http://dx.doi.org/10.7166/26-3-802

MODEL AND ALGORITHMS OF THE FUZZY THREE-DIMENSIONAL AXIAL ASSIGNMENT PROBLEM WITH AN ADDITIONAL CONSTRAINT

C.-J. Lin[1*] & K.-T. Ma[2]

[1]Department of Industrial Engineering and Management
Ta Hwa University of Science and Technology, Taiwan
lcj@tust.edu.tw

[2]Department of Industrial Engineering and Engineering Management
National Tsing Hua University, Taiwan
d9734802@oz.nthu.edu.tw

ABSTRACT

This study constructs a practical fuzzy three-dimensional axial assignment model, and proposes two efficient algorithms to solve the model. In our case, the model is applied to team performance management in a company to promote the performance of all members in a team. Two algorithms, namely the index-based branch and bound (B&B) algorithm and the *f-g* trade-off algorithm, which is a hybrid of the trade-off and B&B concepts, are proposed. A numerical example is presented to illustrate these two algorithms. The computational results show that the proposed algorithms are sufficiently efficient and accurate. Two special cases are also discussed.

OPSOMMING

‘n Praktiese, wasige, driedimensionele aksialetoekenningsmodel word voorgehou en twee doeltreffende algoritmes om die model op te los word voorgestel. Die model word toegepas op die spanprestasiebestuur in ‘n maatskappy met die doel om die spanlede se vertoning te verbeter. Twee algoritmes, naamlik die indeksgebaseerde vertak-en-begrens algoritme en die *f-g* kompromie algoritme, word voorgestel. ‘n Numeriese voorbeeld word ter illustrasie van die twee algoritmes voorgehou. Die resultate toon dat die voorgestelde algoritmes doeltreffend en akkuraat is. Twee spesiale gevalle word ook bespreek.

[*] Corresponding author

1 INTRODUCTION

The three-dimensional (3D) axial assignment problem is an extension of the linear assignment problem – i.e., the 2D assignment problem. The 3D axial assignment problem is actually NP-hard [1]. It is usually stated as follows: assign *n* workers to *n* jobs on *n* machines to minimise the total cost [2]. There should be one-to-one correspondences between workers and machines, workers and jobs, and machines and jobs. These correspondences make up a cubic assignment. The model can be formulated as the following 0-1 integer programming problem [1-5]:

$$
\begin{aligned}
\text{Min } & \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}\cdot x_{ijk} \qquad (c_{ijk} \geq 0) \\
\text{s.t. } & \sum_{j=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \quad \text{for} \quad i = 1, 2, \ldots, n \\
& \sum_{i=1}^{n}\sum_{k=1}^{n} x_{ijk} = 1 \quad \text{for} \quad j = 1, 2, \ldots, n \qquad (1.1) \\
& \sum_{i=1}^{n}\sum_{j=1}^{n} x_{ijk} = 1 \quad \text{for} \quad k = 1, 2, \ldots, n \\
& x_{ijk} \in \{0, 1\} \text{ for } i, j, k = 1, 2, \ldots, n
\end{aligned}
$$

where c_{ijk} is the cost of assigning worker *i* to job *j* on machine *k,* and all c_{ijk}s construct a cubic matrix. If x_{ijk} = 1, then worker *i* is assigned to do job *j* on machine *k.* If x_{ijk} = 0, then worker *i* is either not assigned to job *j* or not assigned to machine *k.* In a deterministic environment, the 3D axial assignment problem is important in operations research. Applications of the 3D axial assignment problem arise in many areas, such as scheduling capital investments, addressing a rolling mill, military troop assignment, and satellite coverage optimisation [1,6]. Algorithms for the 3D axial assignment have been developed. However, the coefficient matrix of Equation 1.1 is not totally unimodular to ensure an integer solution [7]; thus it cannot be treated as efficiently at continuous problem-solving for its optimum as in a 2D assignment problem. Hence, algorithms to search for the exact solution of a 3D axial are mostly implicit enumeration methods [8], such as the branch and bound (B&B) method [3,9]. Some papers have developed good bounds for the B&B method, especially the lower bound for minimising the problem [6,8,10]. Hung and Lim [11] proposed a local search genetic algorithm-based method to solve the 3D axial assignment problem, but they could not guarantee that an optimal solution could be obtained.

Although all c_{ijk}s are deterministic numbers in Equation 1.1, the cost of many real-world applications may not be deterministic numbers. Many researchers have begun to study the 2D assignment problem in a fuzzy environment. These situations include various fuzzy assignment problem (FAP) models and algorithms [12-15], sensitivity analysis of FAP [16], quadratic FAPs [17], multi-criteria FAPs [18-20], traffic FAPs [21], two-objective FAPs [20,22], group decision-making [23], and multi-job FAPs [24]. Solution procedures of these 2D FAPs all take advantage of the totally unimodular property for efficiency. However, no paper discusses the modelling and methods for a fuzzy 3D axial assignment problem. Therefore, this paper presents a fuzzy 3D axial assignment model, and proposes two B&B-based algorithms to solve the model. The next section describes the related fuzzy theory and construction of the fuzzy 3D axial assignment model. Section 3 presents two algorithms for solving this model and a numerical example. Section 4 describes computational results, and Section 5 offers a conclusion to the paper.

2 MODEL CONSTRUCTION

2.1 Basic concept of related fuzzy theory

Fuzzy theory is useful for accessing many real situations of human life. In this section, we first recall several related definitions of fuzzy theory, which are used in this paper.

Definition 2.1.1. A crisp set *A* is a proper subset of a universe of discourse *X* such that $A \subseteq X$; the membership function of *A* can be defined as [25]:

$$u_A(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{if } x \notin A \end{cases} \quad \text{where } x \in X .$$

Definition 2.1.2. A fuzzy set $\tilde{A}$ is a subset of a universe of discourse *X*, which is characterised by a membership function $u_{\tilde{A}}(x),\ x \in X$, which maps *x* to a value within the range of [0,1] [26].

According to **Definitions 2.1.1.** and **2.1.2.**, *x* completely belongs to $\tilde{A}$ if the value of the membership function $u_{\tilde{A}}(x) = 1$; *x* does not belong to $\tilde{A}$ if $u_{\tilde{A}}(x) = 0$.

Definition 2.1.3. The *λ*-cut of $\tilde{A}$ of the universe of discourse X is defined as $\tilde{A}_\lambda = \{x \in X \,|\, u_{\tilde{A}}(x) \geq \lambda\}$ [26].

Therefore, the λ-cut is a subset of the universe of discourse X, where $u_{\tilde{A}}(x) \geq \lambda$.

Definition 2.1.4. The largest value of *λ* of the *λ*-cut of $\tilde{A}$, which is non-empty, is called the height of $\tilde{A}$, denoted as *h*($\tilde{A}$) [26].

Definition 2.1.5. A fuzzy set $\tilde{A}$ of the universe of discourse X is normal if *h*($\tilde{A}$)=1; otherwise, a fuzzy set $\tilde{A}$ of the universe of discourse X is subnormal [15,26,27].

Definition 2.1.6. A fuzzy set $\tilde{A}$ of a universe of discourse *X* is convex if and only if $u_{\tilde{A}}(\mu a + (1-\mu)b) \geq \min(u_{\tilde{A}}(a), u_{\tilde{A}}(b)),\ x \in [a,b], \text{and } \mu \in [0,1]$ [26].

Definition 2.1.7. A fuzzy set $\tilde{A}$ of a universe of discourse *X* is a fuzzy number or fuzzy interval if $\tilde{A}$ is both normal and convex [26].

