,1]$.

Trapezoidal fuzzy number: A fuzzy number $\tilde{a} = (a_1, a_2, \alpha, \beta)$ is said to be a trapezoidal fuzzy number, if its membership function is given by function:

$$\mu_{\tilde{a}}(x) = \begin{cases} \dfrac{x-(a_1-\alpha)}{\alpha} & if \;\; a_1-\alpha \le x \le a_1 \\ 1 & if \quad a_1 \le x \le a_2 \\ \dfrac{(a_2+\beta)-x}{\beta} & if \; a_2 \le x \le a_2+\beta \\ 0 & o.w \end{cases}$$

Now, we define arithmetic on trapezoidal fuzzy numbers. Let $\tilde{a} = (a_1, a_2, \alpha, \beta)$ and $\tilde{b} = (b_1, b_2, \alpha', \beta')$ be two trapezoidal fuzzy numbers. Define

$$x > 0, \quad x \in \mathbb{R}; \quad x\tilde{a} = (xa_1, xa_2, x\alpha, x\beta)$$
$$x < 0, \quad x \in \mathbb{R}; \quad x\tilde{a} = (xa_1, xa_2, -x\alpha, -x\beta)$$
$$\tilde{a} + \tilde{b} = (a_1 + b_1, a_2 + b_2, \alpha + \alpha', \beta + \beta')$$

One convenient approach for solving fuzzy linear programming problems is based on the concept of comparison of fuzzy numbers by use of ranking functions (see [6]). An effective approach for ordering the elements of $F(\mathbb{R})$ is to define a ranking function $R: F(\mathbb{R}) \to \mathbb{R}$ which maps each fuzzy number into the real line, where a natural order exists. We define orders on $F(\mathbb{R})$ by:

$$\tilde{a} \gtrsim \tilde{b} \quad \text{if and only if} \quad R(\tilde{a}) \geq R(\tilde{b})$$
$$\tilde{a} \succ \tilde{b} \quad \text{if and only if} \quad R(\tilde{a}) > R(\tilde{b})$$
$$\tilde{a} \simeq \tilde{b} \quad \text{if and only if} \quad R(\tilde{a}) = R(\tilde{b})$$

Where $\tilde{a}$ and $\tilde{b}$ in $F(\mathbb{R})$. Also we write $\tilde{a} \gtrsim \tilde{b}$ if and only if $\tilde{a} \gtrsim \tilde{b}$. We restrict our attention to linear ranking functions, that is, a ranking function R such that

$$R(k\tilde{a} + \tilde{b}) = kR(\tilde{a}) + R(\tilde{b}),$$

for any $\tilde{a}$ and $\tilde{b}$ belonging to F(ℝ) and any k $\in \mathbb{R}$.

Let $\tilde{a}$ be a fuzzy variable with membership function $\mu_{\tilde{a}}(x)$. Since $\mu_{\tilde{a}}(x)$ is the degree to which x is compatible with $\mu_{\tilde{a}}$, it is proportional to some probability density function $f(x)$. For example, $\mu_{\tilde{a}}(x)$ is proportional to the number of experts who believe in x is good, so that the more experts who believe in x, the greater the chance that is actually good [10].
Letting

$$f(x) = \frac{\mu_{\tilde{a}}(x)}{\int_X \mu_{\tilde{a}}(x)\ dx}$$

We can choose x^* to minimize the average deviation $\int (x - x^*)^2 f(x)\, dx$. Differentiation with respect to x^* yields

$$x^* = \frac{\int_X x\mu_{\tilde{a}}(x)\ dx}{\int_X \mu_{\tilde{a}}(x)\ dx}$$

Which is called centroid value.
So, Letting

$$R(\tilde{a}) = x^*$$

Which reduce to

$$R(\tilde{a}) = \frac{(a_2)^2 - (a_1)^2 + \alpha a_1 + \beta a_2 + \frac{\beta^2}{3} - \frac{\alpha^2}{3}}{\alpha + \beta + 2(a_2 - a_1)}$$

where $\tilde{a} = (a_1, a_2, \alpha, \beta)$.

