International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 12 Issue 10 ǁ October 2024 ǁ PP. 159-165

Attribute and Inter-Attribute domain constraints in fuzzy object oriented database based on hedge algebra

Cong-Hao Nguyen
Hue University, Vietnam
Corresponding Author: Cong-Hao Nguyen, email: nchao@hueuni.edu.vn

Abstract
In this paper, We proposed attribute and inter-attribute domain constraints on the classes of a fuzzy object-oriented database model based on hedge algebra. In this approach, the verification of whether an object satisfies the constraints according to specific conditions and whether the attributes are bound together or not is at level k ($k\in Z^+$), where k is the similarity level determined for each fuzzy attribute that matches the data constraints in the model, and the implementation of direct data matching on linguistic values is more intuitive and simpler than other approaches.

Keywords: *Hedge algebra, Fuzzy object-oriented database model, attribute constraints, inter-attribute constraints*

Date of Submission: 14-10-2024 Date of acceptance: 29-10-2024

I. INTRODUCTION

When designing a fuzzy object-oriented database, in addition to designing objects and object classes, designing constraints is a very important step. Designing and specifying constraints such as requirements that need to be satisfied between attributes within a class or attributes between classes [9-10]. For example, when storing information about lecturers in a university, some constraints can be used on the value domain of lecturer attributes such as citizen identification number, lecturer code, age, teaching and scientific research capacity... in which the value of citizen identification number, lecturer code must not be duplicated, the age of the lecturer is *relatively young*, the teaching and scientific research capacity of the lecturer is *very good*. The existence of constraints is a reality, so the problem is to include them in data models to make the data more meaningful and contextual [1]. Constraints are also used to set standards for updating and processing data to ensure consistency, correctness and completeness.

In general, an object-oriented database is desirable to users when it satisfies all possible constraints on the classes of that database, creating a flexibility and semantic completeness in terms of data storage and manipulation. On the contrary, there will be limitations on the data if a database does not satisfy or does not provide constraints. The structure of the paper is presented in 5 parts, in addition to the introduction, part 2 presents some basic knowledge about hedge algebra and fuzzy object-oriented databases based on hedge algebra, parts 3 and 4 present some data constraints such as value domain constraints and constraints between attributes, part 5 presents the conclusion and future research directions of the paper.

II. RELATED WORK

2.1. Hedge algebra

Hedge algebra is an approach to discovering the algebraic structure of the value domain of linguistic variables [9-10]. According to this approach, each value domain of the linguistic variable X can be understood as an algebra. $A\,X = (X, G, H, \leq)$, where $Dom\,(X) = \underline{X}$ is the domain of linguistic values of the linguistic variable $\underline{X}$ generated from the set of generating elements $G = \{c^-, c^+\}$ with the action of the hedges in $H = H^- \cup H^+$, $\leq$ is a semantic ordering relation on $\underline{X}$, it is induced from the natural qualitative meaning of the word. When some special elements are added, the hedge algebra becomes an abstract algebra $\underline{X}=(X, G, H, \Sigma, \Phi, \leq)$, where Σ, Φ are operators that take the limit of the set of elements generated when subjected to successive actions of hedges in H. Alternatively, if we denote $H(x) = \{h \;\ldots h_p\, x/h_1\,, \;\ldots\;, h_p \in H\}$ then $\Phi x = \inf H(x)$ and $\Sigma x = \sup H(x)$. Thus, the hedge algebra $\underline{X}$ built on the foundation of a hedge algebra $AX = (X, G, H, \leq)$, where $X = H(G)$, by adding two operators Σ, Φ. Then $X = X \cup \mathrm{Lim}(G)$ with $\mathrm{Lim}(G)$ being the set of limiting elements: for all $x \in \mathrm{Lim}(G)$, there exists $u \in X$ such that $x = \Phi u$ or $x = \Sigma u$. These limiting elements are added to the hedge algebra $\underline{X}$ to make the new calculations meaningful and so $\underline{X} = (X, G, H, \Sigma, \Phi, \leq)$ is called a fully linear hedge algebra. The semantic quantification function (υ), the fuzziness measure function (*fm*), the sign function (Sgn) and the properties of the hedge algebras can be found in published works [6-8].