Definition 2.1.8. A fuzzy set $\tilde{A}$ of a universe of discourse *X* is a subnormal fuzzy number or subnormal fuzzy interval if $\tilde{A}$ is both subnormal and convex [27].

Definition 2.1.9. A subnormal fuzzy interval [15,27] is a subnormal fuzzy subset of the universe of discourse X, which is both subnormal and convex.

Accordingly, a fuzzy number and fuzzy interval are a special case of subnormal fuzzy number and subnormal fuzzy interval, such that the maximal height of the membership function $u_{\tilde{A}}(x)$ is 1, respectively.

2.2 Model formulation

Suppose a project team consists of *n* workers and a manager. The workers and the manager are responsible for performing *n* jobs and for minimising the total costs, respectively. Each worker is assigned to one and only one job to perform it. In many cases, the job quality achieved depends on the assigned worker's skill, the machine used, and the input-cost of the job. Generally, a higher job quality indicates greater job cost. The job quality achieved is defined as the performance of the worker. Hence, workers desire high-input job costs and a suitable machine to perform the assigned job well. Thus the cost of worker *i* performing job *j* on machine *k*, which is denoted as $\tilde{c}_{ijk}$, is positively related to the performance of worker *i*.

Let q_{ijk} ($0 < q_{ijk} \leq 1$) be the highest possible quality of job *j* performed by worker *i* on machine *k*. Generally, q_{ijk} should be between 0.5 and 1.0. The membership function of $\tilde{c}_{ijk}$ is then defined as the linear monotone increasing function in Equation 2.1 and Figure 1. In the function, α_{ijk} is the minimal cost for worker *i* to perform job *j* on machine *k,* and β_{ijk} is the minimal cost required to reach the maximal job quality q_{ijk}. When the job input cost is between α_{ijk} and β_{ijk}, the job cost is linearly and positively related to the job quality achieved. However, any expense exceeding β_{ijk} is wasteful because quality can no longer be enhanced. Without loss of generality, it is assumed that $0 < \alpha_{ijk} < \beta_{ijk}$. Condition $x_{ijk} = 1$ is added to Equation 2.1 because there is no real expense if $x_{ijk} = 0$ in any feasible solution.

$$\mu_{ijk}(c_{ijk}) = \begin{cases} q_{ijk} & \text{if } c_{ijk} \geq \beta_{ijk},\ x_{ijk} = 1 \\ \dfrac{q_{ijk}(c_{ijk} - \alpha_{ijk})}{\beta_{ijk} - \alpha_{ijk}} & \text{if } \alpha_{ijk} \leq c_{ijk} \leq \beta_{ijk},\ x_{ijk} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

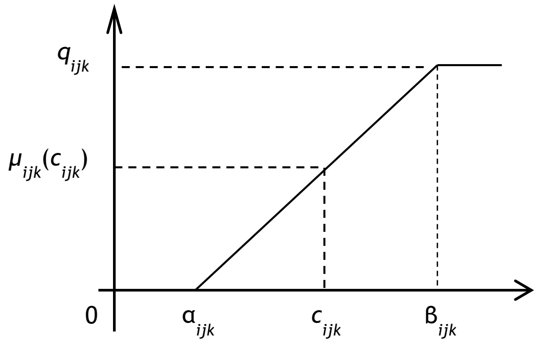


Figure 1: Membership function of $\tilde{c}_{ijk}$

Notation $<\alpha_{ijk}, \beta_{ijk}>$ is used to denote $\tilde{c}_{ijk}$. The cubic matrix $[\tilde{c}_{ijk}]$ is shown as

$$[\tilde{c}_{ijk}] = [<\alpha_{ijk}, \beta_{ijk}>].$$

Additionally, all α_{ijk}s form cubic matrix $[\alpha_{ijk}]$, all β_{ijk}s form cubic matrix $[\beta_{ijk}]$, and all q_{ijk}s form cubic matrix $[q_{ijk}]$. Define the cubic matrix $[\gamma_{ijk}]$ as

$$[\gamma_{ijk}] = [\frac{\beta_{ijk} - \alpha_{ijk}}{q_{ijk}}] \quad \text{for } i, j, k = 1, 2, \ldots, n. \tag{2.2}$$

where γ_{ijk} is the slope of the membership function.

The manager's duty is to keep the total cost ($\tilde{c}_T$) as low as possible. Numbers *a* and *b* are assumed to be constants chosen subjectively by the company. When the total cost is lower than *a*, the manager's performance is 1, when the total cost is greater than *b*, the manager's performance is 0, and when the total cost is between *a* and *b*, the total cost is linearly and negatively related to the manager's performance. Thus, the membership function of total cost $\tilde{c}_T$ is defined as the linear monotonically decreasing function in Equation 2.3), which is shown in Figure 2. Notation <*a*, *b*> denotes the fuzzy interval $\tilde{c}_T$. It is suggested that *a* should be a number less than or equal to the minimum of Equation 1.1, with α_{ijk} as its parameter, and *b* should be a number larger than or equal to the maximum of Equation 1.1, with β_{ijk} as its parameter.

$$\mu_T(c_T) = \mu_T(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}x_{ijk}) = \begin{cases} 1 & , \ c_T \le a \\ \dfrac{b - \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk} \cdot x_{ijk}}{b-a} = \dfrac{b-c_T}{b-a} & , \ a \le c_T \le b \\ 0 & , \ c_T \ge b \end{cases} \quad (2.3)$$

Because the performance of worker *i* (π_i) is the job quality achieved, then

$$\pi_i = \mu_{ijk}(c_{ijk}) \text{ for } x_{ijk} = 1 \text{ and } i = 1, 2, \ldots, n \quad (2.4)$$

The performance of the manager (π_m) is

$$\pi_m = \mu_T(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} c_{ijk}x_{ijk}) \quad (2.5)$$

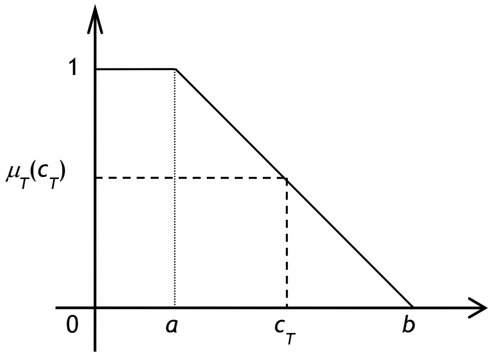


Figure 2: Membership function of $\tilde{c}_T$

Usually, the company takes the lowest performance of the team members, composed of all workers and the manager, as the team performance for promoting overall performance. Thus the performance of every team member must be equally emphasised. To maximise the team performance is then to maximise the minimal performance of all team members. Therefore, the study uses Bellman-Zadeh's criterion [28], which maximises the minimum of all membership functions corresponding to that solution. The objective function is then defined as

$$\text{Max-}\min_{\forall i} (\pi_i, \pi_m)$$

or

$$\text{Max-}\underset{x_{ijk}=1}{\text{Min}}\ (\mu_{ijk}(c_{ijk}),\ \mu_T(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}c_{ijk}x_{ijk})) \tag{2.6}$$

where x_{ijk} is an element of feasible solution **x** of Equation 1.1. The fuzzy 3D axial assignment problem model is represented as follows:

$$\begin{aligned}&\text{Max-}\underset{x_{ijk}=1}{\text{Min}}\ (\mu_{ijk}(\ c_{ijk}),\ \mu_T(\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}c_{ijk}x_{ijk}))\\ &\qquad\text{s.t. the constraints of (1.1)}\end{aligned} \tag{2.7}$$