B. the assignment problem FCAPP

Clearly, the assignment problem is equivalent to find a maximum matching of a complete bipartite graph, such that the weight of the matching is minimum, so we introduce some base concepts with respect to graph. Throughout this paper, all the graphs are directed and simple complete bipartite graph which is usually denoted as $G = (V_1, V_2)$, where the vertices set V of the related graph is defined as $V = V_1 \cup V_2$ and the edges set E is defined as $E = \{(u,v) | u \in V_1, v \in V_2\}$. In order to describe the problems conveniently, we denote $V = \{v_1, v_2, \dots, v_{|V|}\}$, $E = \{e_1, e_2, \dots, e_{|E|}\}$. Let $n = |V_1|$ be the cardinality of vertices set and $m = |V_2|$ that of edges set. Sometimes, we denote $E(G)$ and $V(G)$ the edges set and the vertices set of graph G for convenient, respectively.

Assume that the set $V_1 = \{1, 2, \dots, n\}$ is related to n jobs and $V_2 = \{1, 2, \dots, m\}$ to m workers in a complete bipartite weighted graph. The FCAPP degenerate to the FCAP when m = n, so we assume that $n < m$ in this paper (for the case of $n > m$, we just need to modify the mutation operator of genetic algorithm). Let W_s be a subset of V_2 is related to the *sth* company's workers,. Let $m_s = |W_s|$ for $s = 1,2, \dots, k$. So $\sum_{s=1}^{k} m_s = m$, $\bigcup_{s=1}^{k} W_s = V_2$, means m workers are members k various company. Furthermore, s-company can assign extreme number b_s of workers for assignment problem (FCAPP). ($b_s \le m_s \; \forall s = 1, .. k$, $\sum_{s=1}^{k} b_s \ge n$.

For convenience, assume that the sets $W_1 = \{1, 2, \dots, sum_1\}, W_2 = \{sum_1 + 1, \dots, sum_2\}, \dots, W_k = \{sum_{k-1} + 1, \dots, sum_k\}$ are related to company's workers *first*, *second*,…, *sth*, respectively,($sum_s = \sum_{i=i}^{s} m_i$) .

Let $\tilde{c}_{ij}$, be the direct cost related to the assigning the job j to the worker i,(of the edge (i,j)) for $i \in V_1, j \in V_2$ and $\tilde{d}_s$, be the company cost related to each worker is generated from the *sth* company $s = 1,2, \dots, k$ and $\tilde{g}_i, i = 1,2, \dots, m$, the penalties of vertex i.

Define decision variable x_{ij}.

$$x_{ij} = \begin{cases} 1 & \text{if job } j \text{ is assigned to worker } i \\ 0 & o.w \end{cases}$$

For convenience, any assignment can also be denoted by such a vector x and, $\tilde{c}, \tilde{d}$, and $\tilde{g}$ are the vectors consist of the costs $\tilde{c}_{ij}$, $\tilde{d}_s$ and $\tilde{g}_i$, respectively. Therefore, the cost function of assignment x can be introduce as

$$z(x, \tilde{c}, \tilde{d}, \tilde{g}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \tilde{c}_{ij} x_{ij} + \sum_{s=1}^{k} \tilde{d}_s \sum_{i \in w_s}\sum_{j=1}^{n} x_{ij} + \sum_{i=1}^{m} \tilde{g}_i \left(1 - \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\right)$$

Therefore, the proposed assignment problem has the following form

Stage 1:

$$\begin{cases} \min \quad \sum_{i=1}^{m}\sum_{j=1}^{n}\tilde{c}_{ij}x_{ij} + \sum_{s=1}^{k}\tilde{d}_s \sum_{i\in w_s}\sum_{j=1}^{n}x_{ij} + \sum_{i=1}^{m}\tilde{g}_i\left(1 - \sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}\right) & (1) \\ \qquad\qquad \sum_{i=1}^{m}x_{ij} = 1 \quad , j = 1,2,..,n & (2) \\ s.t \qquad\qquad \sum_{j=1}^{n}x_{ij} \le 1 \quad , i = 1,2,..,m & (3) \\ \qquad\qquad \sum_{i\in w_s}\sum_{j=1}^{n}x_{ij} \le b_s \quad , s = 1,2,\dots,k & (4) \\ \qquad\qquad x_{ij} = 0 \; or \; 1 \quad i = 1,\dots,m, \;\; j = 1,\dots,n & (5) \end{cases}$$

From now on, we assume that all the costs $\tilde{c}_{ij}$, $\tilde{d}_s$ and $\tilde{g}_i$ are fuzzy variables. the objective is found in Eq. (1). The objective is to minimize the cost assignment schedule where *n* jobs are allocated to *m* workers. Constraints (2) ensure that each job must be assigned just one worker. Constraints (3) restrict the number of workers assignment problem for each job. Constraints (4) restrict the number of workers assignment problem in each company. Moreover, Constraint (5) set up the binary restrictions for x_{ij}.

