*Vietnam Journal of Science and Technology* ***62*** *(1) (2024) 123-139*

doi:10.15625/2525-2518/18222

# Some new fuzzy query processing methods based on similarity measurement and fuzzy data clustering

**Nguyen Tan Thuan[1, *], Tran Thi Thuy Trinh[2], Doan Van Ban[3], Truong Ngoc Chau[4], Nguyen Thi Anh Phuong[3], Nguyen Truong Thang[3, *]**

[1]*The University of Danang - University of Technology and Education, 48 Cao Thang, Hai Chau, Da Nang, Viet Nam*

[2]*Duy Tan University, 254 Nguyen Van Linh, Thanh Khe, Da Nang, Viet Nam*

[3]*Institute of Information Technology, Vietnam Academy of Science and Technology, 18 Hoang Quoc Viet, Cau Giay, Ha Noi, Viet Nam*

[4]*The University of Danang - University of Science and Technology, 54 Nguyen Luong Bang, Lien Chieu, Da Nang, Viet Nam*

[*]Emails: *ntthang@ioit.ac.vn,thuannt.it.dtu@gmail.com*

Received: 6 April 2023; Accepted for publication: 15 December 2023

**Abstract.** In relational and object-oriented database systems, fuzzy or uncertain data will always be naturally present. However, these systems have many limitations for handling complex data types of fuzzy nature. Therefore, to represent and process fuzzy data, they need a flexible and efficient fuzzy query system. To solve this challenge, this paper proposes two different approaches to increase the flexibility of the fuzzy interrogation system. For the first and foremost approach, based on similarity measures and fuzzy logic, we develop three fuzzy query processing algorithms for single-condition and multi-condition cases such as FQSIMSC (Fuzzy Query Simulation Single Condition), FQSIMMC (Fuzzy Query Simulation Multiple Condition) based on the proposal of Y. Bashon [27] and FQSEM (Fuzzy Query SEM) based on the proposal of Z. M. Ma [28]. As for the second approach, we combine the fuzzy clustering algorithm EMC (Expectation Maximization Coefficient) and the query processing algorithm based on fuzzy partitioning FQINTERVAL (Fuzzy Query Interval) based on the proposal of T. T. Nguyen [26]. With this approach, we not only improve query processing cost but also support applications and devices equipped with intelligent interactive function that easily interacts with the fuzzy query system. Finally, we evaluate the algorithms against datasets of different sizes (extracted from UCI) and the method of matching the optimal possibilities in terms of processing time and memory space.

*Keywords:* Fuzzy oriented object database, Fuzzy query processing, Similarity measurement, Fuzzy intervals, Expectation maximization coefficient.

*Classification numbers*: 4.8.4, 5.8.1, 5.8.2, 5.8.3

## 1. INTRODUCTION

Information systems have revolutionized the way complex and diverse information is stored and processed. As a result, the volume of information has increased significantly leading

to information overload. Consequently, it becomes difficult to analyze the large amount of available data and make appropriate management decisions. In practice, information systems mainly use relational databases [1 - 2] or object-oriented databases (OODB) [3 - 5], to store these datasets. Both relational and object-oriented database models are capable enough of handling complex objects but are limited to inaccurate or uncertain data representations. Another problem is that the use of object-oriented and relational models has many limitations in describing and handling uncertain and incomplete information. Accordingly, a query process is not suitable for decision making. In addition, these systems can only deal with "hard" (precise and deterministic) data in the wild. However, many real-world applications always involve "soft" (vague and imprecise) data.

Besides, online consulting services have also appeared on web applications through chatbot automated consulting tools [6 - 7] by applying artificial intelligence and cloud data to provide information to customers.

Furthermore, robots can communicate with humans using natural language [8]. Data preprocessing is a very important step in data transformation and fuzzification to facilitate interrogation for non-expert users.

Along with the development of fuzzy math such as probability theory, fuzzy set theory, and similarity relationship [9 - 12], there are fuzzy object-oriented and relational database models proposed by M. Umano *et al*. [13], G. Bordogna *et al*. [14], and Caluwe [15] as well as model of probability proposed by B. Ding *et al*. [16]

When dealing with complex data stored as objects that may contain inaccuracies and uncertainties, fuzzy query processing proposals can be used. One such proposal, based on fuzzy set theory, employs a membership function to represent quantitative values as linguistic values. To improve query execution efficiency, a technique called horizontal fragmentation [18] is used to reduce the number of hits in fragmentation, and an efficient query execution method can be selected. Another approach combines the MapReduce computational model with the kdStreamSky fuzzy clustering technique to create a skyline query processing method, which is introduced in ref. [19]. A fuzzy query processing architecture for relational databases was proposed in ref. [20].

The motivation of studying the FOODB model and query processing to solve the limitations of relational databases, crisp OODB, and FOODB for the treatment of uncertain and incomplete information becomes the subject of this paper. The contributions of this paper focus on two main issues:

- Data preprocessing: Proposing a technique to evaluate the overall similarity of two objects, based on Minskowski and Euclidean distance measures.
- Query processing: Proposing four new fuzzy query processing algorithms, namely FQSIMSC, FQSIMMC, FQSEM, and FQINTERVAL. Among them, three algorithms FQSIMSC, FQSIMMC, FQSEM use similarity measures based on calculations SIM, SEM. The fourth algorithm FQINTERVAL processes queries directly on fuzzy intervals based on the EMC clustering algorithm.

The paper is structured as follows. Part 2 compares two objects based on ambiguous similarity measure and data semantics, part 3 presents fuzzy query processing based on similarity measure, clustering algorithm and separation of fuzzy intervals. The experiment and conclusion are finally stated in parts 4 and 5.

## 2. PRELIMINARY

### 2.1. Measurement of ambiguous data semantics

The semantic space of the ambiguous data type is represented by the ability distribution. The descriptions of the semantic space are represented through semantic relationships. The measure of semantic inclusion and semantic equivalence is usually applied to the degree of semantic inclusion [21 - 22]. Given the universe set $U = \{u_1, u_2, \dots, u_n\}$, with two fuzzy data $\pi_A$ and $\pi_B$ defined on the domain U based on the probability of distribution and $\pi_A(u_i)$, $u_i \in U$, it proves the possibility that $u_i$ is true. The symbol $SID(\pi_A, \pi_B)$ is defined as follows:

$$SID(\pi_A, \pi_B) = \sum_{i=1}^{n} \min_{u_i \in U}(\pi_B(u_i), \pi_A(u_i)) \Big/ \sum_{i=1}^{n} \pi_B(u_i) \quad (1)$$

According to the above definition, the concepts of similarity can be inferred as follows. Let $\pi_A$ and $\pi_B$ be two fuzzy data and $SID(\pi_A, \pi_B)$ is the degree to which $\pi_A$ covers the semantics of $\pi_B$. $SE(\pi_A, \pi_B)$ is defined as follows:

$$SE(\pi_A, \pi_B) = \min\big(SID(\pi_A, \pi_B), SID(\pi_B, \pi_A)\big). (2)$$

### 2.2. Comparison of two objects based on ambiguous similarity measure [27]

**Example 2.1.** Comparing the similarity between two fuzzy objects. An employee wants to rent an apartment and wants to compare apartments to choose the most fit one. Each apartment is defined by area, price and distance to work. Assume that two apartments have been found as shown in Figure 1, and "How can these two apartments be compared?".

| Aparment 1 | Aparment 2 |
|---|---|
| Area: Large;<br>Rental price: $840;<br>VAST: Far | Area: Medium;<br>Rental price: Dearness;<br>VAST : 1km |

*Figure 1.* A good example of an ambiguous object comparison.