Using the membership functions from Equation 2.1 and Equation 2.3, Equation 2.7 can be represented as the following equivalent model:

$$\begin{aligned}&\text{Max } \lambda\\ &\text{s.t. } \lambda\cdot x_{ijk} \le \frac{q_{ijk}(c_{ijk}^{\lambda}-\alpha_{ijk})}{\beta_{ijk}-\alpha_{ijk}}\cdot x_{ijk} \qquad \text{for } i,\ j,\ k=1,\ 2,\ldots,\ n\\ &\qquad \lambda \le \frac{b-\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}c_{ijk}^{\lambda}x_{ijk}}{b-a}\\ &\qquad 0\le \lambda x_{ijk} \le q_{ijk}x_{ijk} \quad \text{for } i,\ j,\ k=1,\ 2,\ldots,\ n\\ &\qquad c_{ijk}^{\lambda}\ge 0 \qquad \text{for } i,\ j,\ k=1,\ 2,\ldots,\ n\\ &\qquad x_{ijk}\in\{0,\ 1\} \quad \text{for } i,\ j,\ k=1,\ 2,\ldots,\ n\\ &\qquad \textit{and} \ \text{ the constraints of (1.1)}\end{aligned} \tag{2.8}$$

where c_{ijk}^{λ} is the λ-cut of $\tilde{c}_{ijk}$ and $\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}c_{ijk}^{\lambda}x_{ijk}$ is the corresponding total cost c_T^{λ}. In Equation 2.8, the x_{ijk}'s, c_{ijk}^{λ}'s, and λ are all decision variables. Hence, Equation 2.8 is a mixed nonlinear programming model.

In our case, Equation 2.8 is applied to team performance in human resource management that equally emphasises the performance of the manager and workers. However, Equation 2.8 can also be used for situations that equally emphasise the quality and quantity, time and cost, humanity and technology, and so on.

2.3 Model transformation

Suppose ω = {(*i*, *j*, *k*) | x_{ijk} = 1} is the corresponding cubic assignment of the feasible solution **x** of (1.1) and the discussion is confined based on ω. Then, Equation 2.8 can be simplified as

$$\begin{aligned}&\text{Max} \quad \lambda\\ &\text{s.t.} \quad \lambda \le \frac{q_{ijk}(c_{ijk}^{\lambda}-\alpha_{ijk})}{\beta_{ijk}-\alpha_{ijk}} \qquad ;\forall\ (i,j,k)\in\boldsymbol{\omega}\\ &\qquad 0\le\lambda\le\frac{b-\sum_{(i,j,k)\in\omega}c_{ijk}^{\lambda}}{b-a}\\ &\qquad \lambda\le q_{ijk} \qquad ;\forall\,(i,j,k)\in\boldsymbol{\omega}\\ &\qquad c_{ijk}^{\lambda}\ge 0 \qquad ;\forall\,(i,j,k)\in\boldsymbol{\omega}\end{aligned} \tag{2.9}$$

Theorem 2.1. Let λ_x be the optimal objective function value of (2.9). Then,

$$\lambda_x = \text{Max Min}\left(\frac{b - \sum\limits_{(i,j,k)\in\omega} \alpha_{ijk}}{(b - a + \sum\limits_{(i,j,k)\in\omega} \gamma_{ijk})} \;,\; \underset{(i,j,k)\in\omega}{\text{Min}} (q_{ijk}) \right) .$$

Proof. Rewriting Equation 2.9 into a linear programming model results in

$$\begin{aligned} &\text{Max } \lambda \\ &\text{s.t. } c_{ijk}^{\lambda} - \frac{(\beta_{ijk} - \alpha_{ijk})}{q_{ijk}} \cdot \lambda \geq \alpha_{ijk} \quad ; \;\forall\, (i,j,k) \in \omega \\ &\sum_{(i,j,k)\in\omega} c_{ijk}^{\lambda} + \quad (b-a)\cdot\lambda \leq b \\ &0 \leq \lambda \leq \underset{(i,j,k)\in\omega}{\min} (q_{ijk}) \\ &c_{ijk}^{\lambda} \geq 0 \;\; ; \;\; \forall\, (i,j,k) \in \omega \end{aligned} \tag{2.10}$$

The dual problem of Equation 2.10 is then

$$\begin{aligned} &\text{Min} \quad \sum_{(i,j,k)\in\omega} (\alpha_{ijk} h_i) + b h_{n+1} + \underset{(i,j,k)\in\omega}{\min} (q_{ijk}) \cdot y \\ &\text{s.t.} \quad h_i + h_{n+1} \quad \geq 0 \qquad \text{for } i=1, 2,\ldots, n \\ &- \sum_{(i,j,k)\in\omega} \frac{(\beta_{ijk} - \alpha_{ijk})}{q_{ijk}} w_i + (b-a) h_{n+1} \; + y \geq 1 \\ &h_i \leq 0 \;, h_{n+1} \geq 0 \;, y \geq 0 \qquad \text{for } i=1, 2,\ldots, n \end{aligned} \tag{2.11}$$

Obviously, Equation 2.10 produces $c_{ijk}^{\lambda} \geq \alpha_{ijk} + \frac{(\beta_{ijk} - \alpha_{ijk})}{q_{ijk}} \cdot \lambda > 0$ for all $(i, j, k) \in \omega$. Using the complementary slackness theorem yields the following:

$$h_i + h_{n+1} = 0 \qquad \text{for } i = 1, 2, \ldots, n$$

or

$$h_{n+1} = -h_1 = -h_2 = \cdots = -h_n \; . \tag{2.12}$$

Based on Equation 2.12 and Equation 2.2, Equation 2.11 can be simplified as follows:

$$\begin{aligned} &\text{Min} \quad (b - \sum_{(i,j,k)\in\omega} \alpha_{ijk}) h_{n+1} + \underset{(i,j,k)\in\omega}{\text{Min}} (q_{ijk}) \cdot y \\ &\text{s.t.} \quad (b - a + \sum_{(i,j,k)\in\omega} \gamma_{ijk}) h_{n+1} + \qquad y \geq 1 \\ &\qquad h_{n+1} \;, y \geq 0 \end{aligned} \tag{2.13}$$

The dual problem of Equation 2.13 is

$$\begin{aligned} &\text{Max} \quad \lambda \\ &\text{s.t.} \quad (b - a + \sum_{(i,j,k)\in\omega} \gamma_{ijk}) \cdot \lambda \leq b - \sum_{(i,j,k)\in\omega} \alpha_{ijk} \\ &\qquad \lambda \leq \underset{(i,j,k)\in\omega}{\min} (q_{ijk}) \\ &\qquad \lambda \geq 0 \end{aligned} \tag{2.14}$$

Rewriting Equation 2.14 as

$$\begin{aligned} &\text{Max} \quad \lambda \\ &\text{s.t.} \quad \lambda \le \frac{b - \sum_{(i,j,k)\in\omega} \alpha_{ijk}}{(b - a + \sum_{(i,j,k)\in\omega} \gamma_{ijk})} \\ &\qquad \lambda \le \min_{(i,j,k)\in\omega} (q_{ijk}) \\ &\qquad \lambda \ge 0 \end{aligned} \tag{2.15}$$

completes the proof.