2.2. Similarity level *k*

When defining a neighborhood of level *k* we want such representative values to be interior points of the neighborhood of level *k*. Therefore, we define the similarity of level *k* as follows: we always assume that each set H^- and H^+ contains at least 2 hedges. Let X_k be the set of all elements of length *k*. Based on the fuzzy intervals of level *k* and the fuzzy intervals of level *k*+1 we informally describe the construction of a partition of the domain [0, 1] as follows:

i) With $k = 1$, the level 1 fuzzy intervals include $I(c^-)$ and $I(c^+)$. The level 2 fuzzy intervals on the interval $I(c^-)$ are $I(h_p c^-) = I(h_{p-1}c^-) = \ldots = I(h_2c^-) = I(h_1c^-) = \upsilon_A(c^-) = I(h_{-1} c^-) = I(h_{-2} c^-) = \ldots = I(h_{-q+1} c^-) = I(h_{-q} c^-)$. Then, we construct a level 1 similarity partition consisting of the following equivalence classes: $S(0) = I(h_p c^-)$, $S(c^-) = I(c^-) \setminus [I(h_{-q}c^-) \cup I(h_pc^-)]$; $S(W) = I(h_{-q}c^-) \cup I(h_{-q}c^+)$; similarly we have $S(c^+) = I(c^+) \setminus [I(h_{-q}c^+) \cup I(h_p c^+)]$ and $S(1) = I(h_pc^+)$.

ii) Similarly, with $k = 2$, we can construct a partition of similarity classes of level 2. In a similar way, we can construct partitions of similarity level *k* at any. However, in practical applications, we can restrict the consecutive hedges acting on the primitive elements c^- and c^+ to some integer k^*. The canonical values and the fuzzy values are said to have similarity of level *k* if their representative values lie in the same similarity class of level *k*.

Example 2.1. Consider the relational schema *U* = {CODE, *FULLNAME, NUMBER_ISI, NUMBER_PRO*} with the meaning: Lecturer code (CODE), Lecturer's full name (FULLNAME) are 2 explicit attributes, Number of articles published in prestigious international journals (*NUMBER_ISI*), Number of topics at all levels chaired (*NUMBER_PRO*) are 2 attributes with fuzzy values. In which $D_{NUMBER\_ISI} = [0, 20]$ and $D_{NUMBER\_PRO} = [0, 10]$. LD_{NUMBER_ISI} and LD_{NUMBER_PRO} have the same set of linguistic values with the generator set {0, *low*, *W*, *high*, 1} and the hedge set {*less*, *possibly*, *more*, *very*}. *Although the* considered fuzzy attributes have the same set of linguistic values, their quantitative semantics are different.

i) ***For the attribute NUMBER_ISI***

We have: *fm*(*high*) = 0.4, *fm*(*low*) = 0.6, μ(*possibly*) = 0.25, μ(*less*) = 0.3, μ(*more*) = 0.2 and μ(*very*) = 0.25. We partition the interval [0, 2 0] *into* 5 intervals similar to level 1: *fm*(*very high*) × 20 = 0.25 × 0.4 × 20 = 2. Therefore, *S*(1) × 20 = (18, 20]; (*fm*(*possibly high)* + *fm* (*more high*)) × 20 = (0.25 × 0.4 + 0.2 × 0.3) × 20 = 3.2 and *S*(*high*)× 20 = (14.8, 18]; (*fm*(*less low*) + *fm*(*less high*)) × 20 = (0.3 × 0.6 + 0.3 × 0.4) × 20 = 6 and *S*(*W*) × 20 = (8.8, 14.8]; (*fm*(*possibly low)* + *fm*(*more low*)) × 20 = (0.25 × 0.6 + 0.2 × 0.6) × 20 = 5.4 and *S*(*low*) × 20 = (3.4, 8.8], *S*(0) × 20 = [0, 3.4].