The description of the two apartments as shown in Figure 1 is ambiguous, since the values of attributes are of mixed form, i.e. numeric values and language values [23]. In other words, Apartment 1 and Apartment 2 are ambiguous objects of the fuzzy apartment class (each ambiguous object has at least one value of the ambiguous property).

The following methods help us calculate the similarity between two ambiguous objects: compare two ambiguous properties, compare a crisp attribute with an ambiguous attribute and vice versa, compare two objects with the same instance of a class, and compare two objects that are instances of two different classes.

#### *2.2.1. Compare two fuzzy attributes*

In this section, to solve case I, we compare objects with fuzzy properties. Initially, determine the similarity of two fuzzy objects through the ambiguity property and then compute the general similarity of the two ambiguous objects using formulas (9) and (10).

Let two objects be $o_1$ and $o_2$, the corresponding sets of attribute sets are as follows:

$$atr_{O_1} = \{at_1, at_1, \dots, at_n\} \text{ and } atr_{O_2} = \{bt_1, bt_1, \dots, bt_n\}.$$

The similarity $S$: $at_{O_1} \times at_{O_2} \rightarrow [0; 1]$ between two attributes corresponding to $at_j, bt_j$ is defined as follows:

$$S(at_j, bt_j) = \frac{1-d(at_j, bt_j)}{1+k_j d(at_j, bt_j)};\ where\ k_j \geq 0, (3)$$

where j, $a_{tj}\ and\ bt_j$ are the $j^{th}$ attribute with j = 1, 2,…, n, with n being the numeral of attributes and the distance measure *d* is denoted by representing $\oplus_j$: $[0; 1]^{m_j} \rightarrow [0; 1]$ as follows:

$$d(at_j, bt_j) = \oplus_j \left(dis(A_{1j}, B_{1j}), dis(A_{2j}, B_{2j}), \dots, dis\left(A_{m_j j}, B_{m_j j}\right)\right), (4)$$

where $A_{m_j j}, B_{m_j j}$ the corresponding attribute values of $at_j\ and\ bt_j$ with $m_j$ is being the representative fuzzy subset for the value of the $j^{th}$ attribute belonging to the domain $U_j$. $\oplus_j$ is represented as follows:

$$d(at_j, bt_j) = \left[\frac{\sum_{i=1}^{m_j} dis(A_{ij}, B_{ij})^2}{m_j}\right]^{1/2} (5)$$

The distance $dis$: $F(U_j) \times F(U_j) \rightarrow [0; 1]$. The difference of fuzzy sets can be determined as follows: If the properties $at_j\ and\ bt_j$ are linguistic values and their semantics are determined using the member function $\mu_{A_{ij}}(x) = \mu_{B_{ij}}(x)$ with every $x \in U_j$, to compare two apartments (see Figure 3 in example 1), then:

$$dis(A_{ij}, B_{ij}) = \left|\mu_{A_{ij}}(x) - \mu_{A_{ij}}(y)\right|; \text{ for any } x, y \in U_j. (6)$$

The similar definition proposed in equation (3) allows us to evaluate the degree of similarity between the properties of two instances. In equation (3), the distance d is adjusted for match based on the parameter $k_j$. The similarity measure $Sim(o_1, o_2)$ between two fuzzy objects $o_1$and$o_2$ is:

$$Sim(o_1, o_2) = \oplus \left(S(at_1, bt_1), S(at_2, bt_2), \dots, S(at_n, bt_n)\right), (7)$$

where the mapping $\oplus_j$: $[0,1]^n \rightarrow [0,1]$ is an aggregate operation such as minimum function and weighted mean:

1. Average weight of similarity points of the attributes:

$$\oplus \left(S(at_1, bt_1), S(at_2, bt_2), \dots, S(at_n, bt_n)\right) = \frac{\sum_{j=1}^{n_j} \alpha_j S(at_j,\ bt_j)}{\sum_{j=1}^{n} \alpha_j}; \alpha_j \in [0,1] (8)$$

2. The minimum similarities of the attributes are:

$$\oplus \left(S(at_1, bt_1), S(at_2, bt_2), \dots, S(at_n, bt_n)\right) min[S(at_1, bt_1), S(at_2, bt_2), \dots, S(at_n, bt_n)] (9)$$

Using the membership function shown in Figure 3, the similarity calculation between these attributes can be measured by:

$$d(at_j, bt_j) = \left[\frac{\sum_{i=1}^{m_j} \left|\mu_{A_{ij}}(x) - \mu_{A_{ij}}(y)\right|^2}{m_j}\right]^{1/2}; x, y \in D_Q\ (10)$$

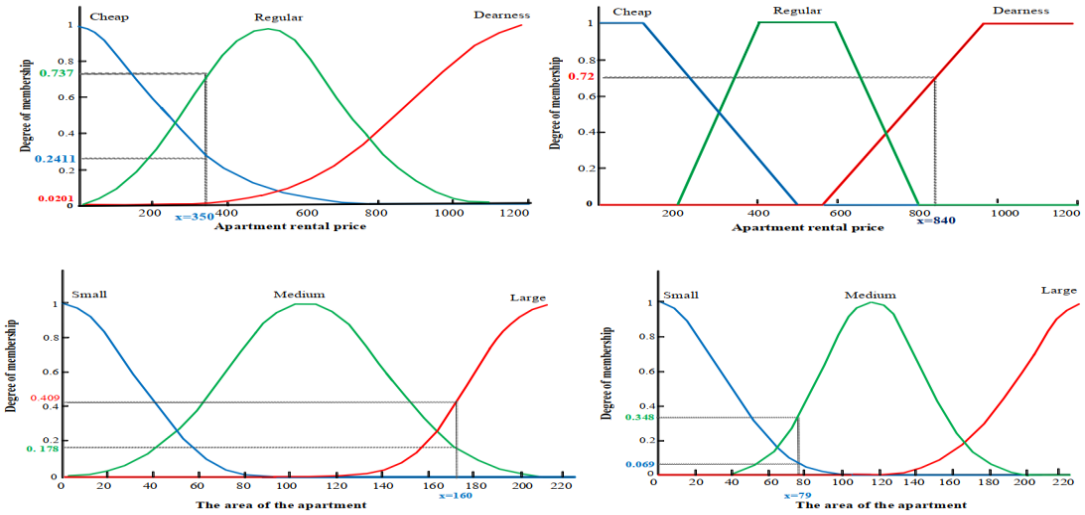


*Figure 2.* Fuzzy representation of area and price of two apartments.