Because λ_x is the maximal value of Equation 2.9 by giving the feasible solution **x** of Equation 1.1, the maximum of λ_x must be the optimal solution of Equation 2.8. Equation 2.8 is rewritten as

$$\begin{aligned} &\text{Max Min}\left(\frac{b - \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}(\alpha_{ijk}x_{ijk})}{(b - a + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}(\gamma_{ijk}x_{ijk}))}, (1-(1-q_{ijk})x_{ijk}) \text{ for } i,j,k = 1,2,\ldots,n \right) \\ &\text{s.t.} \quad \text{the constraints of (1.1)} \end{aligned} \tag{2.16}$$

Obviously, Equation 2.16 is a special multi-fractional max-min 0-1 programming problem.

3 SOLUTION ALGORITHM

This section presents two efficient algorithms for solving the exact solution of Equation 2.16. The constraint of Equation 2.16 is the same as that of Equation 1.1 – i.e., the lack of a totally unimodular property. The methods in the literature for solving FAP are difficult to implement for Equation 2.16. Hence, implicit enumeration methods are employed. One of the proposed algorithms is modified from the index-based B&B algorithm [8-9], which quickly solves Equation 1.1 [9]. The other is a hybrid of the efficient trade-off concept [29] and index-based B&B algorithm. A numerical example is shown to clarify these algorithms.

3.1 The index-based B&B algorithm

This algorithm performs implicit enumeration of all feasible solutions for solving Equation 2.16. Figure 3 shows part of the tree graph of a 4×4×4 problem used to illustrate the algorithm. The proposed algorithm proceeds from tree nodes from top to bottom and from left to right until all nodes are checked or eliminated by a fathoming test. For each node (i, j, k) residing at level i, a *path* leads to it from the root node. The variables in the path are fixed at 1, and the other variables at level t ($0 \le t \le i$) equal 0. Thus, each node is associated with a set of feasible solutions. For example, node (2, 3, 2) at level 2 in Figure 3 corresponds to the feasible solutions $x_{124} = x_{232} = 1$; other variables at level 1 and 2 are 0, while other variables are undetermined.

Consider the following notations:

$$\bar{\alpha}_i = \min(\alpha_{i11}, \alpha_{i12}, \ldots, \alpha_{inn}) \quad \text{for } i = 1, 2, \ldots, n \tag{3.1}$$

$$\bar{\gamma}_i = \min(\gamma_{i11}, \gamma_{i12}, \ldots, \gamma_{inn}) \quad \text{for } i = 1, 2, \ldots, n \tag{3.2}$$

$$SLA(i) = \sum_{\ell=i}^{n} \bar{\alpha}_\ell = \bar{\alpha}_i + \bar{\alpha}_{i+1} + \ldots + \bar{\alpha}_n \quad \text{for } i = 1, 2, \ldots, n \tag{3.3}$$

$$SLG(i) = \sum_{\ell=i}^{n} \bar{\gamma}_\ell = \bar{\gamma}_i + \bar{\gamma}_{i+1} + \ldots + \bar{\gamma}_n \quad \text{for } i = 1, 2, \ldots, n \tag{3.4}$$

$$SLA(n+1) = SLG(n+1) = 0 \tag{3.5}$$

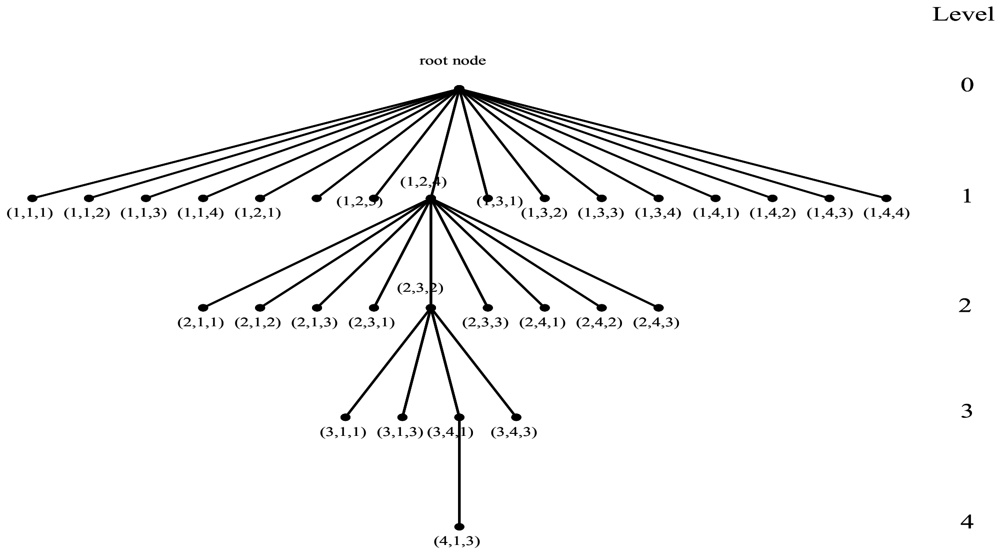


Figure 3: Part of the tree graph of a 4×4×4 problem

Assume that the algorithm has currently reached node (p, j_p, k_p) (p>0) and that nodes (1, j_1, k_1), (2, j_2, k_2),…,(p, j_p, k_p) compose the path — Path (p) — from the root node to the current node. In addition, let

$$TA(0)=SLA(1) \qquad (3.6)$$

$$TG(0)=SLG(1) \qquad (3.7)$$

$$TA(p)=TA(p-1) - SLA(p)+ \alpha_{pj_pk_p} + SLA(p+1) \qquad (3.8)$$

$$TG(p)=TG(p-1)- SLG(p)+ \gamma_{pj_pk_p} + SLG(p+1) \qquad (3.9)$$

Theorem 3.1. Function UPV (p) $=\dfrac{b-TA(p)}{b-a+TG(p)}$ provides an upper bound of the optimal objective value of Equation 2.16 corresponding to Path (p).

Proof. Using Equations 3.3, 3.5, 3.6, and 3.8 yields

$$TA(p)= \alpha_{1j_1k_1} + \alpha_{2j_2k_2} +\ldots+ \alpha_{p-1j_{p-1}k_{p-1}} + \bar{\alpha}_{p+1} + \bar{\alpha}_{p+2} + \ldots + \bar{\alpha}_n \qquad (3.10)$$

Suppose nodes (p+1, j_{p+1}, k_{p+1}), (p+2, j_{p+2}, k_{p+2}),…, and (n, j_n, k_n) compose the optimal solution of Equation 2.16 corresponding to Path (p) and call it Path (n). Then,

$$TA(n)= \alpha_{1j_1k_1} + \alpha_{2j_2k_2} +\ldots+ \alpha_{p-1j_{p-1}k_{p-1}} + \alpha_{p+1j_{p+1}k_{p+1}} + \alpha_{p+2j_{p+2}k_{p+2}} + \ldots + \alpha_{nj_nk_n} \qquad (3.11)$$

Using Equation 3.1 produces $\alpha_{p+1j_{p+1}k_{p+1}} \geq \bar{\alpha}_{p+1}$, $\alpha_{p+2j_{p+2}k_{p+2}} \geq \bar{\alpha}_{p+2}$, …, and $\alpha_{nj_nk_n} \geq \bar{\alpha}_n$. Thus,

$$TA(n) \geq TA(p). \qquad (3.12)$$

Similarly, it can be verified that

$$TG(n) \geq TG(p). \qquad (3.13)$$

Using Equation 3.12 and Equation 3.13 produces

$$\frac{b-TA(p)}{b-a+TG(p)} \geq \frac{b-TA(n)}{b-a+TG(n)}$$

and completes the proof.

The following theorem is derived from Equation 2.16.

Theorem 3.2 If (i, j, k) is a node of Path (p), then q_{ijk} gives an upper bound of the optimal objective function value of Equation 2.16 corresponding to Path (p).