ii) ***For the attribute NUMBER_PRO***

We have: *fm*(*high*) = 0.35, *fm*(*low*) = 0.65, μ(*possibly*) = 0.2, μ(*less*) = 0.25, μ(*more*) = 0.35 và μ(*very*) = 0.2. We partition the interval [0, 10] into 5 intervals similar to level 1: *fm*(*very high*) × 10 = 0.2 × 0.35 × 10 = 0.7. Therefore, *S*(1) × 10 = (9.3, 10]; (*fm*(*possibly high*) + *fm*(*more high*)) × 10 = (0.2 × 0.35 + 0.35 × 0.35) × 10 = 1.925 and *S*(*cao*) × 10 = (7.375, 9.3]; (*fm*(*less low*) + *fm*(*less high*)) × 10 = (0.25 × 0.65 + 0.25 × 0.35) × 10 = 2.5 and *S*(*W*) × 10 = (4.875, 7.375]; (*fm*(*possibly low*) + *fm*(*more low*)) × 10 = (0.25 × 0.65 + 0.35 × 0.65) × 10 = 3.9 and *S*(*low*) × 10 = (0.975, 4.875], *S*(0) × 10 = [0, 0.975].

2.3. Fuzzy object-oriented database

In Fuzzy Object Oriented Database [9], the following four cases can be used to distinguish object class relationships:

i) Clear class and clear object: this case is similar to that in object-oriented databases, meaning that an object belongs or does not belong to a class with certainty.

ii) Sharp class and fuzzy object: class is precisely defined and has precise boundaries, while object is fuzzy because its attribute values can be fuzzy. In this case, object can be a member of class with some degree of belonging .

iii) Fuzzy class and clear object**:** similar to case ii), objects can belong to classes with different degrees of belongingness *k*. For example, a class of young students and a 20 year old student.

iv) Fuzzy class and fuzzy object: in this case, the object also belongs to the class with the degree of belonging level *k*.

For each fuzzy linguistic value *x*, we will define an interval representation for *x*. In practice, the number of hedges in linguistic values is finite so there exists a positive integer k^* such that $0 < |x| \le k^*$, $\forall x \in \boldsymbol{X}$. For any $x \in \boldsymbol{X}$, set $j = |x|$, for every integer *k*, with $1 \le k \le k^*$, the minimal neighborhood k of *x* denoted by $O_{min,k}(x)$ is defined as follows: if $k = j$ then $O_{min,k}(x) = I_{k+1}(h_{-1}x) \cup I_{k+1}(h_1x)$, if $1 \le k < j$ then $O_{min,k}(x) = I_j(x)$ and if $j+1 \le k \le k^*$ then $O_{min,k}(x) = I_{k+1}(h_{-1}y) \cup I_{k+1}(h_1y)$. Therefore, we represent fuzzy linguistic data according to the following definition:

Definition 2.1. Given $x \in \mathbf{X} \cup C$, an interval representation of x is a set *IRp(x)* of intervals defined: $IRp(x) = \{O_{min,k}(x) | 1 \leq k \leq k^*\}$.

