

An interactive algorithm for linear assignment problem on fuzzy graphs

J. Tayyebi¹, E. Hosseinzadeh² and M. Niksirat³

¹Department of Industrial Engineering, Faculty of Industrial and Computer Engineering, Birjand University of Technology, Birjand, Iran.

²Department of Mathematics, Kowsar University, Bojnord, Iran.

³Department of Computer Science, Faculty of Industrial and Computer Engineering, Birjand University of Technology, Birjand, Iran.

javadtayyebi@birjandut.ac.ir, e.hosseinzadeh@kub.ac.ir, niksirat@birjandut.com

Abstract

The assignment problem is a foundational combinatorial optimization problem that requires a decision maker to allocate agents to tasks in order to minimize the total cost incurred. From a graph-theoretic perspective, this problem can be framed as a matching problem on a bipartite graph. This paper presents a natural extension of the traditional assignment problem by considering fuzzy graphs as the underlying framework. This extension proves significant in practical situations where expert opinions contribute to assessing the relative significance level of agents (or tasks) and the satisfaction levels associated with different assignments. The main goal for the decision maker is not just to minimize costs but also to consider expert opinions. This paper addresses this problem by first formulating it as a tri-objective optimization problem. Subsequently, the ϵ -constraint method is applied to propose an interactive algorithm designed to identify satisfactory solutions. Finally, a practical example of assigning teachers to schools is provided to demonstrate the algorithm in a real-world application.

Keywords: Assignment problem, combinatorial optimization, fuzzy graph, significance level, satisfaction level, ϵ -constraint method, interactive algorithm.

2020 MSC: 90B80, 05C72.

1 Introduction

The assignment problem involves m agents and n tasks, where each agent can be assigned to perform any task at a specific cost. In this paper, we assume that $m \geq n$; however, the scenario where $m < n$ can similarly be analyzed. Under this assumption, the objective is to complete all tasks by assigning exactly one agent to each task, with each agent handling at most one task, while minimizing the total cost of the assignments. One can simply formulate a zero-one linear optimization problem as follows:

$$\min \quad \sum_{i \in V_1} \sum_{j \in V_2} c_{ij} x_{ij} \quad (1a)$$

$$\text{s.t.} \quad \sum_{j \in V_2} x_{ij} \leq 1 \quad \forall i \in V_1, \quad (1b)$$

$$\sum_{i \in V_1} x_{ij} = 1 \quad \forall j \in V_2, \quad (1c)$$

$$x_{ij} \in \{0, 1\} \quad \forall i \in V_1, j \in V_2, \quad (1d)$$

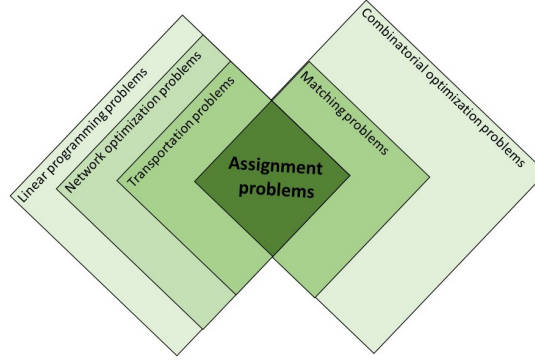


Figure 1: The position of assignment problems among optimization problems

Here, V_1 and V_2 represent the sets of agents and tasks, respectively. Hence, we have $|V_1| = m \geq n = |V_2|$. Moreover, c_{ij} represents the cost of assigning the j -th task to the i -th agent. The variable x_{ij} is a binary variable that indicates whether the j -th task is assigned to the i -th agent.

Although the formulation presented in (1) is a zero-one linear programming problem, one can replace the constraint (1d) with the non-negativity constraint $x_{ij} \geq 0$, utilizing the unimodularity property of the problem [1]. This modification allows the problem to be treated as a linear programming problem, which can subsequently be solved using the well-known Simplex method [2].

From the perspective of graph theory, the problem can be represented on a weighted bipartite graph, where the objective is to find a matching that minimizes the sum of its arc weights. Figure 1 illustrates the position of assignment problems within the broader context of optimization problems.

The assignment problem has numerous applications in various fields, including:

Task Allocation in Workforce Management: The assignment problem can be utilized to optimally assign employees to tasks or projects based on their skills and availability, thus maximizing productivity while minimizing costs [25].

Scheduling Problems: It is widely employed in scheduling applications, such as assigning machines to jobs, workers to shifts, or classrooms to teachers, ensuring efficient resource use within various constraints [22].

Healthcare Resource Management: Hospitals use the assignment problem to allocate resources such as medical staff or equipment to patients or departments, ensuring quality care while minimizing costs [18].

Transportation and Logistics: In the transportation sector, the assignment problem helps determine optimal routes for delivery vehicles, minimizing shipping costs while considering time and distance factors [7].

Social Networks: The assignment problem in social networks can be used to optimally allocate users to groups or events based on their interests and prior interactions, enhancing engagement. It can also help match targeted content or advertisements to users, improving the effectiveness of social media marketing. Additionally, it can identify influential users for collaboration on projects, thereby facilitating networking and resource sharing [3].

Moreover other applications include resource allocation in cloud computing [15], school assignment [10], network routing [14], marketing and advertisement [20], facility location-allocation [28].

1.1 Literature review

In recent years, using fuzzy theory to address assignment problems has received growing attention in operational research. Assignments involve allocating resources to tasks while considering costs and benefits, often facing challenges due to uncertainty in the data. Researchers have used fuzzy theory methodologies to address these issues.

One significant contribution is the work of Emrouznejad et al. (2012), who proposed an alternative formulation for the fuzzy assignment problem, enhancing classical approaches by embracing real-world uncertainties and expanding practical applications [9]. Following this, Gani and Mohamed (2013) introduced a novel ranking method for solving fuzzy assignment problems, refining decision-making processes by providing clearer prioritization of outcomes [11].

Moreover, Kumar and Hussain (2014) focused on balanced intuitionistic fuzzy assignment problems, integrating both membership and non-membership degrees to enhance the decision-making framework [17]. Lin and Wen (2004)

developed a labeling algorithm for fuzzy assignment problems, providing a systematic approach to evaluate assignment options [19].

Sakawa and Akay (2008) explored fuzzy assignment problems within multi-objective decision-making, advancing the optimization of decision processes under uncertainty [25]. Gurukumaresan et al. (2021) employed centroid methods to derive optimal solutions for fuzzy assignment problems, refining methodologies and enhancing applicability [13]. Kumar (2019) proposed a software-based approach integrating intuitionistic fuzzy sets into solid assignment problems, emphasizing the importance of accessible tools for practitioners [16].

Recent advances in fuzzy assignment problems have explored various extensions to handle more complex forms of uncertainty. Sanjana and Ramesh (2024) addressed the Intuitionistic Fuzzy Assignment Problem (IFAP) using Trapezoidal Intuitionistic Fuzzy Numbers (TriFNs) and compared the performance of Hungarian, Brute Force, and Greedy methods with interval arithmetic operations [26]. Devi et al. (2025) introduced Pythagorean Octagonal Neutrosophic Fuzzy Numbers (PONFN) for addressing assignment problems, employing diverse ranking procedures tailored for PONFNs to capture real-world uncertainties and demonstrating the framework's stability and applicability in decision-making contexts [5]. Devipriya and Indrani (2025) developed approaches for solving fuzzy assignment problems using branch-and-bound methods, where the profit matrix consists of triangular fuzzy numbers, and optimal solutions are obtained using the Hungarian algorithm combined with branch and bound techniques after converting fuzzy numbers to crisp values [6]. Srinivas and Prabakaran (2025) extended the traditional assignment problem to the triangular fuzzy neutrosophic assignment problem (TFNAP), utilizing triangular fuzzy neutrosophic numbers (TFNNs) to represent fuzziness and neutrosophy [27]. Their methodology includes a score function to convert triangular fuzzy neutrosophic values to crisp numbers, and the results demonstrate that the TFNAP framework provides more accurate outcomes in dealing with uncertainties compared to traditional methods. Nasir (2025) proposed a multi-expert, multi-criteria approach to the neutrosophic fuzzy assignment problem, addressing scenarios where data comes from multiple experts or criteria with varying degrees of certainty, and where traditional classical and fuzzy methods often fall short [21]. Bihari and Kumar (2025) proposed a new simplex algorithm for interval-valued Fermatean fuzzy linear programming problems. They established the interval-valued Fermatean fuzzy assignment problem (IVFFAP) as a particular case of IVFFLPP, providing a dedicated algorithm for solving IVFFAP with demonstrated applicability [4]. Ribeiro et al. (2026) conducted a systematic literature review on the reviewer assignment problem (RAP), examining various approaches, techniques, and algorithms [24].