It is clear that the value objective function is also a fuzzy variable when the vectors $\tilde{c}, \tilde{d}$ and $\tilde{g}$ are fuzzy vectors. In order to rank $z(x, \tilde{c}, \tilde{d}, \tilde{g})$, different ideas are employed in different situations. If we wants to obtain a assignment with minimum value of cost function $z(x, \tilde{c}, \tilde{d}, \tilde{g})$, then the following concept is considered.

Optimal solution: A assignment x^* is called the optimal solution (optimal assignment) problem Stage 1, if

$$R\left(z(x^*, \tilde{c}, \tilde{d}, \tilde{g})\right) \le R\left(z(x, \tilde{c}, \tilde{d}, \tilde{g})\right)$$

for any assignment x.

III. THE MOD ELS OF FCAPP

In this section, based on the knowledge introduced in Section 2, we will introduce the modeling method of the FCAPP.

Essential idea is to optimize the value of objective function Stage 2 subject to some constraints. Therefore, the FCAPP can be formulated as following model:

Stage 2:

$$\begin{cases} \min \quad R\left(\sum_{i=1}^{m}\sum_{j=1}^{n}\tilde{c}_{ij}x_{ij} + \sum_{s=1}^{k}\tilde{d}_s\sum_{i\in w_s}\sum_{j=1}^{n}x_{ij} + \sum_{i=1}^{m}\tilde{g}_i\left(1-\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}\right)\right) & (6) \\ \sum_{i=1}^{m}x_{ij} = 1 \quad , j = 1,2,..,n & (7) \\ s.t \quad \sum_{j=1}^{n}x_{ij} \le 1 \quad , i = 1,2,..,m & (8) \\ \sum_{i\in w_s}\sum_{j=1}^{n}x_{ij} \le b_s \quad , s = 1,2,\dots,k & (9) \\ x_{ij} = 0 \; or \; 1 \quad i = 1,\dots,m, \;\; j = 1,\dots,n & (10) \end{cases}$$

Eq. (6) describes the fuzzy ranking objective function of the addressed problem which is minimization of the cost assignment schedule the descriptions of constraints (7)–(10) are the same as constraints (2)–(5).

A. The crisp equivalent

If all the costs are characterized by triangular fuzzy variables or trapezoidal fuzzy variables, the Stage 2 can be converted to their crisp equivalents. We just take the trapezoidal fuzzy variables as the example to illustrate this idea.

Lemma 1: Let $\tilde{a}$ and $\tilde{b}$ be two independent fuzzy variable. Then

$$R[\tilde{a} + \tilde{b}] = R[\tilde{a}] + R[\tilde{b}] .$$

Theorem 1: If all $\tilde{c}_{ij}$, $\tilde{d}_s$ and $\tilde{g}_i$, are independent fuzzy variables, then Stage 2, is equivalent to the following model,

Stage 3:

$$\begin{cases} \min \sum_{i=1}^{m}\sum_{j=1}^{n}R(\tilde{c}_{ij})x_{ij} + \sum_{s=1}^{k}R(\tilde{d}_s)\sum_{i\in w_s}\sum_{j=1}^{n}x_{ij} + \sum_{i=1}^{m}R(\tilde{g}_i)\left(1-\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}\right) & (11) \\ \sum_{i=1}^{m}x_{ij} = 1 \quad , j = 1,2,..,n & (12) \\ s.t \quad \sum_{j=1}^{n}x_{ij} \le 1 \quad , i = 1,2,..,m & (13) \\ \sum_{i\in w_s}\sum_{j=1}^{n}x_{ij} \le b_s \quad , s = 1,2,\dots,k & (14) \\ x_{ij} = 0 \; or \; 1 \quad i = 1,\dots,m, \;\; j = 1,\dots,n & (15) \end{cases}$$

Proof. The proof is followed from Lemma 1, immediately.