Consider $\underline{\mathbf{X}}$ as a hedge algebra, with $H^+ = \{h_1, ..., h_p\}$ and $H^- = \{h_{-1}, ..., h_{-q}\}$, where $p, q > 1$. Let H_1 be the set of negative hedges, H_2 be the set of positive hedges in the sense that when acted upon it will change the meaning stronger than the number of hedges in H_1, that is, the sets H_1 and H_2 include: $H_1 = \{h_i, h_{-j} | 1 \leq i \leq [p/2], 1 \leq j \leq [q/2]\}$, $H_2 = \{h_i, h_{-j} | [p/2] \leq i \leq p, [q/2] \leq j \leq q\}$. Let $P_{k+1}(H_n) = \{I(h_i y) | y \in X_k, h_i \in H_n\}$, with n = 1, 2. Two intervals I(x) and I(y) in $P_{k+1}(H_n)$ are called connected if there exist consecutive intervals in $P_{k+1}(H_n)$ ranging from I(x) to I(y). This relation will divide $P_{k+1}(H_n)$ into connected components. We have that, for each $y \in \mathbf{X}_k$, $P_{k+1}(H_1)$ is divided into clusters of the form $\{I(h_i y) h_i \in H_1\}$. Furthermore, $I(h_{-1}y) \leq \upsilon(y) \leq I(h_1 y)$ or $I(h_1 y) \leq \upsilon(y) \leq I(h_{-1}y)$ we have $\upsilon(y) \in \{I(h_i y) | h_i \in H_1\}$. We cluster the fuzzy intervals of $P_{k+1}(H_2)$. Suppose $\mathbf{X}_k = \{x_s | s = 0, \ldots, m-1\}$ consists of m elements arranged in a sequence such that $x_i \leq x_j$ if and only if $i \leq j$. Set $H_2^- = H_2 \cap H^-$ and $H_2^+ = H_2 \cap H^+$. Note that $h_{-q} \in H_2^-$ and $h_p \in H_2^+$. The clusters generated from the fuzzy intervals of $P_{k+1}(H_2)$ are of three types: the cluster to the left of x_0: $I(h_i x_0) | h_i \in H_2^+\}$; the cluster to the right of x_{m-1}:$\{I(h_i x_{m-1}) | h_i \in H_2^+\}$; the cluster between x_s and x_{s+1} with s = 0,…, m-2, depend on $Sgn(h_p x_s)$ and $Sgn(h_p x_{s+1})$ as follows: $\mathbf{C} = \{I(h_i x_s), I(h_j' x_{s+1}) | h_i \in H_2^+, h_j' \in H_2^-\}$, if $Sgn(h_p x_s) = +1$ and $Sgn(h_p x_{s+1}) = +1$; $\mathbf{C} = \{I(h_i x_s), I(h_j' x_{s+1}) | h_i \in H_2^+, h_j' \in H_2^+\}$, if $Sgn(h_p x_s) = +1$ and $Sgn(h_p x_{s+1}) = -1$; $\mathbf{C} = \{I(h_i x_s), I(h_j' x_{s+1}) | h_i \in H_2^-, h_j \in H_2^-\}$, if $Sgn(h_p x_s) = -1$ and $Sgn(h_p x_{s+1}) = +1$; $\mathbf{C} = \{I(h_i x_s), I(h_j' x_{s+1}) | h_i \in H_2^-, h_j \in H_2^+\}$, if $Sgn(h_p x_s) = -1$ and $Sgn(h_p x_{s+1}) = -1$.

The set of all clusters is denoted by $\mathbf{C}$. Because $\{S_k(\mathbf{C}) | C \in \mathbf{C}\}$ is a partition on the reference domain, it defines an equivalence relation and we will call it the *k* -level similarity relation. Due to the nature of the partition, for each value x of the attribute, there exists a unique cluster $\mathbf{C}$ such that $\upsilon(x) \in S_k(\mathbf{C})$ and we define the *k- level similarity interval* as follows:

Definition 2.2. For each *C* in $\mathbf{C}$, we call the similarity interval of level *k* corresponding to $\mathbf{C}$ as: $S_k(\mathbf{C}) = \cup\{I(u) | I(u) \in \mathbf{C}\}$, then $S_k(x) = S_k(\mathbf{C})$.