Despite these advancements, existing papers mainly focus on scenarios where costs are defined as fuzzy numbers, often neglecting the dynamics of fuzzy graphs. The current literature also does not sufficiently address the interactive dimensions of the assignment problem, crucial in dynamic settings with changing user preferences.

This paper introduces an interactive algorithm for the assignment problem on fuzzy graphs, ensuring user interaction while managing underlying graph vagueness. By focusing on fuzzy graphs, this study contributes to the literature and opens new applications of fuzzy graph theory, particularly in contexts where expert opinions are vital.

The remainder of this paper is organized as follows: Section 2 provides foundational concepts related to fuzzy graphs. Section 3 presents the problem statement and formulates the tri-objective optimization model. Section 4 develops the proposed interactive algorithm for solving the problem. Section 5 demonstrates the algorithm through practical examples. Finally, Section 6 offers concluding remarks and directions for future research.

2 Preliminaries

This section presents the fundamental notions and notations of fuzzy graphs that will be used throughout the paper.

Let $G(V, A)$ be a directed graph, where V represents the set of nodes and A denotes the set of arcs. Since this paper focuses on assignment problems, we assume that G is a bipartite graph. This means that the node set V can be partitioned into two subsets, V_1 and V_2 , such that for every arc $(i, j) \in A$, it holds that $i \in V_1$ and $j \in V_2$.

Let X be a non-empty universal set. A fuzzy set (FS) $\tilde{A}$ is defined in terms of a function that maps elements from X to the closed interval $[0, 1]$. Mathematically, it can be expressed as:

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) : x \in X\},$$

where $\mu_{\tilde{A}}(x)$ represents the membership degree of the element x in the fuzzy set $\tilde{A}$.

Now, suppose $\tilde{V}$ is a fuzzy set defined on V , and $\tilde{A}$ is also a fuzzy set defined on A . For simplification, we denote the membership value of the arc (i, j) in the fuzzy set $\tilde{A}$ as $\mu_{\tilde{A}}(i, j)$ rather than $\mu_{\tilde{A}}((i, j))$.

A crucial condition for $\tilde{G}(\tilde{V}, \tilde{A})$ to be classified as a fuzzy graph (FG) is given by the following inequality:

$$\mu_{\tilde{A}}(i, j) \leq \min\{\mu_{\tilde{V}}(i), \mu_{\tilde{V}}(j)\}. \quad (2)$$

If this condition is satisfied, then $\tilde{G}(\tilde{V}, \tilde{A})$ is called a fuzzy graph, while $G(V, A)$ is referred to as its underlying graph.

Additionally, in the special case where the membership degree $\mu_{\tilde{V}}(i) = 0$ or $\mu_{\tilde{A}}(i, j) = 0$, we can assert that the corresponding node i (or arc (i, j)) can be removed from the underlying graph G . This simplification is essential for focusing on relevant components of the fuzzy graph, allowing for more efficient analysis and problem-solving.

3 Problem statement

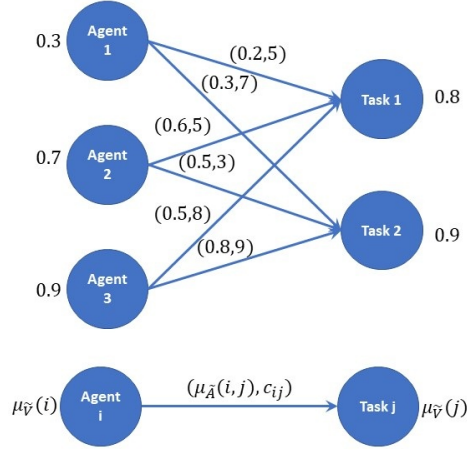


Figure 2: An instance of the assignment problem defined on a fuzzy graph

This section addresses the assignment problems in the context of fuzzy graphs, where the relationships and characteristics between agents and tasks are modeled using expert evaluations.

Let $\tilde{G}(\tilde{V}, \tilde{A})$ denote a bipartite fuzzy graph, where the node set $\tilde{V}$ can be partitioned into two subsets, $\tilde{V}_1$ and $\tilde{V}_2$. The underlying graph of $\tilde{G}$ is represented as $G(V_1 \cup V_2, A)$, where V_1 is the set of agents and V_2 is the set of tasks. The set A includes all allowable assignments (imputations), which do not necessarily correspond to all elements of $V_1 \times V_2$.

In this context, $\mu_{\tilde{V}}(i)$ and $\mu_{\tilde{V}}(j)$ represent the relative importance of agent i and task j , respectively, from the perspective of an expert. Additionally, $\mu_{\tilde{A}}(i, j)$ denotes the relative satisfaction degree attributed by the expert if agent i is assigned to task j . The inequality given in (2) implies that the satisfaction degree of the assignment (i, j) cannot exceed the importance values of agent i and task j . The goal of the assignment F is to identify a maximal subset of A such that no two elements of F are adjacent.

Similar to traditional assignment problems, each assignment (i, j) corresponds to a crisp cost c_{ij} . As previously mentioned, we assume that the number of agents is greater than or equal to the number of tasks, that is, $m \geq n$. This assumption ensures that all tasks are assigned, while some agents may remain unassigned.

To quantify the importance of an arbitrary assignment F , we define an importance degree $id(F)$ as follows:

$$id(F) = \min_{i: \exists (i, j) \in F} \{\mu_{\tilde{V}}(i)\}.$$

In cases where $n > m$, we can similarly define $id(F)$ in terms of the membership values corresponding to the tasks. Additionally, we define a satisfactory degree $sd(F)$ for the assignment F as:

$$sd(F) = \min_{(i, j) \in F} \{\mu_{\tilde{A}}(i, j)\}.$$

Taking into account the expert's evaluations, the objective of solving the assignment problem can be stated as follows:

- Minimize the total cost of the assignment $cs(F) = \sum_{(i, j) \in F} c_{ij}$.
- Maximize the importance degree $id(F)$.

- Maximize the satisfaction degree $sd(F)$.

These objectives represent a multi-faceted approach to problem-solving, ensuring that the assignment is optimized not only in terms of cost but also in relation to the importance and satisfaction levels assigned by the expert.

Example 3.1. Figure 2 illustrates a specific instance of the simple assignment problem involving 3 agents and 2 tasks. In the classical assignment problem, the optimal assignment $F_1 = \{(Agent\ 1, Task\ 1), (Agent\ 2, Task\ 2)\}$ yields a total cost of $5 + 3 = 8$. However, when maximizing the importance degree, the optimal assignment is $F_2 = \{(Agent\ 2, Task\ 2), (Agent\ 3, Task\ 1)\}$, with a total cost of $3 + 6 = 9$. Similarly, when maximizing the satisfaction degree, both F_2 and $F_3 = \{(Agent\ 2, Task\ 1), (Agent\ 3, Task\ 2)\}$ are optimal, with F_3 having a total cost of $8 + 2 = 10$. These costs are computed directly from the edge weights in Figure 2. The trade-offs between cost, importance, and satisfaction are evident: the cheapest solution (cost 8) sacrifices importance and satisfaction, while the solutions with higher importance and satisfaction incur higher costs.

Given that both assignments F_2 and F_3 are deemed optimal under the objective function of maximizing satisfaction, it becomes essential to establish a methodology for identifying an interactive solution that balances the differing objectives. Such an approach would allow decision-makers to navigate the trade-offs between cost efficiency and satisfaction, thereby informing the assignment process more holistically.