Let z* represent the largest known objective function value, which is associated with a feasible solution. Using Theorems 3.1 and 3.2, the solution procedure of the index-based B&B algorithm is described as follows:

Step 1: Find $\overline{\alpha}_i$ and $\overline{\gamma}_i$ for i =1, 2, …, n.

Step 2: Compute $SLA(i)$ and $SLG(i)$ for i =1, 2, …, n+1. Let $TA(0)= SLA(1)$ and $TG(0)= SLG(1)$.

Step 3: Use $x_{111}= x_{222}= …= x_{nnn}= 1$ as the initial feasible solution and its objective function value as z*.

Step 4: Let p= 1 and proceed downward to node (1, 1, 1) from the root node.

Step 5: Suppose node (p, j_p, k_p) is reached.

Let $TA(p) = TA(p-1) - SLA(p) + \alpha_{pj_pk_p} + SLA(p+1)$ and

$TG(p) = TG(p-1) - SLG(p) + \gamma_{pj_pk_p} + SLG(p+1)$.

(1) If $q_{pj_pk_p} \leq$ z* or UPV (p)≤ z*, then node (p, j_p, k_p) is fathomed and moves toward the immediate right node of (p, j_p, k_p) at level p. If node (p, j_p, k_p) is the last node of level p, then move upward to the first node right of (p–1, j_{p-1}, k_{p-1}) at level p–1 and update $p \leftarrow p-1$.

(2) If $q_{pj_pk_p} >$ z*, UPV (p)> z*, and $p \neq n$, then move downward from (p, j_p, k_p) to the first node at level p+1 and update $p \leftarrow p+1$.

(3) If $q_{pj_pk_p} >$ z*, *UPV* (p)> z*, and p= n, then replace z* with UPV (p). Move upward to the first node right of (p–1, j_{p-1}, k_{p-1}) at level p–1 and update $p \leftarrow p-1$.

Step 6: If p= 0, then stop; otherwise, proceed to Step 5.

The main concept of the index-based B&B algorithm is as follows. Whenever the algorithm reaches a new node (p, j_p, k_p), the two upper bounds (UPV (p) and $q_{pj_pk_p}$) are calculated for that path before proceeding forward to any branches leaving from node (p, j_p, k_p). If both UPV (p) and $q_{pj_pk_p}$ are larger than z*, then a better feasible solution may be found by proceeding from node (p, j_p, k_p). However, if UPV (p) or $q_{pj_pk_p}$ are less than or equal to z*, no better feasible solution can be obtained by proceeding forward from node (p, j_p, k_p). That is, (p, j_p, k_p) is fathomed. The algorithm locates the exact optimal solution.

3.2 The *f*-*g* trade-off algorithm

Because the performances of team members are equally emphasised to maximise the team performance, the *f*-*g* trade-off algorithm is proposed. The *f*-*g* trade-off algorithm balances the maximum of the manager's performance and minimum of the workers' performances to maximise team performance. In the algorithm, some procedures are repeated to solve updated problems with different parameters. In each solution, the maximum of the manager's performance decreases (or remains unchanged), and the minimum of the workers' performances simultaneously increases until the maximal team performance is reached.

To obtain the maximal manager's performance, Equation 3.14 is solved.

$$\text{Max } f^{(t)} = \frac{b - \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \alpha_{ijk}^{(t)} x_{ijk}}{b - a + \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} \gamma_{ijk} x_{ijk}}$$

s.t. the constraints of (1.1) (3.14)

where t is an index of iteration and $t = 0$ indicates the original problem parameter. Equation 3.14 is a variant of the 3D axial fractional assignment problem. To solve Equation 3.14 efficiently when $t = 0$, the concept of the fraction cost penalty method (FCPM) [30] is used to obtain a good initial feasible assignment. Cubic matrix $[\psi_{ijk}] = \left[\dfrac{\frac{b}{n}-\alpha_{ijk}}{\frac{b-a}{n}+\gamma_{ijk}}\right]$ is defined as the cost coefficients of Equation 1.1, but its objective function is changed to the maximal case. An approach of the cubic FCPM is applied to $[\psi_{ijk}]$. In $[\psi_{ijk}]$, the penalty for each row, column, and height was calculated, defined as the difference between the two largest elements of the row, column, and height. In the row, column, or height with the largest penalty, the largest ψ_{ijk} must be selected and its row, column, and height crossed out. The procedure is repeated to obtain a feasible assignment and its corresponding objective function value. The obtained objective function value and assignment are z* and ω*, respectively. Other than in the cubic FCPM, z* and ω* represent the largest known objective function value and associated assignment, respectively.

To optimise Equation 3.14, the revised branch and bound (R-B&B) algorithm is defined. It is used repeatedly in the *f*-*g* trade-off algorithm. The R-B&B algorithm is reduced from the index-based B&B algorithm.

The R-B&B algorithm:

All steps of the R-B&B algorithm are the same as those in the index-based B&B algorithm, except that the condition $q_{pj_pk_p} \leq$ z* is deleted from (1) of Step 5 and $q_{pj_pk_p} >$ z* is deleted from (2) and (3) of Step 5.

Whenever the R-B&B algorithm is performed, it produces the optimal objective function value $f^{(t)}$ of Equation 3.14 and its associated assignment $\omega^{(t)}$. Equation 3.15 is then used to compute $g^{(t)}$:

$$g^{(t)} = \underset{(i,j,k)\in\omega^{(t)}}{\text{Min}} \; (q_{ijk}) \tag{3.15}$$

where $g^{(t)}$ is the minimum of all workers' performance at iteration t. The largest known performance of the team z* can then be updated by using $f^{(t)}$ and $g^{(t)}$.

To promote z*, the cubic matrix $[\alpha_{ijk}]$ in (3.14) is updated; the updated rule of $[\alpha_{ijk}]$ is

(1) if $q_{ijk} >$ z*, α_{ijk} remains unchanged.

(2) If $q_{ijk} \leq$ z*, α_{ijk} is replaced with M ($M \to \infty$) - i.e., fathom out that α_{ijk} is useless.

The rule means that $f^{(t)}$ decreases (or remains unchanged) and $g^{(t)}$ increases at each iteration for z* to reach the maximum. Therefore, the flowchart in Figure 4 is used to depict the proposed *f*-*g* tradeoff algorithm.

3.3 Numerical example

The following numerical 3×3×3 fuzzy 3D axial assignment problem illustrates the two proposed algorithms. Table 1 shows the job costs and quality.