So, we have

Stage 4:

$$\begin{cases} \min & z(x) \\ & \sum_{i=1}^{m} x_{ij} = 1 \quad , j = 1,2,..,n \\ s.t & \sum_{j=1}^{n} x_{ij} \le 1 \quad , i = 1,2,..,m \\ & \sum_{i \in w_s} \sum_{j=1}^{n} x_{ij} \le b_s \quad , s = 1,2,\dots,k \\ & x_{ij} = 0 \; or \; 1 \end{cases}$$

Where

$$z(x) = \sum_{i=1}^{m}\sum_{j=1}^{n} \alpha_{ij} x_{ij} + \sum_{s=1}^{k} \beta_s \sum_{i \in w_s}\sum_{j=1}^{n} x_{ij} + \sum_{i=1}^{m} \gamma_i \left(1 - \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\right)$$

$$R(\tilde{c}_{ij}) = \alpha_{ij} = \frac{(c_{ij}^2)^2 - (c_{ij}^1)^2 + c_{ij}^1 c_{ij}^3 + c_{ij}^2 c_{ij}^4 + \frac{(c_{ij}^4)^2}{3} - \frac{(c_{ij}^3)^2}{3}}{c_{ij}^3 + c_{ij}^4 + 2(c_{ij}^2 - c_{ij}^1)}$$

,

$$R(\tilde{d}_s) = \beta_s = \frac{(d_s^2)^2 - (d_s^1)^2 + d_s^1 d_s^3 + d_s^2 d_s^4 + \frac{(d_s^4)^2}{3} - \frac{(d_s^3)^2}{3}}{d_s^3 + d_s^4 + 2(d_s^2 - d_s^1)}$$

and

$$R(\tilde{g}_i) = \gamma_i = \frac{(g_i^2)^2 - (g_i^1)^2 + g_i^1 g_i^3 + g_i^2 g_i^4 + \frac{(g_i^4)^2}{3} - \frac{(g_i^3)^2}{3}}{g_i^3 + g_i^4 + 2(g_i^2 - g_i^1)}$$

Proof. Due to the independent property of weights $\tilde{c}_{ij}$, $\tilde{d}_s$ and $\tilde{}$ and $x_{ij} \geq 0$, it is clear.

For general, the objective functions in Eqs. Stage 2 and 3, usually have many variable and the proposed models need a suitable method. In order to solve the model, we design a heuristic genetic algorithm in next section.

IV. HEURISTIC GENETIC ALGORITHM

There are many algorithms for the assignment problem and its extending problems. The greedy genetic algorithm [1], the Branch-and-bound algorithm [13], genetic algorithm [12], the parallel depth first search branch and bound algorithm [7], the lagrangian relaxation algorithm [8], the extended concentric tabu search method [3] and the improved hybrid genetic algorithm [9] are proposed for solving the assignment problem. Furthermore, the labeling algorithm [5] is proposed for the fuzzy assignment problem and the randomized parallel algorithms [11] is given for solving the multidimensional assignment problem.
In this section, a heuristic genetic algorithm is considered for solving the fuzzy multi-company workers assignment problem with penalty.

A. Representation

One important Stage for the genetic algorithm is representation. There are many ways to represent a solution of optimization problem. In this research, a chromosome is a set of integers value and the length of the chromosome can be exactly defined as the number $m = |V_2|$ which denotes the number of the workers. A chromosome is represented as an array $P = \{p_1, p_2, \dots, p_m\}$, where the value of is equal to the index of the job to which the worker is receive. We take *N* to denote the population size. So the number of chromosomes is equal *N*.
So, we have

$$p_i = \begin{cases} j \in V_1 & \text{if the job } j \text{ assigned to worker } i \\ 0 & \text{if the worker } i \text{ do not receive any job} \end{cases}$$

B. Initialization process

The initialization process of this problem can be described as follows: For $i = 1$ to m, randomly select worker w_j from interval $[1, m]$. Assume that $w_j \in W_s$, if $w_j \neq w_1, \dots, w_{j-1}$ and the number of *sth* company's workers received the jobs $1,2, \dots, j-1$ are lesser than , in the chromosome P , then job j assign worker w_j, for $j = 1, \dots, n$,otherwise select another worker, until all jobs are assigned. We initialize chromosomes $P_1, P_2, \dots, P_N$ by repeating following algorithm *N* times.

In Step 4, For convenience, assume that $N_s^j[P]$ denotes the number of *sth* company's workers received the jobs $1,2, \dots, j$ in the chromosome , (for control *3th* constrain). After finish Initialization process, $N_s[P]$, denotes the number of *sth* company's workers received the jobs $1,2, \dots, n$,in the chromosome .