Definition 2.3 . Given any object o on the attribute set $\{A_1, A_2, ..., A_n\}$ of class *C*, $\underline{\mathbf{X}}$ is a hedge algebra, for each *k*, $1 \leq k \leq k^*$, S_k is a similarity relation of level *k* on the attribute domain A_i of class *C*. Then, for every $u \in \mathbf{X}$, the values $o(A_i)$ and u are said to be equal at level *k*, denoted $o(A_i) =_k u$, if and only if $O_{min,k}(o(A_i)) \in S_k(u)$.

Definition 2.4 . Given any two objects o_1, o_2 on the attribute set $\{A_1, A_2, ..., A_n\}$ of class *C*, $\underline{\mathbf{X}}$ is a hedge algebra , for each k, $1 \leq k \leq k^*$, S_k is a similarity relation of level *k* on the attribute domain A_i of class *C*. Then:

i) Two values $o_1(A_i)$ and $o_2(A_i)$ are said to be equal to level *k*, denoted $o_1(A_i) =_k o_2(A_i)$, if and only if there exists an equivalence class $S_k(u)$ of the similarity relation S_k such that $O_{min,k}(o_1(A_i)) \in S_k(u)$ and $O_{min,k}(o_2(A_i)) \in S_k(u)$.

ii) Two values $o_1(A_i)$ and $o_2(A_i)$ are said to differ by degree *k*, denoted $o_1(A_i) \neq_k o_2(A_i)$, if there does not exist an equivalence class $S_k(u)$ of the similarity relation S_k such that $O_{min,k}(o_1(A_i)) \in S_k(u)$ and $O_{min,k}(o_2(A_i)) \in S_k(u)$.

Definition 2.5. Let $C = \{c_1, c_2, .., c_n\}$ and $O = \{o_1, o_2, \ldots, o_m\}$ be the set of constraints and the set of objects on class C, respectively. For each constraint $c_i \in C$ that one or more objects $o_j \in O$ is fully defined through its syntax and semantics as follows:

Syntax of a constraint:

$c^{<name>}_{<properties>}$[ε] in <class name> [reference to <class name*>]

[satisfied with level of <*k*>]

Where: <name>: name of the constraint; <properties>: Set of properties of the class participating in the constraint; [ε] : with ε either an empty value; or a set of properties; or a numeric value, range or fuzzy value depending on each given constraint; <class name>: name of the class participating in the constraint; [reference to <class name*>]: name of the class that references the constraint if any; [satisfied with level of <*k*>]: Level k that satisfies the constraint (if any).

III. ATTRIBUTE DOMAIN CONSTRAINTS

According to the hedge algebra approach, when considering the relationship between classes/objects or objects that satisfy fuzzy query conditions, that object belongs to a class or satisfies fuzzy query expressions with a level *k*. certain definition. With this idea, from the data constraint perspective, we also consider objects that satisfy constraints with a level *k* due to the fuzziness expressed in the given constraints. For example, as a constraint for the lecturer object, the salary attribute is *relatively high* and the number of published scientific research papers is *quite large.* Therefore, the satisfaction of this constraint should only be evaluated by a certain level depending on the fuzzy interval of level *k*. To evaluate level *k* like that, we must represent the values on the attributes of the objects participating in the constraint as fuzzy intervals at each level. Then, determining the objects satisfying the constraint is done by matching the attribute values of the objects participating in the constraint at each level k with the usual matching method, this is a prominent advantage of the hedge algebra approach.

For fuzzy attributes, the data types are also based on the basic data types as above to allow the representation of fuzzy information. Some specific proposed constraints are as follows:

"not null" constraint: For this constraint, the attribute on the specified object must always have a defined and valid value. To represent this constraint, we use the syntax: $C_{\{id\}}^{not\_null}$ in <class name> (3.1)

Where, id describes the attribute to which the "not_null" constraint is applied on the class named <class name>.