According to Equation 2.16, the numerical example can be formulated as

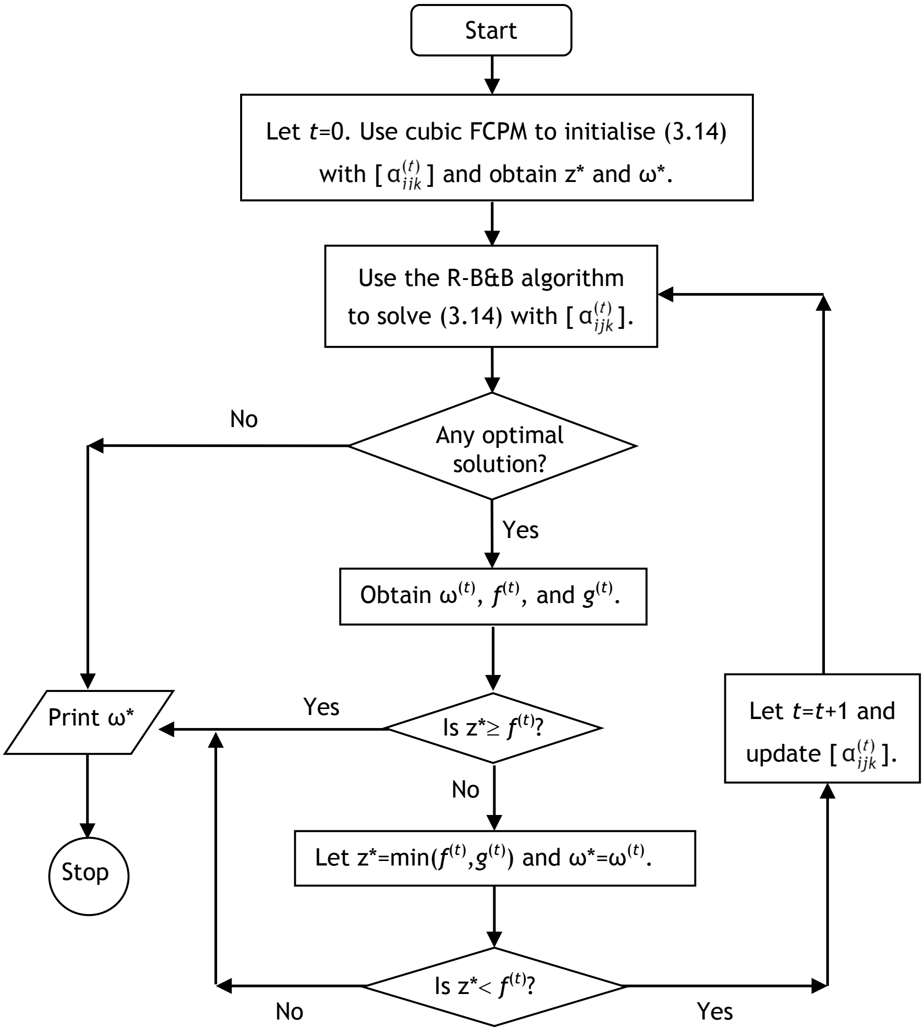
$$\text{Max Min}\left(\frac{612-(101x_{111}+\ldots+104x_{333})}{303+45.31x_{111}+\ldots+26.09x_{333}},\ 1-(1-0.64)x_{111},\ \ldots,\ 1-(1-0.69)x_{333}\right)$$

$$\text{s.t.} \quad \sum_{j=1}^{3}\sum_{k=1}^{3} x_{ijk} = 1 \quad \text{for} \quad i = 1, 2, 3$$

$$\sum_{i=1}^{3}\sum_{k=1}^{3} x_{ijk} = 1 \quad \text{for} \quad j=1,\,2,\,3 \tag{3.16}$$

$$\sum_{i=1}^{3}\sum_{j=1}^{3} x_{ijk} = 1 \quad \text{for} \quad k=1,\,2,\,3$$

$$x_{ijk} \in \{0,\,1\} \quad \text{for } i,\, j,\, k=1,\,2,\,3$$



Start
Let t=0. Use cubic FCPM to initialise (3.14) with [$\alpha_{iik}^{(t)}$] and obtain z* and ω*.
Use the R-B&B algorithm to solve (3.14) with [$\alpha_{ijk}^{(t)}$].
Any optimal solution?
No
Yes
Obtain $\omega^{(t)}$, $f^{(t)}$, and $g^{(t)}$.
Is z*≥ $f^{(t)}$?
Yes
No
Print ω*
Stop
Let z*=min($f^{(t)}$,$g^{(t)}$) and ω*=$\omega^{(t)}$.
Is z*< $f^{(t)}$?
No
Yes
Let t=t+1 and update [$\alpha_{ijk}^{(t)}$].

Figure 4: Flow chart of the *f*-*g* trade-off algorithm

Using Equation 2.2, α_{ijk} and β_{ijk} are transformed into γ_{ijk}, as shown in Table 2, *a* = 309, and *b* = 612. The algorithms are then used to solve Equation 3.16.

(I) The index-based B&B algorithm

Iteration 0:

Step 1: Find $\overline{\alpha}_1$ = 101, $\overline{\alpha}_2$ = 104, $\overline{\alpha}_3$ = 104, $\overline{\gamma}_1$ = 28.17, $\overline{\gamma}_2$ =24, and $\overline{\gamma}_3$ =26.03.

Step 2: Determine *SLA*(1) = 309, *SLA*(2) = 208, *SLA*(3) = 104, *SLA*(4) = 0, *SLG*(1) = 78.2, *SLG*(2) = 50.03, *SLG*(3) = 26.03, *SLG*(4) = 0, *TA*(0) = 309, and *TG*(0)= 78.2.

Step 3: Use x_{111}= x_{222}= x_{333}= 1 as the initial feasible solution and obtain z*= 0.60.

Step 4: Let p = 1 and proceed downward to node (1, 1, 1) from the root node.

Step 5: Identify $TA(1)$ = 309 and $TG(1)$ = 95.34. Because q_{111}= 0.64 > 0.6 = z* and UPV(1) = 0.76 > 0.6 = z*, proceed downward to node (2, 2, 2) and update p = 2.

Step 6: Let p = 2 and proceed to Step 5.

Iteration 1:

Step 5: Because q_{222}= 0.6 ≤ z*, node (2, 2, 2) is fathomed. Proceed toward node (2, 2, 3).

Step 6: Let p = 2 and proceed to Step 5.

Table 1: The cubic matrix of job costs and quality

(101,130) 0.64 (173,193) 0.71 (168,196) 0.64	(163,188) 0.77 (152,173) 0.61 (148,169) 0.64	(126,153) 0.59 (127,154) 0.59 (169,194) 0.69
(186,212) 0.65 (105,129) 0.61 (178,196) 0.75	(163,189) 0.62 (137,157) 0.60 (104,133) 0.69	(177,201) 0.59 (133,162) 0.57 (171,190) 0.68
(142,164) 0.65 (156,185) 0.68 (126,147) 0.57	(185,204) 0.73 (147,174) 0.61 (132,151) 0.63	(169,190) 0.75 (153,174) 0.69 (104,122) 0.69

Legend:

$(\alpha_{ijk}, \beta_{ijk})$
q_{ijk}

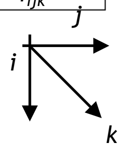


Table 2: The cubic matrix of γ_{ijk}

$[\gamma_{ijk}]$=

45.31 28.17 43.75	32.47 34.43 32.81	45.76 36.23 45.76
40 39.34 24	41.94 33.33 42.03	40.68 50.88 27.94
33.85 42.65 36.84	26.03 44.26 30.16	28 30.43 26.09

<u>*Iteration* 2:</u>

Step 5: Determine q_{223} = 0.69 > 0.6 = z*, *TA*(2) = 309, and *TG*(2) = 113.37. Because UPV(2)= 0.728 > 0.6= z*, proceed downward to node (3, 3, 2) and update *p* = 3.

Step 6: Let *p* = 3 and proceed to Step 5.

<u>*Iteration* 3:</u>

Step 5: Determine q_{332}= 0.69 > 0.6= z*, *TA*(3) = 358, and *TG*(3) = 118.04. Because UPV(3)= 0.604> 0.6= z* and *p*= 3= *n*, replace z*= 0.604 and move upward to node (2, 3, 2) and to the first node right of (2, 2, 3) of level 2. Update *p* = 2.

Step 6: Let *p* = 2 and proceed to Step 5.

The remaining procedures repeat these iterations and obtain the optimal solution x_{111}= x_{223} = x_{332}= 1 and optimal value z*= 0.604.