Step 1. For $= 1$ to *N*, repeat Step 2 to Step 7, *N* times.
Step 2. Let $P_i[i'] = 0,\ i' = 1, \dots, m$.
Step 3. For $j = 1$ to n, repeat Step 4 to Step 7, n times.
Step 4. Randomly generate a positive integer w_j from the interval $[1, m]$.
Step 5. For $s = 1$ to k, repeat Step 6 , k times.
Step 6. If $w_j \in s$ then let g =s.
Step 7. If $w_j \neq w_1, \dots, w_{j-1}$ and $N_g^{j-1}[P_i] < b_g$then assign job j to worker w_j : $P_i[w_j] = j$. Otherwise go to Step 4.

Obviously, all the chromosomes generated by above algorithm are feasible.

C. Crossover operation

Let $P_{cross} \in (0,1)$ be the crossover probability. In order to determine the parents for crossover operation, we repeat the following process from $i = 1$ to *N*: randomly generating a real number r from the interval $(0,1)$, the chromosome P_i is selected as a parent if $r < P_{cross}$. Let chromosomes (P_1', P_2') is selected from the chromosomes $P_1, P_2, \dots, P_N$ for the crossover process.

For $i = 1$ to m, randomly select worker w_i from interval $[1, m]$. Let w_i is even and assume that $w_i \in W_s$, if $w_i \neq w_1, \dots, w_{i-1}$ and number of *sth* company's workers received the jobs $j_1, j_2, \dots, j_{i-1}$ are lesser than b_s, in the chromosome P_1'', then $P_1''[w_i] = P_1'[i] = j_i$, otherwise let $i = i + 1$ until worker w_i received the job. If there isn't any job in chromosome P_1'' then used of the chromosome P_2''. If the worker $w_m$do not receive any job then select job j from between jobs don't assigned, Furthermore, consider all constrain problem then let $P_1''[w_m] = j$, But, if w_i is odd, similarly repeat the upper method, $P_1''[w_i] = P_2'[i] = j_i$.

For example, let $m = 15, n = 10$ and we have five company, $W_1, \dots, W_5$, $|W_1| = 5$, $|W_2| = 3$, $|W_3| = 2$, $|W_4| = 1$, $|W_5| = 4$ and $b_1 = 4$, $b_2 = 2$, $b_3 = 2$, $b_4 = 1$, $b_5 = 3$.
In Fig. 1, for $i = 1$ the worker 9 is selected and the number 9 is odd, so we have:

If $i = 1$ then $w_1 = 9 \Longrightarrow P_1''[9] = P_1'[1] = 1$
If $i = 2$ then $w_2 = 2 \Longrightarrow P_1''[2] = P_2'[1] = 8$

$$P_1' = \{1,6,4,0,5,9,0,0,8,2,7,3,0,0,10\}$$
$$P_2' = \{8,3,0,0,4,0,9,7,10,0,5,1,2,0,6\}$$

Crossover

$$P_1'' = \{2,8,0,9,5,0,7,10,1,3,0,0,4,0,6\}$$
$$P_2'' = \{7,0,8,9,3,6,0,5,10,1,0,4,0,0,2\}$$

Fig. 1. The crossover operation.

This operation can be summarized as the following algorithm:

➢ ***Crossover algorithm***

Step 1. Let $J_1 = 1$ and $J_1 = 1, h = 1$.
Step 2. Let $h = 1$, repeat step 3 to step 11 until $h = m$
Step 3. Randomly select worker w_i from the interval $[1, m]$. If $w_i \neq w_1, \dots, w_{i-1}$ go to Step 6, otherwise go to Step 3.
Step 4. If $J_1 > m$ and $J_2 > m$ then go to 11
Step 5. If $J_1 > m$ then go to 7. If $J_2 > m$ then go to 6.
Step 6. If w_i is odd then, Let $P_1'[J_1] = j_i$ and let $J_1 = J_1 + 1$. Otherwise go to 7.
Step 7.. Let $P_2'[J_2] = j_i$ and let $J_2 = J_2 + 1$.
Step 8. For $s =$ 1 to k, repeat Step 8 , k times.
Step 9. If $w_i \in s$ then let g=s.
Step 10. If $j_i \neq j_1, \dots, j_{i-1}$ and $N_g^{j_{i-1}}[P_1''] < b_g$then $P_1''[w_i] = j_i$ and let $h = h + 1$ and go to Step 2. otherwise go to Step 4.
Step11. Select j and let $j_i = j$ go to Step 10.

D. Mutation operation

Let $P_{mut} \in (0,1)$ be the mutation probability. We use the following operator to select the chromosome to be mutated: for $i = 1$ to N, randomly generate a real number r from interval $(0,1)$; if $r \leq P_{mut}$, then the chromosome P_i is selected to be mutated.