Due to the characteristics of the object-oriented database model and the density of data distribution of this model, applying two bell and trapezoidal membership functions to represent quantitative values in qualitative form is the most effective. Moreover, the calculation and data processing on the two bell and trapezoid member functions is more efficient than the other membership functions due to the symmetry of the bell member function and the common ease of use of the trapezoid. Therefore, choosing these two types of membership functions is suitable as a theoretical basis for analyzing this problem.

*2.2.2. Compare the similarity of two objects of the same class*

Let class C has the following properties: $\{at_1, at_2, \dots, at_n\}$ and two objects $o_1$and $o_2$ are of class C with the same set of attributes. Assuming that $o_1(A_i)$ and $o_2(A_i)$ are ambiguous values, to compare two ambiguous instances $o_1$ and $o_2$ and compute $\mu(o_1, o_2)$, we first compare their respective attributes. For each pair of values of the same attribute $(sayA_i(1 \le i \le n))$ we need to calculate their degree of equivalence, represented by $\mu_{Ai}(o_1, o_2)(0 \le \mu_{Ai}(o_1, o_2) \le 1)$. Here, $\mu_{Ai}(o_1, o_2) = SE(o_1(A_i), o_2(A_i))$. Based on the importance of each attribute, a weight $w_i$ is assigned to each attribute such that $0 \le w_i \le 1\ and \sum w_i = 1,$ with $i = 1, 2, \dots, n$. Formally, the similarity of $o_1$ and $o_2$, calculated by $\mu_{Ai}(o_1, o_2)$, is expressed as follows: $\mu(o_1, o_2) = \sum(\mu_{Ai}(o_1, o_2)) \times w_i)$, with $i = 1, 2, \dots, n$. [28]

*2.2.3. Compare the similarity of two fuzzy objects that are not of the same class*

Let $C_1$ be the fuzzy class having attributes $\{at_1, at_2, \dots, at_k, at_{k+1}, \dots, at_m\}$ and $C_2$ be the class with attributes $\{at_1, at_2, \dots, at_k, at'_{k+1}, \dots, at'_m, at_{m+1}, \dots, at_n\}$. Here the attributes $at'_{k+1}$, ..., and $at'_m$ are overwritten from $at_{k+1}, \dots, and at_m$, where $at_{m+1}, \dots, and at_n$ are special attributes. More specifically, $C_1$ is the superclass of $C_2$ or $C_2$ is the subclass of $C_1$. So, let $o_1$ and $o_2$ be two fuzzy objects corresponding to two fuzzy classes $C_1$ and $C_2$. To compare two ambiguous objects $o_1$of class $C_1$, $o_2$ of class $C_1$ and compute $\mu(o_1, o_2)$, in general, the weighting of the attribute is considered by evaluating the similarity of each pair of attributes. The similarity of each pair of attributes is expressed as $\mu_{Ai}(o_1, o_2)$, with $0 \le \mu_{Ai}(o_1, o_2) \le 1$. For each value

pair of the attribute and its overriding attribute $(i.e.\ A_j v\grave{a}\ A'_j,\ \text{with}\ k+1 \le j \le m)$, we obtain the degree of equivalence, which is $\mu_{Aj}(o_1, o_2)(0 \le \mu_{Aj}(o_1, o_2) \le 1)$. Here $\mu_{Aj}(o_1, o_2) = SE\left(o_1(A_j), o_2(A'_j)\right)$.

Finally, we have [28]:

$$\mu\,(o_1, o_2) = \sum\left(\mu_{Ai}(o_1, o_2) \times w_i\right) + \sum\left(\mu_{Aj}(o_1, o_2)) \times w_j\right) (11)$$

where i = 1,2, …, k; j = k + 1, k + 1,…,m.

## 3. FUZZY QUERY PROCESSING BASED ON SIMILARITY MEASURE, CLUSTERING ALGORITHM AND SEPARATION OF FUZZY INTERVALS

### 3.1. The framework of proposed methods

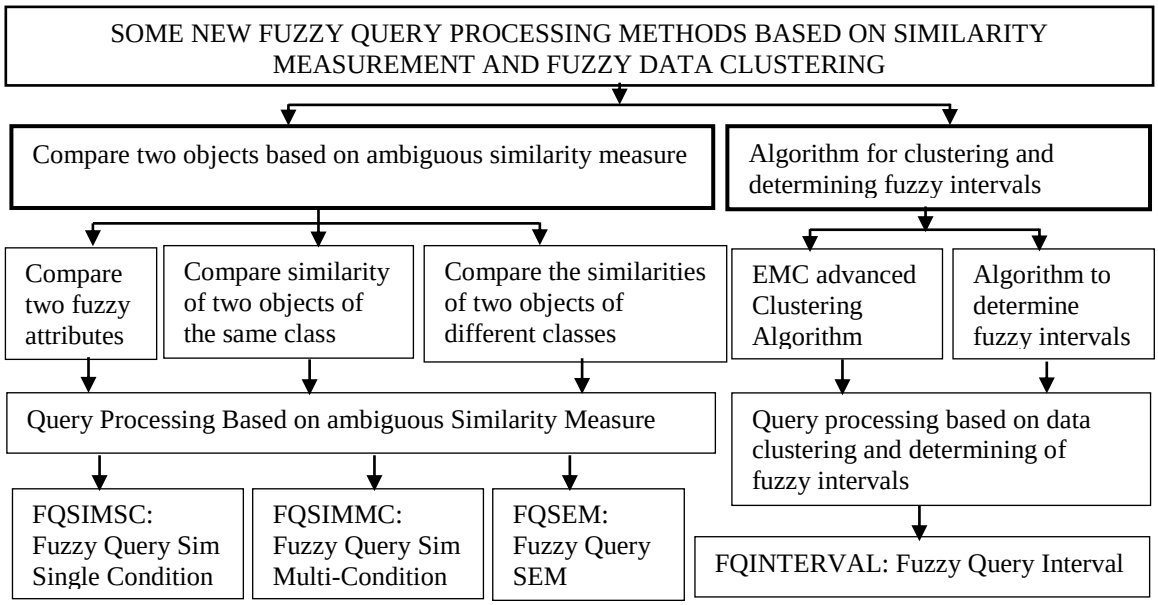


*Figure 3.* The framework of proposed methods for fuzzy query processing.