"null" constraint: For this constraint, the property on the object can contain null value as opposed to the "not_null" case above. To represent this constraint we use the syntax: $C_{\{id\}}^{null}$ in <classname>. (3.2)

"Value" constraint: To represent this constraint, we use the syntax: $C_{\{id\}}^{value}[\ \varepsilon]$ in <class name> Satisfied with level of <k>. (3.3)

Where, εis a value constraint that can be an explicit value, an interval value, and an opaque linguistic value to limit the value domain of the attribute specified as id and k is a given level indicating the constraint satisfaction level of the object on the class named <class name>.

Example 3.1. For example, we constrain the Salary attribute of the Lecturer class to values such as about 10,000,000 or 15,000,000 ≤Salary ≤20,000,000 or "very high" then we have the corresponding constraint representations as follows:

$C_{\{Salary\}}^{value}$[about 10,000,000] in Satisfied Lecturer with level of <k>

$C_{\{Salary\}}^{value}$[15,000,000 ≤ Salary ≤ 20,000,000] in Lecturer Satisfied with level of <k>

$C_{\{Salary\}}^{value}$[very high] in Satisfied Staff with level of <k>.

If the level k equality relation ($k \in Z^+$) occurs, then there exists an equivalence class $S_k(x) \in S_k$, where S_k is an equivalence relation of level k on the attribute id such that the minimum neighbor represents the value of the attribute under consideration and the value constraint εbelong to the same equivalence class $S_k(x)$.

Algorithm 3.1. Check whether an object o satisfies the value constraint c with a given level k.

Input: Object o and attribute id to be considered. Constraint c is represented from (3.3), $k \in Z^+$ is given to indicate the constraint satisfaction level of object o, S_k is an equivalence relation on the domain of attribute id.

Output: Object o either satisfies the value constraint or not.

Method:

```
1. Begin
2. Constructing the hedge algebra $AX_{id}$ for the attribute id includes constructing the fuzzy parameters, hedges and generators.
3. Determine the domain that defines the id attribute as D(id) = ($min_{id}$, $max_{id}$) where $min_{id}$, $max_{id}$ are the smallest and largest values of id.
4. Calculate $O_{min,k}(o(id))$ and $O_{min,k}(\varepsilon)$
5.          if ($O_{min,k}(o(id)) =_k O_{min,k}(\varepsilon)$) then
6.                   return "object o satisfies constraint c with level k"
7.          else
8.                   return "object o does not satisfy constraint c with level k"
9. End.
```

Algorithm 3.1 is correct, algorithm complexity is O(1).

IV. INTER-ATTRIBUTE DOMAIN CONSTRAINTS

In reality, there are constraints of the form "*If two employees o_1, o_2 have relatively good working ability, then the salary of o_1, o_2 is relatively high*", it is seen that there is a relationship between the teaching ability and salary of o_1, o_2. If we consider that the relationship between the teaching ability and salary of two lecturers o_1, o_2 is a constraint in a fuzzy object-oriented database, we call this constraint a constraint between attributes on class C containing these two attributes, or in other words, it is a form of fuzzy data dependency between two attributes.

Definition 4.1. Let A = {A_1, A_2,.., A_m} be a set of attributes and C be a class defined on that set of attributes. Consider X, Y⊆ A. We say that the object $o \in C$ satisfies the inter-attribute constraint X, Y denoted $X \Rightarrow_k Y$, read as Y, the fuzzy inter-attribute constraint X with level k: if $\forall o_i \in C$, $o_i(X) =_k o(X)$ then $o_i(Y) =_k o(Y)$.

The syntax for inter-attribute constraints is:

$C_{\{Y_1, Y_2,\dots, Y_m\}}^{inter\_properties}$[$X_1$, X_2,.., X_n] in <class name> Satisfied with level of <k> (4.1)

In which, X = {X_1, X_2,.., X_n} and Y = {Y_1, Y_2 ,.., Y_m} are the sets of attributes on the class named <class name>, required the level of satisfaction of the constraint between X and Y.

Algorithm 4.1. Algorithm to check whether an object $o \in C$ satisfies the given inter-attribute constraint X, Y with a given level $k \in Z^+$ or not.