Based on this argument, the problem can be formulated as the following tri-objective programming problem.

$$\max(-\sum_{i \in V_1} \sum_{j \in V_2} c_{ij} x_{ij}, sd, id), \quad (3a)$$

$$\sum_{j \in V_2} x_{ij} \leq 1 \quad \forall i \in V_1, \quad (3b)$$

$$\sum_{i \in V_1} x_{ij} = 1 \quad \forall j \in V_2, \quad (3c)$$

$$sd \leq \mu_{\bar{A}}(i, j) + M(1 - x_{ij}) \quad \forall (i, j) \in A, \quad (3d)$$

$$id \leq \mu_{\bar{v}}(i) + M(1 - \sum_{j \in V_2} x_{ij}) \quad \forall i \in V_1, \quad (3e)$$

$$x_{ij} \in \{0, 1\} \quad \forall (i, j) \in A. \quad (3f)$$

It is important to note that the constant M appearing in constraints (3d) and (3e) serves as a penalty parameter to enforce the logical implication that if $x_{ij} = 0$, then the corresponding constraint is effectively deactivated. While a sufficiently large M is required, choosing a value that is too large may lead to numerical instability in the solution process, particularly when using branch-and-bound algorithms in mixed-integer programming solvers. To mitigate this issue, we propose the following computationally practical choice:

$$M = \sum_{i \in V_1} \sum_{j \in V_2} c_{ij} + 1. \quad (4)$$

This value represents the maximum possible total cost plus one, ensuring that any infeasible assignment cost exceeds the cost of any feasible assignment. Since all c_{ij} are non-negative and finite, this bound is both tight and computationally safe. In cases where cost coefficients are particularly large, one may alternatively scale the cost coefficients to keep the value of M within numerically stable ranges.

Remark 3.2. It is worth emphasizing a crucial distinction regarding the total unimodularity property. The classical assignment problem possesses a totally unimodular constraint matrix, which allows the integrality constraint $x_{ij} \in \{0, 1\}$ to be relaxed to $x_{ij} \geq 0$, enabling solution via standard linear programming techniques. However, the reformulated problem (3) includes additional constraints (3d) and (3e), which contain M and the binary variables x_{ij} in a nonlinear logical relationship. The introduction of these constraints does not preserve the total unimodularity of the constraint matrix. Specifically, the Big-M formulation introduces a matrix structure where the columns corresponding to the x_{ij} variables contain both 0/1 coefficients and the large coefficient M , potentially violating the total unimodularity condition (which requires all square subdeterminants to be 0, 1, or -1). Consequently, the linear programming relaxation of (3) may yield fractional solutions, and the integrality constraints $x_{ij} \in \{0, 1\}$ cannot be simply relaxed. This is why we explicitly retain the binary restrictions in formulation (3).

To find a satisfactory solution of this problem, let us apply the ϵ -constraint method for the second and third objective functions. We recall that the ϵ -constraint method in multiobjective optimization is used to find solutions that balance

multiple objectives. Instead of trying to optimize all objectives at once, this method focuses on one objective at a time while treating the other objectives as constraints with specified limits, known as ϵ -values. By solving a series of single-objective optimization problems and varying the ϵ -values, the method generates a set of solutions that represent different trade-offs between the objectives. This set of solutions forms what is known as the Pareto front, which includes all the non-dominated solutions where no other solution is better in all objectives. This approach is efficient because it leverages existing single-objective optimization techniques to explore the trade-offs between multiple objectives [8]. So, we have

$$z(sd, id) = \min \sum_{i \in V_1} \sum_{j \in V_2} c_{ij} x_{ij}, \quad (5a)$$

$$\sum_{j \in V_2} x_{ij} \leq 1 \quad \forall i \in V_1, \quad (5b)$$

$$\sum_{i \in V_1} x_{ij} = 1 \quad \forall j \in V_2, \quad (5c)$$

$$sd \leq \mu_{\bar{A}}(i, j) + M(1 - x_{ij}) \quad \forall (i, j) \in A, \quad (5d)$$

$$id \leq \mu_{\bar{v}}(i) + M(1 - \sum_{j \in V_2} x_{ij}) \quad \forall i \in V_1, \quad (5e)$$

$$x_{ij} \in \{0, 1\} \quad \forall (i, j) \in A. \quad (5f)$$

Problem (5) is a parametric optimization problem with respect to the parameters sd and id . To tackle this problem, we introduce an auxiliary graph $G_{sd, id}(V, A_{sd, id})$ for every fixed pair of values $sd, id \in [0, 1]$. In this auxiliary graph, the node set is identical to the original set V . Furthermore, the arc set $A_{sd, id}$ contains an arc (i, j) if the following conditions are satisfied:

$$\mu_{\bar{v}}(i) \geq id \quad \text{and} \quad \mu_{\bar{A}}(i, j) \geq sd.$$

Based on this definition, the following result is immediate.

Lemma 3.3. *F is an optimal assignment of $G_{sd, id}(V, A_{sd, id})$ if and only if $\mathbf{x}^*$ defined by*

$$x_{ij}^* = \begin{cases} 1 & (i, j) \in F, \\ 0 & (i, j) \notin F, \end{cases} \quad \forall (i, j) \in A,$$

is an optimal solution of problem (5).

Proof. The proof is straightforward.

Consider the optimal value of problem (5), denoted by $z(sd, id)$, which can be regarded as a bivariate function defined over the interval $[0, 1] \times [0, 1]$. It is readily observable that this function is a non-increasing step function, characterized by piecewise constancy with respect to both variables (see Figure 3 for an instance).

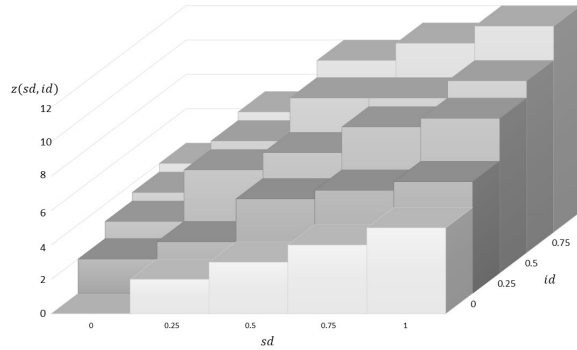


Figure 3: Visualization of a function $z(sd, id)$ at $[0, 1] \times [0, 1]$

For a more in-depth analysis, we note that the function $z(sd, id)$ is not necessarily defined across the entirety of the Cartesian product $[0, 1] \times [0, 1]$. For instance, the function $G_{sd, id}(V, A_{sd, id})$ does not permit any feasible assignment as

sd approaches 1^- . In such a scenario, the problem outlined in (5) is deemed infeasible, resulting in an undefined value for $z(sd, id)$ at that juncture.

Furthermore, we observe that the function behaves as a trivial constant function when sd tends to 0^+ . In this context, the optimal assignment of $G_{sd,id}(V, A_{sd,id})$ coincides with that of $G(V, A)$. It is crucial to highlight that this particular case yields only a weakly efficient solution to the broader problem (3), rather than an efficiently optimal one.

In light of these observations, we propose the introduction of a subinterval $[sd_{min}, sd_{max}] \subseteq [0, 1]$ that excludes the aforementioned edge cases concerning the parameter sd . Here, sd_{max} signifies the maximum value of sd within the interval $[0, 1]$ such that there exists at least one feasible assignment in $G_{sd_{max},id}(V, A_{sd_{max},id})$ for some values $sd \in [0, 1]$. Conversely, sd_{min} is defined as the minimum membership value of the arcs belonging to the set F , where F represents the optimal assignment with respect to the first objective function in the graph $G(V, A)$.

Parallel reasoning can be applied concerning the parameter id . Accordingly, we delineate a desired subinterval $[id_{min}, id_{max}] \subseteq [0, 1]$ for this parameter. The value id_{max} identifies the maximum permissible value of id within the interval $[0, 1]$, such that there is at least one assignment in $G_{sd,id_{max}}(V, A_{sd,id_{max}})$ for some values of sd . Similarly, id_{min} represents the minimum membership value of the nodes belonging to the set F , which pertains to the optimal assignment relative to the first objective function in the graph $G(V, A)$.