(II) The *f*-*g* trade-off algorithm

<u>When *t* = 0:</u>

Equation 3.14 produces Equation 3.17.

$$\text{Max } f = \frac{612-(101x_{111}+\ldots+104x_{333})}{303+45.31x_{111}+\ldots+26.09x_{333}}$$

s.t. the constraints of Equation 3.16 (3.17)

After using the cubic FCPM and R-B&B program to solve Equation 3.17, $\omega^{(0)}$ = (x_{111}, x_{222}, x_{333}), $f^{(0)}$=0.662, and $g^{(0)}$ = 0.6. Let ω*= $\omega^{(0)}$ and z*= Min (0.662, 0.6)= 0.6 = $g^{(0)}$.

<u>*When t* = 1:</u>

Update [$\alpha_{ijk}^{(0)}$] by replacing α_{131}, α_{132}, α_{222}, α_{231}, α_{232}, and α_{313} with M and obtain [$\alpha_{ijk}^{(1)}$]. Solving Equation 3.17 again by using [$\alpha_{ijk}^{(1)}$] produces $\omega^{(1)}$ = (x_{111}, x_{223}, x_{332}), $f^{(1)}$=0.604, and $g^{(1)}$= 0.64. Because $f^{(1)}$ = 0.604 > 0.6= z*, update ω*= $\omega^{(1)}$ and z*= Min ($f^{(1)}$, $g^{(1)}$)= 0.604.

<u>*When t* = 2:</u>

Update [$\alpha_{ijk}^{(1)}$] by replacing α_{131}, α_{132}, α_{222}, α_{231}, α_{232}, and α_{313} with M and obtain [$\alpha_{ijk}^{(2)}$]. Solving Equation 3.17 again by using [$\alpha_{ijk}^{(2)}$] produces $\omega^{(2)}$= (x_{111}, x_{223}, x_{332}), $f^{(2)}$= 0.604, and $g^{(2)}$ = 0.64. Because $f^{(2)}$= 0.604 = z*, the algorithm stops. The current ω*= (x_{111}, x_{223}, x_{332}) and z*= 0.604 are the optimal solution and optimal objective function value, respectively.

4 COMPUTATIONAL RESULTS AND DISCUSSION

The computational test verifies the efficiency of the proposed algorithms. Both proposed algorithms were coded in Visual Basic 6.0 and compared with the LINGO 11 package by using the global optimal function. They were all run on an Intel Pentium III-500 CPU.

Problem sizes from 3×3×3 to 10×10×10 were tested. For each problem of N×N×N size, uniform distribution numbers between 10 and 10+20N were randomly generated as α_{ijk}s, and uniform distribution numbers between 6N to 10N were randomly generated as the difference between β_{ijk}s and α_{ijk}s. Uniform distribution numbers between 0.6 and 1.0 were generated as q_{ijk}s. For each problem size, 30 problems were generated. Table 3 shows the average computational times (in seconds) of the two proposed algorithms and the LINGO package. The results show that both proposed algorithms have shorter computational times than the LINGO package.

Table 3: Computational results when the quality interval is [0.6, 1]

Problem size	The index-based B&B algorithm	The *f*-*g* trade-off algorithm	LINGO
	Average time (s)	Average time (s)	Average time (s)
3×3×3	3.33×10^{-4}	2×10^{-3}	33.192
4×4×4	3.67×10^{-3}	6×10^{-3}	42.250
5×5×5	9.71×10^{-3}	1.47×10^{-2}	85.830
6×6×6	6.91×10^{-2}	0.154	162.452
7×7×7	0.396	1.117	544.152
8×8×8	2.111	7.721	T. L.[a]
9×9×9	14.788	65.314	T. L.
10×10×10	71.816	325.255	T. L.

[a] T. L. denotes the time limit exceeds 1,000 CPU seconds.

In general cases, the manager or any worker may be the person with the lowest performance and may determine the team's performance. Here, two special cases are presented. In the first case, the company is assumed to support the team with sufficient funds, e.g., 10 times of *b*. The computational results of the three approaches are shown in Table 4. The results show that the index-based B&B algorithm becomes more efficient. The time taken by the LINGO package also reduces. The *f*-*g* trade-off algorithm, on the contrary, becomes inefficient. This is because the optimal values $f^{(t)}$ of Equation 3.14 approach 1, such that the fathom speed of α_{ijk} decreases. To discuss this further theoretically, $b\to\infty$ moves the membership function in Figure 2 close to the horizontal line $\mu_T(c_T)$ =1 and

$$\operatorname{limit}_{b\to\infty} \frac{b-\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}(\alpha_{ijk}x_{ijk})}{b-a+\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}(\gamma_{ijk}x_{ijk})}=1,$$

i.e., the manager's performance approaches 1. The manager's performance is greater than the maximal of q_{ijk}. Thus the manager never determines the team performance; team performance is decided by a worker. The workers also obtain sufficient expense and reach the highest quality (performance). Therefore, Equation 2.16 is simplified to

$$\text{Max Min}\Big((1-(1-q_{ijk})x_{ijk}) \text{ for } i,j,k=1,2,\ldots,n\Big)$$
$$\text{s.t. the constraints of Equation 1.1.} \qquad (4.1)$$

Equation 4.1 is a variant of the 3D axial bottleneck assignment problem. Hence, a proper algorithm for the 3D axial bottleneck assignment problem will improve the efficiency if $b\to\infty$ is known in advance.

Table 4: Computational results using 10*b*

Problem size	The index-based B&B algorithm	The *f*-*g* trade-off algorithm	LINGO
	Average time (s)	Average time (s)	Average time (s)
3×3×3	3.33×10^{-4}	3.33×10^{-3}	29.067
4×4×4	1.67×10^{-3}	1.01×10^{-2}	43.930
5×5×5	3.33×10^{-3}	3.81×10^{-2}	65.052
6×6×6	4.01×10^{-3}	0.299	87.776
7×7×7	1.37×10^{-2}	2.065	159.977
8×8×8	2.64×10^{-2}	15.030	402.167
9×9×9	6.88×10^{-2}	106.987	T. L.[a]
10×10×10	0.104	676.851	T. L.

[a] T. L. denotes that the time limit exceeds 1,000 CPU seconds.

In the second case, *b* is small; that is, the team is underfinanced. For example, define the manager's performance as 0 when the total cost is greater than (*a*+*b*)/2. The computational results of the three approaches are shown in Table 5. The results show that the *f*-*g* trade-off algorithm is the most efficient. Theoretically, when

$$b \le (Z_{[\alpha]}^{*} - a \times \min(q_{ijk}))/(1 - \min(q_{ijk})),$$

where $Z_{[\alpha]}^{*}$ is the optimal value of the crisp 3D axial assignment problem (min) using $[\alpha_{ijk}]$ as the cost coefficient, the *b* is too small for the manager's performance to reach the minimum of q_{ijk}. Team performance is then determined mainly by the manager, despite the max-min objective function equalising the performance of the *n* workers and the manager. Thus Equation 2.16 is simplified to Equation 3.14 with *t*=0. Hence, a proper specific algorithm for the 3D axial fractional assignment problem will improve the efficiency.