Let P is selected from the chromosomes $P_1, P_2, \dots, P_N$ for the mutation process. The base The base ideas of the crossover are illustrated by the Fig. 2. The base operations are described as follows: Randomly select two workers w_1, w_2, assume that the workers w_1, w_2 are received to the jobs j_1, j_2 in the chromosome P, so $P[w_1] = j_1, P[w_2] = j_2$, then exchange the jobs j_1 and j_2. If $P[w_1] \neq P[w_2] \neq 0$, then exchange the jobs j_1 and j_2 , if $X[w_1] = P[w_2] = 0$, then Randomly select another two workers. But if $P[w_1] = 0$ or $P[w_2] = 0$, for example $P[w_1] = 0$, $w_1 \in W_s$, if $N_s[P] < b_s$ then exchange the jobs j_1 and j_2, otherwise Randomly select

another two workers, until that exchange the jobs $j_1$and j_2. In Fig. 2, the workers 2 and 7 are selected. Following is the mutation operator.

$$P = \{2,3,0,0,6,0,9,7,10,0,5,1,8,0,4\}$$

mutation

$$P = \{2,9,0,0,6,0,3,7,10,0,5,1,8,0,4\}$$

Fig. 2. The mutation operation.

➢ ***Mutation algorithm:***

Step 1. For $i = 1$ to *N*, repeat Steps 2–8 *N* times.
Step 2. randomly generate a real number r from interval $(0,1)$; if $r \leq P_m$, then go to Step 3. Otherwise, go to Step 1.
Step 3. Randomly select two workers w_1, w_2 from the interval $[1, m]$. If $P[w_1] = j_1 = 0$ and $P[w_2] = j_2 = 0$ then go to Step 3, If $P[w_1] \neq P[w_2] \neq 0$ then go to Step 8. Otherwise go to Step 4.
Step 4. If $P[w_1] = 0$ then let $w = w_1$,otherwise let $w = w_2$ and go to Step 5.
Step 5. For $s = 1$ to k, repeat Step 6 , k times.
Step 6. If $w \in s$ then let g=s.
Step 7. If $N_g[P_i] < b_g$then go to Step 8, otherwise go to Step 3.
Step 8. Exchange the jobs j_1 and j_2 which the job Jp is assigned in the chromosome *P*, respectively, by the operation $j = P[w_1],\ P[w_1] = P[w_2], P[w_2] = j$.

E. Selection process

For selection process, we determine the fitness function $z_i^{'}$ to evaluate the *ith* chromosome $i = 1,2,\ldots,N$. Suppose that z_i denote the value of the objective function in the Stage 2. Therefore we have:

$$E(P_i) = \frac{z_i^{'}}{\sum_{k=1}^{N} z_k^{'}} \times 100 \qquad ,p_i = \sum_{k=1}^{i} E(P_i)$$

Where

$$f_i^{'} = \frac{\sum_{i=1}^{N} R\left(z_i(x, \tilde{c}, \tilde{d}, \tilde{g})\right)}{R\left(z_i(x, \tilde{c}, \tilde{d}, \tilde{g})\right)},$$

$$z_i(x, \tilde{c}, \tilde{d}, \tilde{g}) = \sum_{i=1}^{m}\sum_{j=1}^{n} \tilde{c}_{ij}x_{ij} + \sum_{s=1}^{k} \tilde{d}_s \sum_{i \in w_s}\sum_{j=1}^{n} x_{ij} + \sum_{i=1}^{m} \tilde{g}_i \left(1 - \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}\right)$$

Then we use the spanning roulette wheel to prefer the chromosomes: randomly generate a number $p \in (0, 100)$; if $p \in [p_{i-1}, p_i)$, then the chromosome P_i is selected. Following is the algorithm.
This process can be described as the following algorithm:

➢ ***Selection algorithm:***

Step 1. Let $k = 1$, repeat Step 2 until $k = N$

Step 2. Randomly generate a number $p \in (0, 100)$; if $p \in [p_{i-1}, p_i)$, then chromosome P_i is selected and let $k = k + 1$.

- ***Genetic algorithm:***

Step 1. Randomly initialize *N* chromosomes.

Step 1. Let $k = 1$, repeat Step 2 to Step 7 until $k = T$ (until a given number times (T)).

Step 2. Calculate the fitness of each chromosome according to the objective values.

Step 3. Select the chromosomes by spanning the roulette wheel.