Drawing upon the preceding discourse, we shall henceforth constrain the domain of the function $z(sd, id)$ to the product space $[sd_{min}, sd_{max}] \times [id_{min}, id_{max}]$. To investigate the behavior of this function effectively, it suffices to evaluate its values at the designated breakpoints. Let the sequence of breakpoints with respect to id be denoted as

$$(id_{min} =) id_1 < id_2 < \dots < id_p (= id_{max}), \quad (6)$$

and let the breakpoints corresponding to sd be represented as

$$(sd_{min} =) sd_1 < sd_2 < \dots < sd_q (= sd_{max}). \quad (7)$$

We denote by ID and SD the sets of these breakpoints, respectively. Thus, the range of the function $z(sd, id)$ can be characterized as the set

$$Range(z) = \bigcup_{i=1}^q \bigcup_{j=1}^p \{z(sd_i, id_j)\}.$$

Remark 3.4. To ascertain the breakpoints of z , it is adequate to evaluate all of the pairs $(\mu_{\tilde{V}}(k), \mu_{\tilde{A}}(i, j))$ for each $k \in V$ and each $(i, j) \in A$. Each distinct pair that yields a unique value of $z(sd, id)$ qualifies as a breakpoint. Therefore, the breakpoints of z can be identified by solving at most $n \times m$ assignment problems.

4 Algorithm design

Based on the aforementioned points of the previous section, we recall that the function $z(sd, id)$ is non-decreasing with respect to both of its components. This implies that as the levels of importance and satisfaction increase, the value of this function also rises, indicating that a higher cost is imposed on the decision-maker.

Our proposed algorithm is interactive in nature. It first prompts the decision-maker with two questions:

1. What is the minimum level that satisfies your degree of satisfaction for assignments?
2. What is the minimum level that satisfies your degree of importance for agents?

The responses to these two questions are very important because they specifies the decision-maker's minimum relative expectation. Theoretically, by selecting these values, the feasible range of this function is limited to a smaller rectangle. Subsequently, the algorithm searches for values among the elements of sets ID and SD that are not less than these levels. The sets of these values, denoted by

$$ID' = \{id_{p'}, \dots, id_p\}, \quad SD' = \{sd_{q'}, \dots, sd_q\}, \quad (8)$$

represent all the values that the decision-maker is satisfied with, where $id_{p'}$ and $sd_{q'}$ are the minimum values of ID' and SD' respectively. For these sets, we calculate the mean value of the function $z(sd, id)$ that the decision maker can expect to obtain. This can be obtained as

$$\begin{aligned}
z^* &= \frac{1}{(sd_q - sd_{q'})(id_p - id_{p'})} \int_{sd_{q'}}^{sd_q} \int_{id_{p'}}^{id_p} z(sd, id) d(id) d(sd) \\
&= \frac{1}{(sd_q - sd_{q'})(id_p - id_{p'})} \times \sum_{i=p'+1}^p \sum_{j=q'+1}^q z(sd_{i-1}, id_{j-1})(sd_i - sd_{i-1})(id_i - id_{i-1}).
\end{aligned} \tag{9}$$

By using this mean value, the decision-maker can obtain a representative value that reflects the average cost imposed by the function $z(sd, id)$ across the specified sets ID' and SD' . This allows the decision-maker to make informed decisions based on a balanced assessment of the levels of importance and satisfaction.

Although z^* is a good candidate for the total cost, it is possible that this value does not correspond to any value within the range of $z(sd, id)$. Formally, this means $z^* \notin \text{Range}(z)$. If this occurs, we cannot find any values for id and sd such that $z(sd, id) = z^*$, and thus cannot provide a feasible solution to the decision maker.

To address this issue, we identify the closest values within $\text{Range}(z)$ to z^* as follows:

- z_{comp}^L is the largest value in the restricted range of z which is defined as

$$\text{Range}'(z) = \bigcup_{sd \in SD'} \bigcup_{id \in ID'} \{z(sd, id)\}, \tag{10}$$

that is less than or equal to z^* .

- z_{comp}^U is the smallest value in $\text{Range}'(z)$ that is greater than or equal to z^* .

So, we can construct the smallest interval $[z_{comp}^L, z_{comp}^U]$ such that $z^* \in [z_{comp}^L, z_{comp}^U]$. Obviously, these values are chosen such that they are the closest feasible values to the target z^* . Note that if $z^* \in \text{Range}(z)$, then $z_{comp}^L = z_{comp}^U = z^*$.

We then present the decision maker with two alternative solutions corresponding to these bounds:

- **Solution 1 (Lower Bound z_{comp}^L):** This solution corresponds to the cost z_{comp}^L . While this cost is lower than (or equal to) z_{comp}^U , the degrees of importance and satisfaction associated with this solution are also lower (or equal to) than those of the other solution.
- **Solution 2 (Upper Bound z_{comp}^U):** This solution corresponds to the cost z_{comp}^U . This cost is higher than (or equal to) z_{comp}^L , but the degrees of importance and satisfaction associated with this solution are also higher (or equal to) than those of the other solution.

By presenting these two options, we offer the decision maker a trade-off between cost and the levels of importance and satisfaction. The decision maker can then choose the solution that best balances these competing objectives based on their preferences. This approach ensures that even when the ideal cost z^* is infeasible, we can still provide viable and meaningful alternatives.

For notational simplicity, let z_{comp} represent the cost value selected by the decision maker from the alternatives z_{comp}^L or z_{comp}^U . While this solution is preferred to the other, it is possible that the decision maker is still not fully satisfied with the cost z_{comp} rather than the other solutions. Therefore, we need a mechanism to iteratively refine the solution and achieve a level of relative satisfaction.

The core idea is to explore trade-offs between cost and the degrees of satisfaction and importance. We aim to reduce the overall cost $z(sd, id)$ by strategically decreasing the selected satisfaction and/or importance levels. To guide this process, we assess the potential cost reduction achievable by adding another element to the satisfaction or importance sets ID' and SD' .

Specifically, we first evaluate the potential cost reduction if an additional element were included in the satisfaction set SD' . This is the maximum value of $SD \setminus SD'$ which also serves as the maximum lower bound for SD' . We perform a similar evaluation for the importance set ID' . This analysis informs the question we pose to the decision maker:

“To improve (reduce) the cost by $[X]$ units, it is necessary to reduce the satisfaction level by $[Y]$ units and/or the importance level by $[Z]$ units. You can choose to reduce either the satisfaction level, the importance level, or both. Which option do you choose?”

in which

- $[X]$ represents the calculated potential cost reduction.
- $[Y]$ represents the corresponding decrease in the satisfaction level required.
- $[Z]$ represents the corresponding decrease in the importance level required.

This question presents the decision maker with concrete trade-offs. They can choose to reduce the satisfaction level, the importance level, or both, depending on their priorities.

If the decision maker selects an option (e.g., reducing the satisfaction level by $[Y]$ units), we implement this change by updating the corresponding sets and recalculating the cost z_{comp} . This process is then repeated, offering the decision maker further opportunities to refine the solution.

If the decision maker is unwilling to reduce either the satisfaction or importance levels (because, for example, they are now at an acceptable level), we consider the current solution as representing the relative satisfaction of the decision maker. The process terminates, and the current value of z_{comp} , along with the associated satisfaction and importance levels, is reported as the final solution. This iterative approach allows us to progressively refine the solution until the decision maker achieves a satisfactory balance between cost, satisfaction, and importance.

We are now ready to formally present our proposed algorithm for solving problem (3). It is detailed in Algorithm 1.