We propose fuzzy query processing methods for the two cases described in Figure 3. For case 1, we compare two objects based on similarity measures such as Compare two fuzzy attributes, Compare the similarity of two objects of the same class and different classes to develop three algorithms for fuzzy query processing such as: FQSIMSC, FQSIMMC, FQSEM. For case 2, we implement a fuzzy interval division algorithm based on the results of the improved clustering algorithm EMC. From the results obtained, we develop a clustered fuzzy query processing algorithm named FQINTERVAL.

### 3.2. Query Processing Based on ambiguous Similarity Measure

Based on the fuzzy class model and fuzzy graph, we build a FOQL fuzzy query processing structure with the following general form.

SELECT < attribute list >FROM <$Class_1$WITH threshold$_1$;..... ;$Class_m$ WITH threshold$_1$> WHERE < Query condition > THOLD threshold

Where, <Query condition> is a fuzzy condition and all thresholds are sequences of numbers in [0;1]. By using such FOQL, one can extract these objects that belong to the subthreshold class, while satisfying the query condition to be below the threshold. Note that the THOLD threshold entry can be omitted. In this case the default of the exact threshold is 1.

**Case 1:** Processing queries for objects with crisp and fuzzy attribute values. In this case, we rely on the calculation of the DIS distance measure and the SIM analog calculation to perform DIS and SIM calculations using various member functions to convert the fuzzy values of the properties contained in the database and the fuzzy values of the user from the conditional clause to the value form membership function. For example, the attribute "Area" with an opacity value of "Medium" is converted to a membership function value of "0.348", or the attribute "Price" with an explicit value of "840$" is converted to a membership function value of "0.7" with the blur interval value "Dearness". We build a query with single-condition and multi-condition clauses represented by the following algorithms:

1. Fuzzy query processing for single-condition case has the following form:

SELECT .....FROM C WHERE $A_{attr}$θfvalue THOLD fthreshold. The condition $A_{attr}$θ fvalue, where fvalue is the fuzzy value and $A_{attr}$ is the fuzzy attribute of the fuzzy class C, θ∈{=θ, ≠θ, <θ, >θ}, threshold $0 \le fthreshold \le 1$. Use the member function in Figure 4 to convert the fvalue to a value of membership.

**Algorithm1: FQSIMSC (Fuzzy Query Sim Single Condition)**.

**Input:** Class C with attributes {$A_1$, $A_2$,…, $A_n$}, set of objects of class C: {$O_i$, i = 1,…,m}, parameter fthreshold and $\alpha \in [0;1]$, K K positive integers with the default value K=1.

**Output:** Object set $O_{result}$ satisfying for all t ∈ O we have t[$A_{attr}$] θ fvalue with a give fthreshold.
Initialisation: K=1; $\alpha = 0.5$; $O_{result} = \emptyset$;

*1:* **Begin**
*2:* t ∈ O;
3: fvalue[m] $= fuzzy.$gaussmf($fvalue$) ;
*4:* **forAll** t ∈ O **do** */*t is the object extracted from the set O*/*
*5:* **Attvalue**[m] $= fuzzy.$gaussmf($\boldsymbol{t}$[$Aattr$]) ;/*m number of fuzzy subsets*/
*7:* **For** i = 0 to m **do** /*Apply formula (**10**) to calculate d(fvalue, Attvalue)*/
*8:* **d** +=|fvalue[i] − Attvalue[i]|$^{\mathbf{2}}$**;**
**End for**
*9:* D = (d/m)$^{1/2}$; /*Apply formula (**1**) to calculate S(fvalue, Attvalue)*/
*10:* $S = \frac{1-d}{1+K*d}$; /*Apply formula (**7**) and (**8**) to calculate Sim(fvalue, Attvalue)*/
*11:* $Sim = \frac{\alpha * S}{\sim}$;
*12:* **if** θisoperator =$_\theta$ **then**
*13:* **ifSim**(fvalue, Attvalue) == fthreshold**then**
*14:* $O_{result} = O_{result} \cup$ t;
*15:* **end if**

```
16:                Else
17:                    Case θof
18:                        ≠θ :if  Sim(fvalue, Attvalue) ≠ fthresholdthen
19:                                O_result = O_result ∪ t;
20:                        <θ :if  Sim(fvalue, Attvalue) < fthresholdthen
21:                                O_result = O_result ∪ t;
22:                        >θ :if Sim(fvalue, Attvalue) > fthresholdthen
23:                                O_result = O_result ∪ t;
24                         End case
25:                    End if
26:        end for
27:        return O_result;
28:    End
```

2. Fuzzy query processing for multi-conditions has the following form: SELECT ….. FROM C WHERE $A_{attr1}\theta_1 fvalue_1 \xi A_{attr2}\theta_2 fvalue_2$ THOLD fthreshold.

The condition $A_{attr1}\theta_1 fvalue_1$, $A_{attr2}\theta_2 fvalue_2$ where $fvalue_1, fvalue_2$ are fuzzy values and $A_{attr1}, A_{attr2}$ are fuzzy attributes C, $\theta_1$ $and$ $\theta_2 \in \{=_\theta, \neq_\theta, <_\theta, >_\theta\}$ threshold $0 \leq fthreshold, \leq 1$. $\xi$is the operation (**and/or).** Using gaussian membership functions in figure 4 to convert the values fvalue1, fvalue2 into member values.

**Algorithm2: FQSIMMC (Fuzzy Query Sim Multi-Condition)**

**Input**: Class C with attributes {$A_1$, $A_2$,…, $A_n$}, set of objects of class C: {$O_i$, i = 1,…,m}, parameter fthreshold and $\alpha \in [0; 1]$, K K positive integers with the default value K=1.