Step 4. Perform crossover process and mutation process on the chromosomes.

Step 5. If $k = T$ report the best chromosome as the optimal solution.

Step 6. Arrange the chromosomes in decreasing order of processing times to form a sequencing priority list.

Step 7. Select 50% from the best chromosome and another 50% Randomly select from the remain chromosomes and let $k = k + 1$.

V. THE NUMERICAL EXAMPL

In this section, the efficiency of the proposed heuristic algorithm is showed by solving an example. In the example, let $n = 10$, $m = 15$ and the problem's data, costs are given in the Table 1, Table 2 and Table 2.

Obviously, the difference between the Stage 2, 3 and 4 is just the objective function. Therefore, we just take the stage 2, as an example to solve the numerical example.

i / j	1	2	3	4	5	6	7	8	9	10
1	10,15,11,3	20,22,6,14	30,60,1,2	40,54,6,3	15,18,12,5	20,38,2,6	40,44,9,3	5,5,10,9	10,11,8,4	12,15,10,7
2	50,60,5,2	55,59,16,1	50,61,2,9	35,39,9,9	47,55,6,6	1,14,7,4	34,39,9,6	10,18,8,8	17,24,7,9	11,14,9,8
3	30,38,3,2	21,30,9,9	12,18,5,3	21,30,1,4	30,44,8,5	45,48,2,2	55,59,3,3	12,15,4,4	26,34,5,9	13,38,2,2
4	40,44,2,4	61,68,1,2	32,55,10,3	33,25,3,5	23,39,9,2	12,15,11,5	11,18,4,3	23,31,3,6	22,24,9,5	23,38,7,6
5	22,40,8,9	34,39,8,3	56,61,3,1	28,28,6,10	11,22,5,5	31,38,2,12	22,25,8,3	23,28,8,3	47,55,3,8	12,30,9,6
6	33,44,10,4	45,58,10,2	34,34,6,11	18,28,11,3	44,59,2,12	51,65,4,4	32,48,3,3	90,93,1,15	19,31,9,8	41,55,8,7
7	24,28,10,9	42,50,3,3	21,43,2,1	4,22,12,2	50,58,4,2	52,52,10,8	40,46,7,5	15,20,5,4	32,38,8,3	32,34,6,6
8	94,100,8,11	10,15,2,8	12,14,10,10	2,18,15,6	19,31,2,4	64,66,4,5	10,23,4,6	44,51,13,3	42,43,6,8	31,32,3,4
9	34,40,8,2	5,14,10,4	42,48,2,4	14,35,3,6	2,31,7,5	12,18,7,10	10,20,5,10	15,18,4,2	17,20,1,10	1,28,4,2

10	10,50,13,3	12,43,6,5	22,50,3,6	55,55,14,5	50,52,10,4	35,46,2,4	20,47,3,7	21,21,10,8	34,34,7,3	10,20,5,4
11	15,17,3,5	11,14,4,6	18,30,2,3	21,31,3,8	12,70,5,4	23,23,6,2	30,51,3,4	20,43,4,4	1,10,8,2	20,21,7,3
12	1,1,13,4	30,40,3,7	2,15,6,8	29,30,5,8	22,40,8,6	2,22,3,7	12,32,4,5	2,12,3,2	40,50,7,3	31,55,7,7
13	42,50,5,8	20,35,3,4	31,47,4,2	12,45,2,2	21,34,8,12	3,10,2,4	5,17,8,8	4,34,3,7	23,30,8,5	12,21,10,8
14	1,12,6,6	5,5,12,9	21,30,7,2	32,455,10	44,55,5,8	5,12,3,8	10,28,4,4	12,31,2,4	50,54,3,10	55,61,12,6
15	50,72,1,3	35,39,2,8	47,72,12,5	21,29,3,3	12,24,14,2	14,30,5,5	17,29,2,3	1,9,1,4	30,38,2,5	21,21,10,15

Table 1. The *direct costs* of the numerical example, $\tilde{c}_{ij}$

S	m_S	b_S	$\tilde{d}_s$
1	5	4	4, 6, 1, 2
2	3	2	3, 7, 2, 3
3	2	2	5, 8, 4, 2
4	1	1	6,7, 4, 2
5	4	3	5, 6, 2, 3

Table 2. The *company costs*, $\tilde{d}_s$, m_s, b_s

i	$\tilde{g}_i$
1	4, 9, 5, 2
2	9, 12, 3, 3
3	2, 5, 2, 3
4	5, 7, 1, 1
5	5, 8, 2, 2
6	4, 6, 3, 3
7	7, 9, 1, 3
8	2, 9, 4, 2
9	5, 8, 2, 3
10	8, 10, 2, 4
11	5, 6, 2, 3
12	5,7, 3, 2
13	6, 8, 4, 1
14	4, 9, 5, 2
15	10, 12, 1, 3

Table 3. The *penalty* of the numerical example, $\tilde{g}_i$

.