Algorithm 1 Iterative refinement for relative satisfaction

- 1: **Input:** An instance of an assignment problem defined on a fuzzy graph $G(\tilde{V}, \tilde{A})$ with cost vector $\mathbf{c}$.
 - 2: **Output:** Final cost z_{comp} , Final satisfaction set sd_{final} , Final importance set id_{final} .
 - 3: Determine the sets ID and SD defined by (6) and (7), respectively, by solving assignment problems for every pair.
 - 4: Ask the decision maker to specify the minimum values $id_{p'}$ and $sd_{q'}$.
 - 5: Determine the sets ID' and SD' defined by (8).
 - 6: Obtain the value z^* as defined in (9) by solving the assignment problems on $G(V_{sd,id}, A_{sd,id})$.
 - 7: **while** the decision maker is not satisfied **do**
 - 8: Obtain the values $z_{comp}^L, z_{comp}^U \in Range'(z)$, which are close to z^* and contain it.
 - 9: $cost_reduction_sd \leftarrow$ Calculate potential cost reduction by adding a new element to SD' .
 - 10: $cost_reduction_id \leftarrow$ Calculate potential cost reduction by adding a new element to ID' .
 - 11: **if** $cost_reduction_sd > 0$ or $cost_reduction_id > 0$ **then**
 - 12: Set $[X] \leftarrow$ Maximum of $cost_reduction_sd$ and $cost_reduction_id$ (or their combined effect if both are positive).
 - 13: Set $[Y] \leftarrow$ Corresponding decrease in satisfaction level.
 - 14: Set $[Z] \leftarrow$ Corresponding decrease in importance level.
 - 15: Display the following options to the decision maker: options to reduce either the satisfaction level, the importance level, both or none. Set ch to the Decision maker's choice (reduce sd , reduce id , reduce both, none),
 - 16: **if** none **then**
 - 17: Break,
 - 18: **end if**
 - 19: **if** ch involves reducing sd **then**
 - 20: Update SD' by adding the maximum element $sd \in SD \setminus SD'$.
 - 21: **end if**
 - 22: **if** ch involves reducing id **then**
 - 23: Update ID' by adding the maximum element $id \in ID \setminus ID'$.
 - 24: **end if**
 - 25: Recalculate z^* based on the updated satisfaction and importance sets.
 - 26: **end if**
 - 27: **end while**
 - 28: Decision maker is satisfied by $z_{comp} \in \{z_{comp}^L, z_{comp}^U\}$.
 - 29: Set $sd_{final} \leftarrow sd$.
 - 30: Set $id_{final} \leftarrow id$.
-

5 Practical example

This section provides two practical examples to motivate the problem addressed in this paper and to demonstrate the application of the proposed algorithm.

5.1 Assigning teachers to schools

Consider a scenario where a school district's education department is responsible for assigning teachers to various schools within its jurisdiction. This department faces a complex decision-making problem in optimally allocating teachers, balancing multiple, often conflicting objectives.

Each teacher possesses a certain level of “expertise” or “qualification”, which can be quantified through various metrics such as years of experience, advanced degrees, specialized certifications, or student performance evaluations. This expertise represents the teacher’s “importance” to the district, and the department aims to maximize the overall expertise of the teachers assigned to its schools. This ensures students are taught by highly qualified educators.

Furthermore, each school has its own set of criteria for an “ideal” teacher. These criteria might include specific subject matter expertise, experience with particular age groups, familiarity with certain teaching methodologies, or alignment with the school’s educational philosophy. The degree to which a teacher matches these criteria represents the school’s “satisfaction” with that teacher. The education department seeks to maximize the overall satisfaction of its schools with their assigned teachers, ensuring a good fit between teachers and school needs.

Finally, assigning teachers to schools incurs costs. These costs could include salary differences based on experience or qualifications, relocation expenses, or specialized training required for certain assignments. The department aims to minimize the total cost associated with teacher assignments while balancing the other objectives.

The goal is to determine an optimal assignment of teachers to schools that maximizes the overall expertise of the assigned teachers (importance), maximizes the overall satisfaction of the schools with their assigned teachers, and minimizes the total cost of the assignments. This problem can be modeled as an assignment problem on fuzzy graphs, where the education department seeks to find a set of assignments that represents a satisfactory balance between these competing objectives. For example, the department might be willing to accept a slightly higher cost to ensure that schools receive highly qualified teachers who are a good fit for their specific needs. The fuzzy graph approach described in this paper provides a framework for addressing this tri-objective problem and finding such satisfactory solutions.

In the context of the iterative refinement algorithm presented earlier, the superintendent of the school district acts as the high-level decision maker. After the algorithm presents initial solutions, the superintendent is consulted regarding potential trade-offs between cost and the levels of importance and satisfaction. The superintendent’s feedback guides the algorithm in refining the solution towards a compromise that best reflects the district’s priorities.

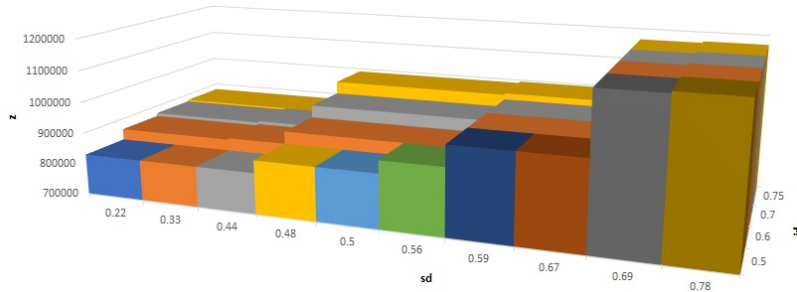


Figure 4: Visualization of $z(sd, id)$ for the numerical example

In this application, we present a numerical example involving four schools, which are designated by the integers 1, 2, 3, and 4, and seven teachers, designated by the letters A, B, C, D, E, F, and G. The relevant data pertaining to this example can be found in Tables 1 and 2. Furthermore, Table 3 illustrates the values of $z(sd, id)$ for every $sd = \mu_{\bar{A}}(i, j)$ and each $id = \mu_{\bar{V}}(i)$. Appendix A contains the Python code which generates the data of Table 3. This code contains a function for evaluating $z(sd, id)$ which is required for the process of Algorithm 1. Additionally, Figure 4 presents a graphical representation of $z(sd, id)$.

It is important to note that the breakpoints of the function $z(., .)$ correspond to points represented as $(id, sd, z(sd, id))$, which are as follows:

Table 1: Teacher popularity and school satisfaction

Teacher	Popularity	School 1	School 2	School 3	School 4
A	0.9	0.78	0.89	0.67	0.56
B	0.6	0.59	0.56	0.48	0.44
C	0.7	0.56	0.67	0.89	0.33
D	0.85	0.67	0.78	0.44	0.89
E	0.8	0.78	0.69	0.56	0.67
F	0.5	0.44	0.33	0.33	0.5
G	0.75	0.33	0.44	0.67	0.22

Table 2: Teacher costs for each school

School Number	Teacher A	Teacher B	Teacher C	Teacher D	Teacher E	Teacher F	Teacher G
1	300000	250000	270000	220000	325000	210000	190000
2	280000	220000	260000	240000	300000	200000	180000
3	320000	210000	275000	250000	220000	230000	210000
4	260000	250000	255000	290000	285000	220000	250000

Table 3: $z(sd, id)$ for different values of id and sd in the numerical example

$sd \backslash id$	0.5	0.6	0.7	0.75
0.22	830000	865000	875000	880000
0.33	830000	865000	875000	880000
0.44	830000	870000	880000	880000
0.48	870000	910000	950000	990000
0.5	870000	910000	950000	990000
0.56	910000	910000	950000	990000
0.59	975000	975000	975000	995000
0.67	975000	975000	975000	995000
0.69	1165000	1165000	1165000	-
0.78	1170000	1170000	1170000	-

$$\{(0.6, 0.78, 1170000), (0.6, 0.67, 975000), (0.6, 0.56, 910000), (0.6, 0.44, 870000), \\ (0.6, 0.33, 865000), (0.6, 0.69, 1165000), (0.7, 0.56, 950000), (0.7, 0.44, 880000), \\ (0.7, 0.33, 875000), (0.5, 0.44, 830000), (0.75, 0.67, 995000), (0.75, 0.56, 990000)\}.$$

Subsequently, we can deduce the following sets:

$$\begin{aligned} ID &= \{0.5, 0.6, 0.7, 0.75\}, \\ SD &= \{0.33, 0.44, 0.56, 0.67, 0.69, 0.78\}, \\ \text{Range}(z) &= \{830000, 865000, 870000, 875000, 880000, 910000, 950000, 975000, \\ &\quad 990000, 995000, 1165000, 1170000\}. \end{aligned}$$

We ask the decision maker to specify the minimum acceptable values for ID and SD . He proposes values of 0.7 for ID and 0.67 for SD . Consequently, we arrive at the following sets:

$$\begin{aligned} ID' &= \{0.7, 0.75\}, \\ SD' &= \{0.67, 0.69, 0.78\}. \end{aligned}$$

To compute the optimal value z^* as defined in equation (9), we have:

$$\begin{aligned} z^* &= \frac{(0.75 - 0.7)(0.69 - 0.67)z(0.7, 0.67) + (0.75 - 0.7)(0.78 - 0.69)z(0.7, 0.69)}{(0.75 - 0.7)(0.78 - 0.67)} \\ &= 113090.9. \end{aligned}$$

In this case, the values $z(0.67, 0.7) = 975000$ and $z(0.69, 0.7) = 1165000$ are determined by solving two assignment problems on their corresponding auxiliary graphs $G(V_{sd,id}, A_{sd,id})$. At this stage, we will recommend adjustments to lower the minimum values of ID and SD that fulfill his requirements. We specify the following observations:

- If the minimum expected value of ID is decreased from 0.7 to 0.6, the cost remains unchanged.
- If he decreases the minimum expectation of SD from 0.67 to 0.56, then the cost decreases from 975000 to 950000.
- If he decreases both the minimum expectations of ID and SD , from 0.7 and 0.67 to 0.7 and 0.56, then the cost further decreases from 975000 to 910000.