**Output:** Object set $O_{result}$ satisfying for all t ∈ O, we have t[$A_{attr}$] θ$fvalue_1$ $and$ $fvalue_2$with a given fthreshold. Initialisation: K=1; $\alpha = 0.5$; $O_{result} = \emptyset$;

```
1:    Begin
2:        O_result = ∅;
3:            fvalue1[m] = fuzzy.gaussmf(fvalue1) ;
4:        fvalue2[m] = fuzzy.gaussmf(fvalue2);
5:        t ∈ O;/*t is the object extracted from the set O*/
          For all t ∈ O do
6:    /*Using python gaussian member functions figure 2 convert the values t[A_attr1],
      t[A_attr2] to the values of membership*/
7:            Attvalue1[m] = fuzzy.gaussmf(t[A_attr1] ) ;
8:            Attvalue2[m] = fuzzy.gaussmf(t[A_attr2] );
9:            For i = 0 to m  /*m number of fuzzy subsets*/
10:               D1 +=(|fvalue1[i] - Attvalue1[i] |^2 ); /*Apply formula (10) to calculate D
11:               D2 +=(|fvalue2[i] - Attvalue2[i] |^2 );
```

*12:* **End for**

*13:* $D_1=( D_1 /m)^{\wedge} ½$ ;

*14:* $D_2=( D_2 /m)^{\wedge} ½$ ;

*15:* $S_1 = \frac{1-D_1}{1+K*D_1}$ ; */*Apply formula (**1**) to calculate $S_1$ */*

*16:* $S_2 = \frac{1-D_2}{1+K*D_2}$; */*Apply formula (**1**) to calculate $S_2$ */*

*17:* $Sim_1 = \frac{\alpha*S_1}{\alpha}$; */*Apply formula* (**7**) and (**8**) *to calculate $Sim_1$ */*

*18:* $Sim_2 = \frac{\alpha*S_2}{\alpha}$; */*Apply formula* (**7**) and (**8**) *to calculate $Sim_2$ */*

*19:* **if** ( $Sim_1(\text{fvalue}_1, \text{Attvalue}_1)\theta_1$ fthreshold) **and** ($Sim_2(\text{fvalue}_2, \text{Attvalue}_2)\theta_2$ fthreshold) **then**

*20:* $O_{result} = O_{result} \cup t$;

*21:* **End if**

*22:* **if** $\{Sim_1(\text{fvalue}_1, \text{Attvalue}_1)\theta_1$ fthreshold$\}$ **or** $\{Sim_j(\text{fvalue}_2, \text{Attvalue}_2)\theta_2$ fthreshold$\}$ **then**

*23:* $O_{result} = O_{result} \cup t$;

*24:* **End if**

*25:* **End for**

*26:* **return** $O_{result}$ ;

*27:* **End**

**Case 2:** Query processing for objects with estimated attribute values and Query processing for objects with estimated attribute values and objects of different classes. In this case, we rely on the probability distribution and semantic similarity calculations of SID and SEM. To perform calculations of SID and SEM, through the conditional clause, we use different suggestions [22 - 25] to separate and convert the estimated values of the attribute and the user's value to the distribution form ability. For example, the time attribute "about 21" is represented by a distribution with the following possibilities:$\left\{\frac{0.5}{18}, \frac{0.7}{19}, \frac{0.9}{20}, \frac{1.0}{21}, \frac{0.8}{22}, \frac{0.6}{23}, \frac{0.4}{24}\right\}$. The query for case 2 is represented as follows:

SELECT ….. FROM$C_1$, $C_2$WHERE $A_{attr1}\theta_1 fvalue_1 \xi A_{attr2}\theta_2 fvalue_2$ THOLD W.

The condition $A_{attr1}\theta_1 fvalue_1$, $A_{attr2}\theta_2 fvalue_2$ where $fvalue_1$, $fvalue_1$ are fuzzy values and $A_{attr1}$, $A_{attr2}$ are fuzzy attributes $C_1$and $C_2$, $\theta_1$ and $\theta_2\theta_1$ *and* $\theta_2 \in \{=_\theta, \neq_\theta, <_\theta, >_\theta\}$. $\xi$ is the operation (**and/or**). $\boldsymbol{w}$ is assigned to each attribute of fuzzy classes $C_1$ and $C_2$ based on its importance such that threshold $0 \leq w \leq 1$.

**Algorithm3: FQSEM (Fuzzy Query SEM)**

**Input:** Let $C_1$, be the class with attributes $\{a_1, a_2, \dots, a_k, a_{k+1}, \dots, a_m\}$ and $C_2$ be the class with attributes $\{a_1, a_2, \dots, a_k, a'_{k+1}, \dots, a'_m, a_{m+1}, \dots, a_n\}$ the set of objects of class $C_1$ and $C_2$: $\{O_i, i = 1, \dots, m\}$

**Output:** Oresult object set satisfying for all $t \in O_{result}$ with $0 < SE \leq 1$ given threshold $\boldsymbol{w}$

*Initialisation:* $O_{\text{result}} = \theta$

```
Begin
    fvalue1[m] = stats.norm(fvalue1)
    fvalue2[m] = stats.norm(fvalue2) ;
    t ∈ O/*t is the object extracted from the set O*/
    For All t ∈ O do
        Attvalue1[m] = stats.norm(t[Aattr1] ) ;
        Attvalue2[m] = stats.norm(t[Aattr2] );
    For k=0 to m /*m number of fuzzy subsets Apply formula (1) to calculate SID
        SID + = min (Attvalue2[k], fvalue2[k]);
        SID1 = SID /sum(fvalue2[k]);
        SID2 = SID /sum(fvalue2[k]); /*Apply formula (4) to calculate SE*/
        SE(SID1, SID2)=min(SID1, SID2) ; /*Apply formula (11) to calculate μ*/
        μ(Attvalue1, fvalue2) += SE(SID1, SID1) ∗ w;
        End for
    If (0 < μ(Attvalue1, fvalue1) ≤ 1 and 0 < μ(Attvalue2, fvalue2) ≤ 1) then
            Oresult = Oresult ∪ t;
    End if
    If (0 < μ(Attvalue1, fvalue1) ≤ 1 or 0 < μ(Attvalue2, fvalue2) ≤ 1) then
            Oresult = Oresult ∪ t;
    End if
  End for
return Oresult ;
End
```

**Case 1:** From the data in Table 1, it is necessary to extract information about "FOID, Apartment Type" with the condition of the query as "Area=Large and Price=Regular".

The query has the following form: FOQL1: SELECT C.FOID,C.ApartmentName FROM C WHERE C.Area=Large and Price=Regular THOLD 0.92. The results of fuzzy query (FOQL1) for case 1 are described in Table 2.

**Case 2:** From the data in Table 1, it is necessary to extract information about "FOID, Apartment Type". In which, the data extraction condition of the query is "Price=about $840" or "Area= Large". The structure of the fuzzy query is shown below: FOQL2: SELECT C.FOID, C.[Apartment Type] FROM C WHERE C.Price="about $840" or Area= "Large" THOLD 1/6. The results of fuzzy query (FOQL2) for case 2 are described in Table 3.