Table 5: Computational results using (*a*+*b*)/2

Problem size	The index-based B&B algorithm	The *f*-*g* trade-off algorithm	LINGO
	Average time (s)	Average time (s)	Average Time (s)
3×3×3	7.00×10^{-4}	4.00×10^{-3}	8.459
4×4×4	4.67×10^{-3}	4.00×10^{-3}	24.475
5×5×5	9.00×10^{-3}	2.04×10^{-2}	48.587
6×6×6	7.01×10^{-2}	6.28×10^{-2}	82.735
7×7×7	0.430	0.424	262.198
8×8×8	2.919	1.904	891.336
9×9×9	22.442	15.663	T. L.[a]
10×10×10	129.076	86.651	T. L.

[a] T. L. denotes that the time limit exceeds 1,000 CPU seconds.

5 CONCLUSIONS

This paper first formulates a fuzzy 3D axial assignment problem model to overcome the actual uncertain environment, and then focuses on developing algorithms to obtain the exact solutions of the proposed model efficiently. The initially-constructed fuzzy 3D axial assignment problem is a mixed nonlinear programming problem. For simplicity, it is transformed into a special multi-fractional max-min 0-1 programming problem. An index-based B&B algorithm and the *f*-*g* trade-off algorithm are developed. Computational results show that the proposed methods are both more efficient than LINGO in every test. In addition, two special cases are discussed. One is when the company amply supports the team with funds, while the other is when the team is underfunded. With respect to computational time, the index-based B&B algorithm is better for the former case. However, the *f*-*g* trade-off algorithm is better for the latter case. These two special cases are further considered from a theoretical view. For the case with unbounded funds, the model is reduced to a 3D axial bottleneck assignment problem; for the case with rare funds, the model is reduced to a 3D axial fractional assignment problem. Hence, the developments of algorithms for these two problems would be interesting and practical directions for future research. Moreover, a sensitivity analysis procedure for identifying the largest sensitivity range – one that maintains the current optimal assignment – is worth studying.

REFERENCES

[1] **Burkard, R., Dell'Amico, M. & Martello, S.** 2012. *Assignment problems.* 1st edition, SIAM.

[2] **Pentico D.W.** 2007. Assignment problems: A golden anniversary survey. *European Journal of Operational Research,* 176(2), pp. 774-793.

[3] **Balas, E. & Saltzman, M.J.** 1991. An algorithm for the three-index assignment problem. *Operations Research,* 39(1), pp. 150-161.

[4] **Poore, A.B. & Robertson, A.J.** 1997. A new Lagrangian relaxation based algorithm for a class of multidimensional assignment problems. *Computational Optimization and Applications*, 8(2), pp. 129-150.

[5] **Pierskalla, W.P.** 1968. The multidimensional assignment problem. *Operations Research*, 16(2), pp. 422-431.

[6] **Kim, B.J., Hightower, W.L., Hahn, P.M., Zhu, Y.R. & Sun, L.** 2010. Lower bounds of the axial three-index assignment problem, *European Journal of Operational Research*, 202(3), pp. 654-668.

[7] **Adlakha V. & Arsham, H.** 1998. Managing cost uncertainties in transportation and assignment problems. *Journal of Applied Mathematics & Decision Sciences*, 2(1), pp. 65-104.

[8] **Burkard, R.E. & Rudolf, R.** 1993. Computational investigations on 3-dimensional axial assignment problems. *Belgian Journal of Operations Research, Statistics & Computer Science*, 32(1-2), pp. 85-98.

[9] **Pasiliao, E.L., Pardalos, P.M. & Pitsoulis, L.S.** 2005. Branch and bound algorithms for the multidimensional assignment problem. *Optimization Methods & Software*, 20(1), pp. 127-143.

[10] **Sergeev, S.I.** 2006. The three-dimensional assignment and partition problems - New lower bounds. *Automation & Remote Control*, 67(2), pp. 242-250.

[11] **Huang, G. & Lim, A.** 2006. A hybrid genetic algorithm for the three-index assignment problem, *European Journal of Operational Research*, 172(1), pp. 249-257.

[12] **Lia, F., Xu, L., Jina, D.C. & Wang, H.** 2012. Study on solution models and methods for the fuzzy assignment problems. *Expert Systems with Applications*, 39(12), pp. 11276-11283.

[13] **Lin, C.J. & Wen, U.P.** 2004. A labeling algorithm for the fuzzy assignment problem. *Fuzzy Sets and Systems*, 142(3), pp. 373-391.

[14] **Mukherjee, S. & Basum, K.** 2012. Solution of a class of Intuitionistic fuzzy assignment problem by using similarity measures. *Knowledge-Based Systems*, 27, pp. 170-179.

[15] **Lin, C.J.** 2013. Assignment problem for team performance promotion under fuzzy environment. *Mathematical Problems in Engineering*, 2013, pp. 1-10.

[16] **Lin, C.J., Wen, U.P. & Lin, P.Y.** 2011. Advanced sensitivity analysis of the fuzzy assignment problem. *Applied Soft Computing*, 11(8), pp. 5341-5349.

[17] **Liu, L.Z. & Li, Y.Z.** 2006. The fuzzy quadratic assignment problem with penalty: New models and genetic algorithm. *Applied Mathematics and Computation*, 174(2), pp. 1229-1244.

[18] **Belacela, N. & Boulasselb, M.R.** 2001. Multicriteria fuzzy assignment method: A useful tool to assist medical diagnosis. *Artificial Intelligence Medicine*, 21(1-3), pp. 201-207.

[19] **Dubois, D. & Fortemps, P.** 1999. Computing improved optimal solutions to max-min flexible constraint satisfaction problems. *European Journal of Operational Research*, **118**(1), pp. 95-126.

[20] **Huang, D.K., Chiu, H.N., Yeh, R.H. & Chang, J.H.** 2009. A fuzzy multi-criteria decision making approach for solving a bi-objective personnel assignment problem. *Computers & Industrial Engineering*, 56(1), pp. 1-10.

[21] **Ridwan, M.** 2004. Fuzzy preference based traffic assignment problem. *Transportation Research Part C: Emerging Technologies*, 12(3-4), pp. 209-233.

[22] **Feng, Y. & Yang, L.** 2006. A two-objective fuzzy k-cardinality assignment problem. *Journal of Computational & Applied Mathematics*, 197(1), pp. 233-244.

[23] **Bashiri, M. & Badri, H.** 2011. A group decision making procedure for fuzzy interactive linear assignment programming. *Expert Systems with Applications*, 38(5), pp. 5561-5568.

[24] **Liu, L. & Gao, X.** 2009. Fuzzy weighted equilibrium multi-job assignment problem and genetic algorithm. *Applied Mathematical Modelling*, 33(10), pp. 3926-3935.

[25] **Mutingi, M. & Mbohwa, C.** 2014, A fuzzy-based particle swarm optimisation approach for task assignment in home healthcare. *The South Africa Journal of Industrial Engineering*, 25(3), pp. 84-95.

[26] **Klir, G.J., Clair, U.S. & Yuan, B.** 1997, *Fuzzy set theory: Foundations and applications.* 1st edition, Prentice Hall PTR.

[27] **Baruah, H.K.** 2011, Theory of fuzzy sets: The case of subnormality. *International Journal of Energy, Information and Communications*, **2**(3), pp. 1-8.

[28] Bellman, R.E. & Zadeh, L.A. 1970. Decision-making in a fuzzy environment. *Management Science*, **17B**(4), pp. 141-164.

[29] **Geetha, S. & Vartak, M.N.** 1989. Time-cost tradeoff in a three dimensional assignment problem. *European Journal of Operational Research*, 38(2), pp. 255-258.

[30] **Joshi, V.D. & Gupta, N.** 2009. A heuristic for obtaining a better initial solution for the linear fractional transportation problem. *Bulletin of Australian Society of Operations Research*, 28(4), pp. 45-49.