If he accepts any of these scenarios, the process will be repeated. Otherwise, we suggest to the decision maker two options: he can choose either the cost of 975000 with $id = 0.7$ and $sd = 0.67$, or the cost of 1165000 with $id = 0.7$ and $sd = 0.69$. The first option corresponds to the assignment:

$$C \rightarrow 2, \quad D \rightarrow 1, \quad E \rightarrow 4, \quad G \rightarrow 3,$$

while the second option corresponds to:

$$A \rightarrow 1, \quad C \rightarrow 3, \quad D \rightarrow 4, \quad E \rightarrow 2.$$

5.2 Emergency response team assignment

To further illustrate the complexity and non-trivial behavior of the proposed interactive algorithm, we expand the second application to an unbalanced, high-dimensional assignment framework. This setup introduces significant combinatorial trade-offs among economic constraints, expert reliability ratings (agent importance), and operational suitability scores (assignment satisfaction).

Consider a large metropolitan Emergency Operations Center (EOC) responding to a widespread disaster zone. The center must deploy exactly five specialized rescue and response teams to cover five distinct critical incidents ($V_2 = \{I_1, I_2, I_3, I_4, I_5\}$) simultaneously. The EOC has a pool of eight available teams ($V_1 = \{A, B, C, D, E, F, G, H\}$), meaning that exactly three teams will remain unassigned on standby. The five critical incidents are categorized as follows:

- Incident I_1 : Structural collapse of a high-rise hospital facility with trapped civilians.
- Incident I_2 : Major hazmat leakage and chemical spill near the main water treatment plant.
- Incident I_3 : Extensive fire propagation across a dense commercial and industrial grid.
- Incident I_4 : Flash flooding in a low-lying residential sector requiring swift-water operations.
- Incident I_5 : Critical failure of the local telecommunication and radar networks.

Senior disaster management experts have evaluated all eight teams to assign a relative importance degree $\mu_{\bar{V}}(i)$, which reflects the team's historical performance, technological readiness, and command stability under stress:

$$\begin{aligned} \mu_{\bar{V}}(A) &= 0.95, & \mu_{\bar{V}}(B) &= 0.90, & \mu_{\bar{V}}(C) &= 0.85, & \mu_{\bar{V}}(D) &= 0.80, \\ \mu_{\bar{V}}(E) &= 0.75, & \mu_{\bar{V}}(F) &= 0.70, & \mu_{\bar{V}}(G) &= 0.65, & \mu_{\bar{V}}(H) &= 0.60. \end{aligned}$$

The expert panel established a bipartite fuzzy graph configuration by determining the conditional satisfaction degrees $\mu_{\bar{A}}(i, j) \in [0, 1]$ for each team–incident pair, obeying the structural property $\mu_{\bar{A}}(i, j) \leq \min\{\mu_{\bar{V}}(i), \mu_{\bar{V}}(j)\}$. These operational suitability values are presented in Table 4. Concurrently, each potential assignment is associated with a crisp operational deployment cost c_{ij} (expressed in thousands of dollars) which accounts for mobilization logistics, distance, and asset depreciation. The cost matrix is detailed in Table 5.

To demonstrate how severely the objectives conflict in this non-trivial setup, we highlight three baseline solutions that optimize each objective independently:

1. **Pure Cost Minimization (cs):** Ignoring the fuzzy parameters, the cheapest plan activates the lowest-tier teams:

$$F_{\text{cost}} = \{(D, I_2), (E, I_3), (F, I_1), (G, I_4), (H, I_5)\}.$$

The total financial outlay is $190 + 150 + 140 + 110 + 95 = 685$ (\$685,000). However, the minimum team importance is severely bottlenecked by Team H ($id = 0.60$), and the satisfaction level drops to $sd = 0.55$ due to Team H at I_5 .

Table 4: Satisfaction degrees ($\mu_{\bar{A}}$) for the expanded 8×5 assignment problem

Team	I_1 (Collapse)	I_2 (Chemical)	I_3 (Fire)	I_4 (Flood)	I_5 (Telecom)
A	0.90	0.45	0.80	0.50	0.75
B	0.40	0.85	0.60	0.70	0.55
C	0.75	0.50	0.80	0.45	0.65
D	0.50	0.75	0.55	0.75	0.40
E	0.65	0.60	0.70	0.55	0.70
F	0.60	0.40	0.55	0.65	0.60
G	0.45	0.55	0.40	0.60	0.50
H	0.50	0.35	0.50	0.40	0.55

Table 5: Operational deployment costs (c_{ij}) in thousands of dollars

Team	I_1 (Collapse)	I_2 (Chemical)	I_3 (Fire)	I_4 (Flood)	I_5 (Telecom)
A	250	310	260	290	270
B	280	240	270	230	290
C	190	210	180	220	200
D	220	190	230	175	240
E	160	180	150	195	170
F	140	165	135	150	160
G	120	140	130	110	125
H	105	115	110	100	95

2. **Maximum Satisfaction Maximization (sd):** Prioritizing operational compliance forces the deployment of top-tier teams:

$$F_{\text{sat}} = \{(A, I_1), (B, I_2), (C, I_3), (D, I_4), (E, I_5)\}.$$

This elite layout maximizes satisfaction at $sd = 0.70$ (with the minimum bounded tightly by $\mu_{\bar{A}}(E, I_5) = 0.70$).

3. **Maximum Strategic Importance (id):** To ensure only the absolute highest-reliability teams are deployed, the EOC maximizes id by selecting five teams out of $\{A, B, C, D, E\}$. Thus, the absolute maximum lower bound for team importance is $id = 0.75$.

We now trace the interactive execution of Algorithm 1 over the expanded breakpoint grid to demonstrate how a compromise is achieved. By sweeping the parameter space via the ϵ -constraint method, the algorithm identifies the distinct parameter breakpoint sets:

$$ID = \{0.60, 0.65, 0.70, 0.75\}, \quad SD = \{0.50, 0.55, 0.60, 0.65, 0.70\}.$$

- **Step 1: Elicitation and Range Restriction.** The EOC director sets an initial strict requirement: a minimum acceptable team importance threshold of $id_{\min} = 0.70$ and a minimum assignment satisfaction score of $sd_{\min} = 0.60$. The algorithm filters the active sub-grids:

$$ID' = \{0.70, 0.75\}, \quad SD' = \{0.60, 0.65, 0.70\}.$$

The search domain is restricted to the box bounds $id \in [0.70, 0.75]$ and $sd \in [0.60, 0.70]$.

- **Step 2: Calculating the Continuous Expected Cost Target (z^*).** Applying the double-integral discretization from Equation (9) across the active steps, the algorithm aggregates the weighted costs over the restricted decision zone:

$$\begin{aligned}
z^* &= \frac{1}{(0.70 - 0.60)(0.75 - 0.70)} \sum_i \sum_j z(sd_{i-1}, id_{j-1}) \Delta sd_i \Delta id_j \\
&= \frac{1}{0.10 \times 0.05} \left[905(0.05)(0.05) + 940(0.05)(0.05) \right] \approx 922.50.
\end{aligned}$$

- **Step 3: Determining Feasible Enclosing Bounds.** Since the calculated continuous target $z^* = 922.50 \notin \text{Range}'(z)$, the algorithm isolates the closest lower and upper valid step costs within the restricted box:

$$\begin{aligned} - z_{\text{comp}}^L &= \max\{z \in \text{Range}'(z) : z \leq 922.50\} = 905, \\ - z_{\text{comp}}^U &= \min\{z \in \text{Range}'(z) : z \geq 922.50\} = 940. \end{aligned}$$

The algorithm establishes the target decision interval $[905, 940]$.