Input: A = {$A_1, A_2, \ldots, A_p$} is the set of attributes on class C, the object $o \in C$ to be considered. C contain the set of objects $o_1, o_2, \ldots, o_q$ and $X, Y \subseteq A$, X = {$X_1, X_2, \ldots, X_n$}, Y = {$Y_1, Y_2, .., Y_m$}, $m, n \le p$; the inter-attribute constraint is shown at (4.1), $k \in Z^+$.

Output: Object *o* satisfy the inter-attribute constraint $X \sim>_k Y$ with level $k \in Z^+$ or not.

Method:

```
1. Begin
2. Construct the hedge algebras for the fuzzy attributes in X and Y, assuming the sets X, Y have s, r fuzzy attributes in X and Y respectively as S = {$X_1, X_2, \ldots, X_s$}, R = {$Y_1, Y_2, \ldots, Y_r$}.
3. Determine the value domain of the $X_i$ attributes $\in S$ and $Y_i \in R$ is $D(X_i) = (min_{Xi}, max_{xi})$ and $D(Y_i) = (min_{Yi}, max_{Yi})$ where $(min_{Xi}, max_{Xi})$ and $(min_{Yi}, max_{Yi})$ are the minimum and maximum values of $X_i$. $Y_i$, respectively.
4. i: = 1; stop: = false;
5. while (i $\le$ q-1) and (stop = false) do
6.       begin
7.             if SatisfyX($o_i(X) =_k o(X)$) then
8.                   begin
9.                         if SatisfyY($o_i(Y) =_k o(Y)$) then Stop:=false;
10.                        else Stop: =true;
11.                  end i: = i+1;
12.      end
13. if (i $\ne$ q-1) then return "o does not satisfy the constraint"
14. else  if (i = q-1) then return "o satisfies the constraint"
15. End.
//Building function SatisfyX, function SatisfyY
1. Begin
2. r: = 1; Stop:=false;
3. while (r $\le$ n) and (stop = false) do
4.       begin
5.             if ($X_r$ is an explicit attribute) then
6.                   begin
7.                         if $o_i(X_r) = o(X_r)$ then stop:= false
8.                         else stop: = true;
9.                   end
10.            else if ($X_r$ is a fuzzy attribute) then
11.            begin
12.                  Find $O_{min,k}(o_i(X_r))$ and $O_{min,k}(o(X_r))$;
13.                  if ($O_{min,k}(o_i(X_r)) =_k O_{min,k}(o(X_r))$) then stop:=false
14.                  else stop:= true;
15.            end
16.            r: = r+1;
17.      end
18. if (r = n) then return SatisfyX: = true
19. else return SatisfyX: = false;
20. End.
//Building the SatisfyY function
1. Begin
2. z: = 1; Stop:= false;
3. while (z $\le$ m) and (stop = false) do
4.       begin
5.             if ($Y_z$ is an explicit attribute) then
6.                   begin
7.                         if $o_i(Y_z) = o(Y_z)$ then stop:= false
8.                         else stop:= true;
9.                   end
10.            else if ($Y_z$ is a fuzzy attribute) then
11.            begin
12.                  Find $O_{min,k}(o_i(Y_z))$ and $O_{min,k}(o(Y_z))$;
```

```
13.                     if ($O_{min,k}(o_i(Y_z)) =_k O_{min,k}(o(Y_z))$) then stop:=false
14.                     else stop:= true;
15.                end
16.                z: = z+1;
17.          end
18. if (z = m) then return SatisfyY: = true
19. else return SatisfyY: = false;
20. End.
```

With the object *o* that needs to satisfy the constraints between the proposed attributes, algorithm 4.1 need to go through the remaining t-1 objects on that class. Algorithm 4.1 is correct, ensuring stopping and stopping when the function *SatisfyX*($o_i(X) =_k o(Y)$) satisfies and the function *SatisfyY*($o_i(Y) =_k o(Y)$) does not satisfy or has gone through all the objects of the class. The algorithm complexity of the *Satisfy*X and *Satisfy*Y functions is O(n) and O(m) respectively. Therefore, the complexity of algorithm 4.1 is O(t), with n, m ≤ t.