*Table 1.* Data list of fuzzy objects about apartment (for case 1).

| FOID | Apartment_T | Area_QV | Area_LT | Area_MV | Price _QV | Price_LT | Price_MV |
|---|---|---|---|---|---|---|---|
| Foid1 | Dual key | 200 | Large | 0.8 | 997 | Dearness | 0.95 |
| Foid2 | Penhouse | 80 | Medium | 0.78 | 785 | Dearness | 0.5 |
| Foid3 | Studio | 175 | Large | 0.409 | 350 | Regular | 0.737 |
| Foid4 | Penhouse | 79 | Medium | 0.348 | 200 | Dearness | 0.0201 |
| Foid5 | Dual key | 173 | Large | 0.409 | 350 | Regular | 0.737 |
| Foid6 | Studio | 79 | Medium | 0.348 | 840 | Dearness | 0.72 |

*Table 2.* Fuzzy query results for case 1.

| FOID | Apartment_T | Area_QV | Area_LT | Area_MV | Price _QV | Price_LT | Price_MV |
|---|---|---|---|---|---|---|---|
| Foid3 | Studio | 175 | Large | 0.409 | 350 | Regular | 0.737 |
| Foid5 | Dual key | 173 | Large | 0.409 | 350 | Regular | 0.737 |

*Table 3.* Data list of fuzzy objects about apartment (for case 2).

| FOID | Apartment_T | Area_QV | Area_LT | Area_MV | Price _QV | Price_LT | Price_MV |
|---|---|---|---|---|---|---|---|
| Foid1 | Dual key | 200 | Large | 0.8 | 997 | Dearness | 0.95 |
| Foid2 | Penhouse | 80 | Medium | 0.78 | 785 | Dearness | 0.5 |
| Foid3 | Studio | 175 | Large | 0.409 | 350 | Regular | 0.737 |
| Foid5 | Dual key | 173 | Large | 0.409 | 350 | Regular | 0.737 |
| Foid6 | Studio | 79 | Medium | 0.348 | 840 | Dearness | 0.72 |

### 3.3. Query processing based on data clustering and determining fuzzy intervals

*Table 4.* The result of the clustering algorithm EMC and Fuzzy interval.

| Fuzzy intervals | Fuzzy intervals |
|---|---|
| $Area._{Small}$ | $Area._{Small}$ |
| $Area._{Medium}$ | $Area._{Medium}$ |
| $Area._{Large}$ | $Area._{Large}$ |

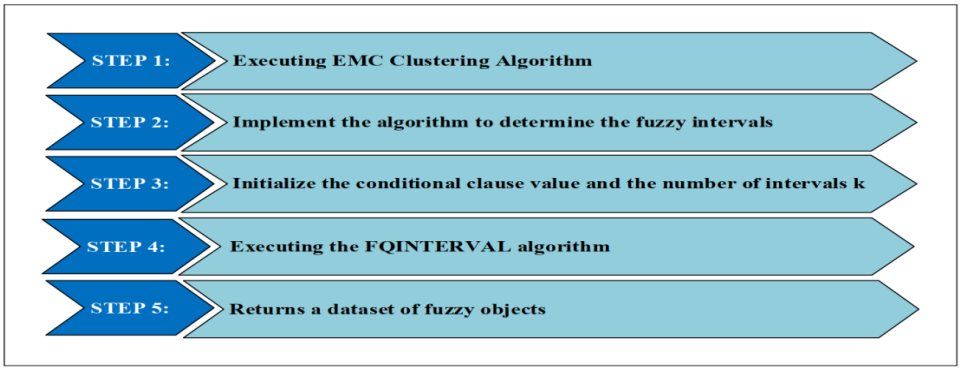


*Figure 4.* Description of the query processing algorithm on fuzzy intervals.

Besides the fuzzy query processing methods introduced above, in order to increase the efficiency and flexibility in the query, the paper proposes a fuzzy cluster query method based on the improved clustering algorithms EMC and the fuzzy partitioning algorithm of [26] as the basis for developing the query problem based on fuzzy intervals. The data in Table 4 show that the crisp value of the area attribute is assigned to three fuzzy areas, namely $\text{Area}_{Small\ (40,52,79,79,79)}$, $\text{Area}_{\text{Medium}(79,79,79,80,160,173)}$ and $\text{Area}_{\text{Large}(160,173,175,200)}$. The purpose of fuzzy interval classification is to convert quantitative values to linguistic values with linguistic variables as representative attributes. From there, to make data extraction by query statements more flexible and natural, or specifically, the value of the conditional clause is a linguistic value such as Small, Medium, or Large.

Implement fuzzy FOQL query algorithm based on fuzzy intervals

**Algorithm 5: FQINTERVAL (Fuzzy Query Interval)**.

.

**Input**: Let C be the class with attributes $\{a_1, a_2, \ldots, a_k, a_{k+1}, \ldots, a_m\}$ the set of objects of class $C: \{O_i, i = 1, \ldots, m\}$. The query has the following form: SELECT . . . FROM C WHERE $A_{attr} = fvalue$ THOLD 1.0

**Output:** The object set of $O_{interval}$ is satisfied with $A_{atr}\theta \in fvalue$. Initialisation: $O_{interval} = \emptyset$

*1:* **Begin**

*2:* Executing EMC Clustering Algorithm;

*3:* Implement the algorithm to determine the fuzzy intervals;
interval [k] contains fuzzy intervals after implementing EMC,

*4:* **for** $i = 0\ to\ k$ **do**

*5:* **if** $interval\ [i].value\ \boldsymbol{is}\ fvalue$ **then**

*6:* $O_{interval} = O_{interval} \cup interval[i]$;

*7:* End if

*8:* **End for**

*9:* **Return** $O_{interval}$ ;

From the data in Table 4, it is necessary to extract information about "FOID, Apartment Type" with the condition of the query as "Area is Small".

The query has the following form: FOQL3:SELECT C.FOID, C.ApartmentName, C.Area FROM C WHERE C.Area is 'Small' THOLD 0.92. The results of fuzzy query (FOQL3) is described in Table 5. By using the language variable of the area property and the linguistic value of this variable being small as the conditional clause, we can obtain a list of corresponding crisp values that is $\text{Area}_{Small\ (40,52,79,79,79)}$.

*Table 5.* Fuzzy query results based on fuzzy intervals.

| Foid | Apartment type | Area |
|---|---|---|
| Foid4 | Penhouse | 79 m |
| Foid6 | Studio | 79 m |

## 4. EXPERIMENTAL EVALUATION

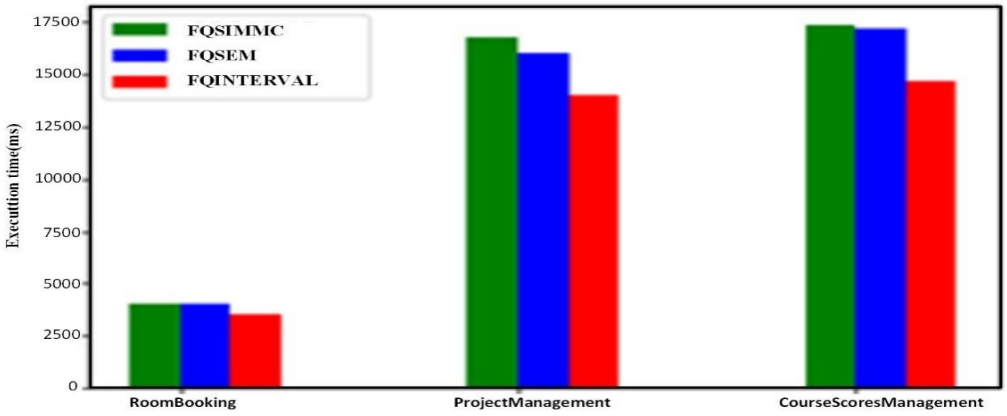


*Figure 5.* Algorithm execution time.