(worker, job)									
(12,1)	**(9,2)**	**(8,3)**	**(7,4)**	**(15,5)**	**(2,6)**	**(13,7)**	**(1,8)**	**(11,9)**	**(10,10)**

Table 4.The optimal assignment

Let the crossover probability is $p_{cross} = 0.8$ and the mutation probability is $p_{mut} = 0.5$. All the evolution parameters are obtained by the statistic and analyze of the experiment results of a numerical example with 15 workers and 10 jobs.

The optimal value of the objective function is equal to 167.82. It is well known that the efficiency of the genetic algorithm can be mainly characterized by the evolution process and the absolute errors or the relative errors. For the given example, the first we consider 200 generations, with the given evolution parameters $p_{cross} = 0.8$ and $p_{mut} = 0.5$ then the optimal solution obtain at the 600th generation. If we consider 500 generations then the optimal solution obtain the 400th generation.

VI. CONCLUSIONS

In this paper, one important assignment problem is studied. There have been many algorithms for the assignment problem and its extending problems [1, 8, 10, 11, 12]. For solving the given problem FCAPP, we introduced a heuristic genetic algorithm. And using this algorithm the optimal solution (optimal assignment) the proposed problem is obtained.

By considering the number of jobs, workers and type assignment each job to each worker, this problem can be extended to the assignment problem which for solving it should be used a different genetic algorithm or another algorithm.

REFERENCES

[1] R.K. Ahuja, J.B. Orlin, A. Tiwari, A greedy genetic algorithm for the quadratic assignment problem, Compute. Oper. Res. 27 (2000) 917–934.
[2] Bogomolnaia, A new solution to the random assignment problem, Journal of Economic, Theory 100 (2001) 295–328.
[3] Z. Drezner, The extended concentric tabu for the quadratic assignment problem, Eur. J.Oper . Res. 160 (2005) 416–422.
[4] J.D. Lamb, A note on the weighted matching with penalty problem, Recognition Letters 19 (1998) 261–263.
[5] C.J. Lin, U.P. Wen, A labelling algorithm for the fuzzy assignment problem, Fuzzy Sets Syst. 142 (2004) 373–391.
[6] N. Mahdavi-Amiri, S.H. Nasseri, Duality results and a dual simplex method for linear programming problems with trapezoidal fuzzy variables, Fuzzy Sets and Systems 158 (2007) 1961–1978.
[7] B. Mans, T. Mautor, C. Roucairol, A parallel depth first search branch and bound algorithm for the quadratic assignment problem, Eur. J. Oper. Res. 81(1995) 617–628.
[8] L.Z. Milis, V.F. Magirou, A lagrangian relaxation algorithm for sparse quadratic assignment problems, Oper. Res. Lett. 17 (1995) 69–76.
[9] Misevicius, An improved hybrid genetic algorithm: new results for the quadratic assignment problem, Knowledge-Based Syst. 17 (2004) 65–73.
[10] H.T. Nguyen, E.A. Walker, Department of Mathematical Sciences New Mexico State University Las Cruces, New Mexico, 2006.
[11] C.A.S. Oliveira, P.M. Pardalos, Randomized parallel algorithms for the multidimensional assignment problem, Appl. Numer: Math. 49 (2004) 117–133.
[12] L. Liu, The fuzzy quadratic assignment problem with penalty: new models and genetic algorithm, Appl. Math. Compute. 174 (2006) 1229–1244.
[13] P. Hahn, Thomas Grant, Nat Hall, A branch-and-bound algorithm for the quadratic assignment problem based on the Hungarian method, Eur. J. Oper. Res. 81 (1998) 629–640.
[14] N. Shasavari Pour, M. Modarres, R. Tavakkoli-Moghaddam, E. Najafi, Optimizing A Multi-Objective Time-Cost-Quality Trade-Off Problem by a new Hybrid Genetic Algorithm, Word Applied Sciences Journal 10(3): 335-363, 2010.
[15] L.A. Zadeh, Fuzzy sets, Information and Control 8 (1965) 338–353.