- **Step 4: Interactive Trade-off Presentation (Iteration 1).** The algorithm presents the director with two operational choices:
 - *Solution 1 (Lower Bound):* Total cost of \$905,000, ensuring a minimum satisfaction level of $sd = 0.60$ and a minimum team reliability rating of $id = 0.70$.
 - *Solution 2 (Upper Bound):* Total cost of \$940,000, ensuring an elevated satisfaction level of $sd = 0.65$ and a minimum team reliability rating of $id = 0.70$.

The director selects Solution 1 ($z_{\text{comp}} = 905, sd = 0.60, id = 0.70$) but requests a lower budget if possible. The algorithm initiates the relaxation bargaining logic.

- **Step 5: Relaxation Analysis and Bargaining.** The algorithm scans adjacent cells outside the restricted boundaries to determine the exact marginal gains of dropping thresholds:
 - *Relaxing Satisfaction (sd):* Lowering the satisfaction floor from 0.60 to 0.55 does not yield a cheaper feasible assignment while maintaining the current importance threshold ($id = 0.70$); the minimum achievable cost remains at \$905,000.
 - *Relaxing Importance (id):* Lowering the importance floor from 0.70 to 0.65 permits the inclusion of lower-tier assets (Teams F and G). The deployment cost drops to \$790,000, saving \$115,000.

The system prompts the decision-maker with the following trade-off query:

“Reducing your operational cost is not possible by lowering the satisfaction threshold alone. However, to achieve major savings of \$115,000, you must accept lower-tier teams on the field, reducing the reliability threshold by 0.05 units (from 0.70 to 0.65). Do you accept?”

- **Step 6: Convergence to Final Compromise Solution.** The EOC director determines that allowing the reliability rating to fall to 0.65 introduces considerable operational risk for core disaster response. Nevertheless, given the tight budget constraints, they agree to this relaxation. The selection is saved, updating the parameters to $sd_{\text{final}} = 0.60$ and $id_{\text{final}} = 0.65$. The final matching layout is:

$$F^* = \{(C, I_3), (D, I_2), (E, I_5), (F, I_1), (G, I_4)\},$$

which has a total cost of \$790,000 with $sd = 0.60$ and $id = 0.65$. This assignment is returned as the final optimal compromise.

6 Conclusion

This paper addressed the assignment problem on fuzzy graphs, incorporating expert knowledge through membership values representing the importance of agents/tasks and satisfaction with assignments. The problem was formulated as a tri-objective optimization model and solved using an interactive algorithm based on the ϵ -constraint method. By leveraging an auxiliary graph, we reduced the problem to a series of classical assignment problems, enabling efficient solution. Practical examples demonstrated the applicability and effectiveness of the proposed approach.

This work lays the foundation for further research in several directions. While this study focused on fuzzy graphs, the proposed methodology can be extended to more expressive fuzzy graph models. Exploring extensions of fuzzy graphs, such as intuitionistic fuzzy graphs [23] and hesitant fuzzy graphs [12], presents a promising avenue for future work. Intuitionistic fuzzy graphs, with their membership and non-membership degrees, could capture a more nuanced representation of importance and satisfaction, allowing for the modeling of hesitancy or uncertainty in expert opinions. Similarly, hesitant fuzzy graphs, which allow for a set of possible membership values, could be employed to handle situations where the expert’s preferences are not precisely defined. Furthermore, investigating alternative methods for combining the multiple objectives, beyond the mean value theorem, could lead to potentially improved compromise solutions.

Declarations

Data availability: No data.

Compliance with Ethical Standards:

- The authors declare that they have no conflict of interest.
- This paper does not contain any studies with human participants or animals performed by any of the authors.
- Informed consent was obtained from all individual participants included in the study.

Declaration of Generative AI and AI-Assisted Technologies:

- The authors declare that no generative AI or AI-assisted technologies were used in the preparation of this manuscript.

Acknowledgement

During the preparation of this manuscript, the authors used DeepSeek solely for language editing and improving clarity. The authors reviewed and verified all AI-assisted output and took full responsibility for the accuracy, originality, validity, integrity, and scientific content of the manuscript. No generative AI or AI-assisted technology was used to generate, modify, fabricate, manipulate, or interpret the primary data, scientific results, statistical analyses, or conclusions reported in this manuscript.

References

- [1] R. K. Ahuja, T. L. Magnanti, J. B. Orlin, *Network flows: Theory, applications and algorithms*, Prentice-Hall, 1993.
- [2] M. S. Bazaraa, J. J. Jarvis, H. D. Sherali, *Linear programming and network flows*, John Wiley and Sons, 2009. <https://doi.org/10.1002/9780471703778>
- [3] I. Belik, *The analysis of split graphs in social networks based on the k-cardinality assignment problem*, International Journal of Network Science, **1**(1) (2016), 53-62. <https://doi.org/10.1504/IJNS.2016.073557>
- [4] R. Bihari, Jeevaraj S, A. Kumar, *A new simplex algorithm for interval-valued Fermatean fuzzy linear programming problems*, Computational and Applied Mathematics, **44**(1) (2025). <https://doi.org/10.1007/s40314-024-02949-3>
- [5] N. R. Devi, S. Sowmiya, Y. Parthiban, *Solving of assignment problem by Pythagorean octagonal neutrosophic fuzzy number*, International Journal of Information Technology, **17**(2) (2025), 1079-1085. <https://doi.org/10.1007/s41870-024-02243-9>
- [6] G. Devipriya, M. J. Indrani, *Solution of fuzzy assignment problem Hungarian algorithm and using branch and bound method*, International Journal of Current Research and Techniques (IJCRT), **15**(2) (2025), 50856-50865. <https://doi.org/10.5281/zenodo.15639927>
- [7] A. Ebrahimnejad, J. L. Verdegay, *A new approach for solving fully intuitionistic fuzzy transportation problems*, Fuzzy Optimization and Decision Making, **17**(4) (2018), 447-474. <https://doi.org/10.1007/s10700-017-9280-1>
- [8] M. Ehrgott, *Multicriteria optimization*, **491**, Springer Science and Business Media, 2005. <https://doi.org/10.1007/3-540-27659-9>
- [9] A. Emrouznejad, L. M. Z. Angiz, W. Ho, *An alternative formulation for the fuzzy assignment problem*, Journal of the Operational Research Society, **63** (2012), 59-63. <https://doi.org/10.1057/jors.2011.14>
- [10] S. Faudzi, S. Abdul-Rahman, R. Abd Rahman, *An assignment problem and its application in education domain: A review and potential path*, Advances in Operations Research, **2018**(1) (2018), 8958393. <https://doi.org/10.1155/2018/8958393>
- [11] A. N. Gani, V. N. Mohamed, *Solution of a fuzzy assignment problem by using a new ranking method*, International Journal of Fuzzy Mathematical Archive, **2** (2013), 8-16. <http://www.researchmathsci.org/ijfmaart/ijfma-v2-2.pdf>