The results of the SQL3 query of example 7 show that the time to execute algorithm 5 (FQINTERVAL) for the RoomBooking database is less than that of algorithm 2 (FQSIMMC) and algorithm 3 (FQSEM), respectively to examples 5 and 6. The execution times of the RoomBooking, ProjectManagement and CourseScoresManagement datasets for algorithm 5 (FQINTERVAL) are 3503, 14012 and 14712, respectively (see Table 5). Compared with 2 algorithms (FQSIMMC) and (FQSEM) for RoomBooking, ProjectManagement and ManageCourseScores are 2045: 2024, 6135: 6072 and 6544: 6555, respectively (see Figure 5).

Through the experimental results performed on an Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz  2.90 GHz, 16GB RAM, Operating system Windows 10, We found that algorithm 5 (FQINTERVAL) is expensive lowest time. That is because the data extraction is performed directly on the preprocessed fuzzy partitions based on EMC clustering algorithm and fuzzy partitioning.

Analyzed data are presented in tables and figures with taking care to avoid unnecessary repetition of tabular data. Information presented in tables should not be repeated in figures, or vice versa. Standard deviations/errors help readers follow the trend of results and should be supplied whenever appropriate.

*Table 6.* Execution time in algorithms.

| **Dataset** | **Execution time in algorithms** | | |
|---|---|---|---|
| | **FQSIMMC** | **FQSEM** | **FQINTERVAL** |
| RoomBooking | 4045 ms | 4001 ms | 3503 ms |
| Project Management | 16810 ms | 16004 ms | 14012 ms |
| Course Scores Management | 17393 ms | 17204 ms | 14712 ms |

According to Table 7, the memory usage of the two algorithms FQSIMMC and FQSEM is larger than that of the FQINTERVAL algorithm. The memory usage of these algorithms for RoomBooking, ProjectManagement and CourseScoresManagement is 896, 2688 and 2867, respectively (see Figure 6). The two algorithms FQSIMMC and FQSEM use larger memory because both of them load all data into main memory for processing. However, the

FQINTERVAL algorithm uses less memory because the query processing is only performed directly on the defined fuzzy intevals.

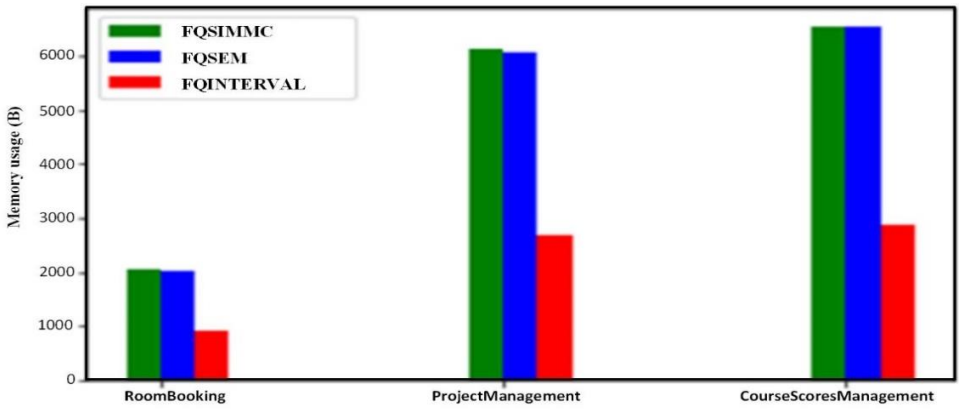


*Figure 6.* Evaluations of memory usage for different data sets.

*Table 7.* Memory usage in algorithms.

| Dataset | Execution time in algorithms | | |
|---|---|---|---|
| | **FQSIMMC** | **FQSEM** | **FQINTERVAL** |
| RoomBooking | 2045b | 2024b | 8896b |
| Project Management | 6135b | 6072b | 2688b |
| Course Scores Management | 6544b | 6555b | 2867b |

Conclusion for the experimental evaluation: From the experimental evaluation results of processing time and memory storage space, it can be seen that the clustered query processing algorithm FQINTERVAL consumes the best processing time and memory space compared to the other two algorithms (FQSIMMC, FQSEM). To explain this result, the FQINTERVAL algorithm performs direct data extraction based on pre-loaded fuzzy partitions in the main memory area and has the complexity of the O(n) algorithm. However, the FQINTERVAL algorithm has some limitations due to the lack of operations on the set (Union, intersec). In addition, the two algorithms FQSIMMC and FQSEM have $\boldsymbol{O(n^3)}$ complexity and large memory capacity because they have to read the internal data files from the auxiliary memory devices to the main memory. However, the advantages of these two algorithms are their continuous data updates and their flexibility when dealing with complex data.

## 5. CONCLUSIONS

The article proposes different methods to handle fuzzy queries effectively. By applying techniques such as semantic similarity assessment, similarity assessment for sharp and fuzzy data. From there, the article proposes four effective fuzzy query processing algorithms: FQSIMSC for single condition cases, FQSIMMC for multi-condition cases, FQSEMand clustered fuzzy query processing algorithm, which which are based on the improved clustering algorithm EMC and fuzzy partitioning method. However, each of these four algorithms has advantages and disadvantages. Therefore, depending on different situations, we choose the

appropriate query, such as the data type or frequency of data access as well as the size of the data, specifically:

- The FQINTERVAL cluster query processing algorithm performs data extraction directly based on preloaded fuzzy partitions in the main memory area due to measuring fast processing time. However, this algorithm has some limitations due to the lack of operations on sets (Union, intersec).

- The three algorithms FQSIMSC, FQSIMMC and FQSEM have large processing times and memory capacity due to having to read internal data files from secondary memory devices into main memory. However, the advantage of these three algorithms is that they update new and flexible data.

The evaluation of the proposed results is performed based on different datasets extracted from the UCI database.

***Acknowledgements.*** This research was funded by Research Project CS22.04 from the Institute of Information Technology, Vietnam Academy of Science and Technology.

***CRediT authorship contribution statement.*** Nguyen Tan Thuan: Methodology, Investigation, Funding acquisition, Formal analysis. Truong Ngoc Chau: Formal analysis. Nguyen Thi Anh Phuong and Tran Thi Thuy Trinh: Writing-original draft, Writing-review and edit, Methodology. Nguyen Truong Thang and Doan Van Ban: Methodology, Formal analysis, Funding acquisitionand Supervision.