- [12] Z. Gong, J. Wang, *Hesitant fuzzy graphs, hesitant fuzzy hypergraphs and fuzzy graph decisions*, Journal of Intelligent and Fuzzy Systems, **40**(1) (2021), 865-875. <https://doi.org/10.3233/JIFS-201016>
- [13] D. Gurukumaresan, C. Duraisamy, R. Srinivasan, V. Vijayan, *Optimal solution of fuzzy assignment problem with centroid methods*, Materials Today: Proceedings, **37**(Part 2) (2021), 553-555. <https://doi.org/10.1016/j.matpr.2020.05.582>
- [14] H. Huang, S. Gao, *Trajectory-adaptive routing in dynamic networks with dependent random link travel times*, Transportation Science, **52**(1) (2018), 102-117. <https://doi.org/10.1287/trsc.2016.0691>
- [15] B. Jennings, R. Stadler, *Resource management in clouds: Survey and research challenges*, Journal of Network and Systems Management, **23**(3) (2015), 567-619. <https://doi.org/10.1007/s10922-014-9307-7>
- [16] P. S. Kumar, *Intuitionistic fuzzy solid assignment problems: A software-based approach*, International Journal of System Assurance Engineering and Management, **10**(4) (2019), 661-675. <https://doi.org/10.1007/s13198-019-00794-w>
- [17] P. S. Kumar, R. J. Hussain, *A method for solving balanced intuitionistic fuzzy assignment problem*, International Journal of Engineering Research and Applications, **4**(3) (2014), 897-903. <file:///C:/Users/Mona/Downloads/P.SENTHILKUMAR.pdf>
- [18] H. Lane, M. Sarkies, J. Martin, T. Haines, *Equity in healthcare resource allocation decision making: A systematic review*, Social Science and Medicine, **175** (2017), 11-27. <https://doi.org/10.1016/j.socscimed.2016.12.012>
- [19] C. J. Lin, U. P. Wen, *A labeling algorithm for the fuzzy assignment problem*, Fuzzy Sets and Systems, **142**(3) (2004), 373-391. [https://doi.org/10.1016/S0165-0114\(03\)00017-4](https://doi.org/10.1016/S0165-0114(03)00017-4)
- [20] M. K. Mantrala, *Allocating marketing resources*, In *Handbook of Marketing*, (2002), 409-435.
- [21] D. V. K. Nasir, Priyanka U, *A multi-expert, multi-criteria approach to the neutrosophic fuzzy assignment problem*, Neutrosophic Sets and Systems, **90**(1) (2025), 54.
- [22] T. Öncan, *A survey of the generalized assignment problem and its applications*, INFOR: Information Systems and Operational Research, **45**(3) (2007), 123-141. <https://doi.org/10.3138/infor.45.3.123>
- [23] R. Parvathi, M. G. Karunambigai, *Intuitionistic fuzzy graphs*, Computational Intelligence, Theory and Applications: International Conference 9th Fuzzy Days in Dortmund, Germany, Sept. 18-20, 2006 Proceedings, Springer Berlin Heidelberg, (2006), 139-150. https://doi.org/10.1007/3-540-34783-6_15
- [24] A. C. Ribeiro, A. Sizo, L. P. Reis, *Investigating the reviewer assignment problem: A systematic literature review*, Journal of Information Science, **52**(1) (2026), 39-59. <https://doi.org/10.1177/01655515231176668>
- [25] M. Sakawa, *Fuzzy multiobjective optimization*, In *Multicriteria Decision Aid and Artificial Intelligence: Links, Theory and Applications*, (2013), 235-271. <https://doi.org/10.1002/9781118522516.ch10>
- [26] R. Sanjana, G. Ramesh, *Optimization techniques of assignment problem using trapezoidal intuitionistic fuzzy numbers and interval arithmetic*, Statistics, Optimization and Information Computing, **12**(6) (2024), 1984-1999. <https://doi.org/10.19139/soic-2310-5070-1842>
- [27] S. Srinivas, K. Prabakaran, *A novel approach for solving the triangular fuzzy neutrosophic assignment problem*, Yugoslav Journal of Operations Research, **35**(2) (2025), 437-466. <https://doi.org/10.2298/YJOR240316025S>
- [28] M. Wen, K. Iwamura, *Fuzzy facility location-allocation problem under the Hurwicz criterion*, European Journal of Operational Research, **184**(2) (2008), 627-635. <https://doi.org/10.1016/j.ejor.2006.11.029>

A Python code to generate the data of Table 3

This appendix provides the complete Python code used to generate the numerical results presented in Table 3. The code employs the Gurobi optimization library to solve a series of assignment problems on the auxiliary graph $G_{sd,id}$.

The code iterates over all distinct values of id (taken from $\mu_{\tilde{V}}(i)$) and sd (taken from $\mu_{\tilde{A}}(i, j)$). For each pair (sd, id) , the function `compute_z(sd, id)` constructs a Gurobi model of the assignment problem on the auxiliary graph $G_{sd,id}$. In this model, a binary decision variable x_{ij} is created for each teacher–school pair. If the conditions $\mu_{\tilde{V}}(i) \geq id$ and $\mu_{\tilde{A}}(i, j) \geq sd$ are satisfied, the variable is assigned the actual cost c_{ij} ; otherwise, it is assigned a large penalty M to discourage its selection. The model enforces that each teacher is assigned to at most one school and that each school receives exactly one teacher. If a feasible assignment exists, the optimal cost is recorded; otherwise, the value is marked as infeasible (denoted by ‘–’ in Table 3). The results are then aggregated into Table 3, which forms the basis for the breakpoint analysis.

```
import gurobipy as gr

# -----
# Parameters
# -----
M = 1000000 # Large constant to penalize infeasible assignments

# Lists of schools and teachers
schools = [1, 2, 3, 4]
teachers = ['A', 'B', 'C', 'D', 'E', 'F', 'G']

# Teacher importance (membership values for nodes in V1)
mu_V = {
    'A': 0.9,
    'B': 0.6,
    'C': 0.7,
    'D': 0.85,
    'E': 0.8,
    'F': 0.5,
    'G': 0.75
}

# Satisfaction degrees for each teacher-school pair (membership values for arcs)
mu_A = {
    ('A', 1): 0.78, ('A', 2): 0.89, ('A', 3): 0.67, ('A', 4): 0.56,
    ('B', 1): 0.59, ('B', 2): 0.56, ('B', 3): 0.48, ('B', 4): 0.44,
    ('C', 1): 0.56, ('C', 2): 0.67, ('C', 3): 0.89, ('C', 4): 0.33,
    ('D', 1): 0.67, ('D', 2): 0.78, ('D', 3): 0.44, ('D', 4): 0.89,
    ('E', 1): 0.78, ('E', 2): 0.69, ('E', 3): 0.56, ('E', 4): 0.67,
    ('F', 1): 0.44, ('F', 2): 0.33, ('F', 3): 0.33, ('F', 4): 0.50,
    ('G', 1): 0.33, ('G', 2): 0.44, ('G', 3): 0.67, ('G', 4): 0.22
}

# Assignment costs for each teacher-school pair
cost = {
    ('A', 1): 300000, ('A', 2): 280000, ('A', 3): 320000, ('A', 4): 260000,
    ('B', 1): 250000, ('B', 2): 220000, ('B', 3): 210000, ('B', 4): 250000,
    ('C', 1): 270000, ('C', 2): 260000, ('C', 3): 275000, ('C', 4): 255000,
    ('D', 1): 220000, ('D', 2): 240000, ('D', 3): 250000, ('D', 4): 290000,
    ('E', 1): 325000, ('E', 2): 300000, ('E', 3): 220000, ('E', 4): 285000,
    ('F', 1): 210000, ('F', 2): 200000, ('F', 3): 230000, ('F', 4): 220000,
    ('G', 1): 190000, ('G', 2): 180000, ('G', 3): 210000, ('G', 4): 250000
}

# Distinct values of id and sd
L_V = set(mu_V.values())
L_A = set(mu_A.values())

# -----
# Function to compute z(sd, id)
# -----
def compute_z(sd, id_val):
    """
    Solves the assignment problem on the auxiliary graph G_{sd,id}.
    Returns the optimal cost if a feasible assignment exists; otherwise,
    returns a value >= M (indicating infeasibility).
    """
```

```

model = gr.Model()
model.setParam('OutputFlag', False) # Suppress Gurobi output

x = {}

# Create decision variables
for teacher in teachers:
    for school in schools:
        # Check if the arc satisfies the importance and satisfaction thresholds
        if mu_V[teacher] >= id_val and mu_A[(teacher, school)] >= sd:
            # Feasible arc: assign the actual cost
            x[(teacher, school)] = model.addVar(obj=cost[(teacher, school)], vtype='B')
        else:
            # Infeasible arc: penalize with a large cost M
            x[(teacher, school)] = model.addVar(obj=M, vtype='B')

# Constraint: each teacher is assigned to at most one school
for teacher in teachers:
    model.addConstr(gr.quicksum(x[(teacher, school)] for school in schools) <= 1)

# Constraint: each school receives exactly one teacher
for school in schools:
    model.addConstr(gr.quicksum(x[(teacher, school)] for teacher in teachers) == 1)

# Solve the model
model.optimize()

# Return the optimal objective value
return model.ObjVal

# -----
# Compute breakpoints
# -----
breakpoints = {}
for id_val in L_V:
    for sd in L_A:
        zz = compute_z(sd, id_val)
        if zz < M: # Only store feasible solutions
            breakpoints[(id_val, sd)] = zz

# Display the results (optional)
for (id_val, sd), zz in sorted(breakpoints.items()):
    print(f"id = {id_val}, sd = {sd} -> z = {zz}")

```