***Declaration of competing interest.*** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## REFERENCES

1. Date C. J., and Warden A. - Relational database writings (1985–1989), Addison-Wesley Longman Publishing Co., Inc, 1990.
2. Ilyas I. F., Beskales G., and Soliman M. A. - A survey of top-k query processing techniques in relational database systems, ACM Computing Surveys (CSUR) **40** (4) (2008) 1-58. https://doi.org/10.1145/1391729.1391730.
3. De Tré G., De Caluwe R., and B der Cruyssen. - A generalised object-oriented database model, Recent issues fuzzy databases (2000) 155-182. https://doi.org/10.1007/978-3-7908-1845-1_8.
4. Bertino E. and Martino L. - Object-oriented database management systems: concepts and issues, Computer (Long. Beach. Calif) **24** (4) (1991) 33-47. https://doi.org/10.1109/2.76261.
5. Deng W. - Object-Oriented Database and O/R Mapping Technology, in Big Data Analytics for Cyber-Physical System in Smart City: BDCPS 2020, 28-29 December 2020, Shanghai, China (2021) 800-806. https://doi.org/10.1007/978-981-33-4572-0_115.
6. Simon J. P. - Scope, players, markets and geography, Digit. Policy, Regul. Gov., Artificial intelligence (2019). https://doi.org/10.1108/DPRG-08-2018-0039.
7. Expósito Solis A. and others - Implementation of a Telegram chatbox and webplatform for hypertension, 2020.
8. Liu C., Li X**.,** Li Q., Xue Y., Liu H., and Gao Y. - Robot recognizing humans intention and interacting with humans based on a multi-task model combining ST-GCN-LSTM model and YOLO model, Neurocomputing **430** (2021) 174-184. https://doi.org/10.1016/j.neucom.2020.10.016

9. Gupta M. M. and Yamakawa T. - Fuzzy logic in knowledge-based systems, decision and control, Elsevier Science Inc. (1988). https://doi.org/10.1109/40.566209.
10. Zadeh L. A. - Fuzzy probabilities, in Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems: Selected Papers by Lotfi A Zadeh, World Scientific, 1996, pp. 643-652.
11. Durrett R. - Probability: theory and examples, Cambridge university press **49** (2019).
12. Cheng C. B., Shih H. S., and Lee E. S. - Possibility Theory and Fuzzy Optimization, in Fuzzy and Multi-Level Decision Making: Soft Computing Approaches, Springer, 2019, pp. 73-88.
13. Umano M., Imada T., Hatono I., and Tamura H. - Fuzzy object-oriented databases and implementation of its SQL-type data manipulation language, in 1998 IEEE International Conference on Fuzzy Systems Proceedings, IEEE World Congress on Computational Intelligence (Cat. No. 98CH36228) **2** (1998) 1344-1349. https:// doi.org/ 10.1109/ FUZZY.1998.686314.
14. Bordogna G., Pasi G., and Lucarella D. - A fuzzy object-oriented data model for managing vague and uncertain information, Int. J. Intell. Syst. **14** (7) (1999) 623-651. https://doi.org/10.1002/(SICI)1098-111X(199907)14:7<623::AID-INT1>3.0.CO;2-G.
15. Van Gyseghem N. and De Caluwe R. - Imprecision and uncertainty in the UFO database model, J. Am. Soc. Inf. Sci. **49** (3) (1998) 236–252. https://doi.org/10.1002/(SICI)1097-4571(199803)49:3<236::AID-ASI5>3.0.CO;2-B.
16. Wedashwara W., Mabu S., Obayashi M., and Kuremoto T. - Evolutionary rule based clustering for making fuzzy object oriented database models, in 2015 IIAI 4th International Congress on Advanced Applied Informatics (2015) 517-522. https://doi.org/10.1109/IIAI-AAI.2015.167.
17. Srivastava A., Yadav S., Srivastava N., and Khan Z. - Fuzzy Query, An Impression in Query processing, 2016.
18. Drissi A., Nait-Bahloul S., Benouaret K., and Benslimane D. - Horizontal fragmentation for fuzzy querying databases, Distrib, Parallel Databases **37** (3) (2019) 441-468. https://doi.org/10.1007/s10619-018-7250-4
19. Zeng Y., Zhou Y., Zhou X., and Zheng F. - Fuzzy clustering-based skyline query preprocessing algorithm for large-scale flow data analysis, J. Supercomput **76** (2) (2020) 1321-1330. https://doi.org/10.1007/s11227-018-2523-2.
20. Mama R. and Machkour M. - Fuzzy Questions for Relational Systems, in The Proceedings of the Third International Conference on Smart City Applications (2019) 104-114. https://doi.org/10.1007/978-3-030-37629-1_9.
21. Liefke K. and Werning M. - Evidence for Single-Type Semantics An Alternative To/-Based Dual-Type Semantics, J. Semant **35** (4) (2018) 639-685. https://doi.org/10.1093/ jos/ffy009.
22. Ma Z. M., Zhang W. J., and Ma W. Y. - Assessment of data redundancy in fuzzy relational databases based on semantic inclusion degree, Inf. Process. Lett. **72** (1–2) (1999) 25-29. https://doi.org/10.1016/S0020-0190(99)00124-6.
23. Rahman K., Abdullah S., Ali A., and Amin F. - Interval-valued Pythagorean fuzzy Einstein hybrid weighted averaging aggregation operator and their application to group decision making, Complex & Intell. Syst. **5 (**1) (2019) 41-52. https://doi.org/ 10.1007/s40747-018-0076-x.
24. Dwibedy D., Sahoo L., and Dutta S. - A New Approach to Object Based Fuzzy Database Modeling, Int. J. Soft Comput. Eng. **3 (**1) (2013) 182-186. https://doi.org/10.35940/ijsce

25. Singpurwalla N. D. and Booker J. M. - Membership functions and probability measures of fuzzy sets, J. Am. Stat. Assoc. **99** (467) (2004) 867-877. https://doi.org/10.1198/016214504000001196.
26. Nguyen T. T., Van Doan B., Truong C. N., and Tran T. T. T. - Clustering and query optimization in fuzzy object-oriented database, Int. J. Nat. Comput. Res. **8** (1) (2019) 1-17. https://doi.org/ 10.4018/IJNCR.2019010101.
27. Bashon Y., Neagu D., and Ridley M. J. - A new approach for comparing fuzzy objects, in International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (2010) 115-125. https://doi.org/10.1007/978-3-642-140587_12.
28. Ma Z. M. - Object comparison in fuzzy object-oriented databases, In 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems **3** (2009) 672-675. https://doi.org/10.1109/ICICISYS.2009.5358091.