Corollary 4.1: Given Object $o \in C$, $A = \{A_1, A_2,..., A_m\}$ is the attribute set of class *C*. Let $X,Y \subseteq A$ with Y fuzzy inter-attribute constraint at level *k*, denoted $X \rightsquigarrow_k Y$, we have:

i) With *X* being fuzzy or clear. If object *o* satisfies constraint $X \rightsquigarrow_k Y$ at level *k* then *o* also satisfies constraint $X \rightsquigarrow_{k'} Y$ at all levels $k' \in Z^+$: $k' < k$.

ii) With *X* it is clear. If object *o* does not satisfy constraint $X \rightsquigarrow_k Y$ at level *k* then *o* also does not satisfy $X \rightsquigarrow_{k'} Y$ at all levels $k' \in Z^+$: $k' > k$.

Proof.

i) Because object *o* satisfies constraint $X \rightsquigarrow_k Y$ level *k* , we have $\forall o_i \in C$, $o_i \neq o$: $o_i(X) =_k o(X) \Rightarrow o_i(Y) =_k o(Y)$ or $o_i(X_i) =_k o(X_i) \Rightarrow o_i(Y_i) =_k o(Y_i)$ (*), $\forall X_i \in X$, $1 \leq i \leq n$, *n* is the number of attributes in *X* and $\forall Y_i \in Y$, $1 \leq i \leq m$, *m* is the number of attributes in *Y*. For the left side of (*), for each $o_i(Y_i) =_k o(X_i)$, if $o_i(X_i) =_k o(X_i)$ then $o_i(X_i) =_{k'} o(X_i)$, $\forall k' < k$. Similarly for the right side of (*), for each $o_i(Y_i) =_k o(Y_i)$ then we have $o_i(Y_i) =_{k'} o(Y_i)$, $\forall k' < k$. Therefore, for $o_i(X_i) =_k o(X_i) \Rightarrow o_i(Y_i) =_k o(Y_i)$ we have $o_i(X_i) =_{k'} o(X_i) \Rightarrow o_i(Y_i) =_{k'} o(Y_i)$ with $\forall k' < k$ or $o_i(X) =_{k'} o(X) \Rightarrow o_i(Y) =_{k'} o(Y)$ with $\forall k' < k$ if *o* satisfies the constraint $X \rightsquigarrow_{k'} Y$ for all levels $k' \in Z^+$: $k' < k$.

ii) Because object *o* does not satisfy constraint $X \rightsquigarrow_k Y$ level *k*, we have $\exists o_i \in C$, $o_i \neq o$: $o_i(X) =_k o(X) \Rightarrow o_i(Y) \neq_k o(Y)$ or at least one attribute Y_i *exists* $\in Y$: $o_i(X_i) =_k o(X_i) \Rightarrow o_i(Y_i) \neq_k o(Y_i)$(**),$\forall X_i \in fK$, $1 \leq i \leq n$, *n* is the number of attributes in *X*. For the right side of (**), if $o_i(Y_i) \neq_k o(Y_i)$ then $o_i(Y_i) \neq_{k'} o(Y_i)$, $\forall k' > k$. Similarly for the left side of (**), since *X* is a clear key ($\forall X_i \in X$ is an explicit attribute) so for $o_i(X_i) =_k o(X_i)$ then $o_i(X_i) =_{k'} o(X_i)$, $\forall X_i \in X$, $\forall k' > k$. Therefore, $\forall k' > k$, $\exists o_j \in C$, $o_j \neq o$: $o_j(X_i) =_{k'} o(X_i) \Rightarrow o_j(Y_i) \neq_{k'} o(Y_i)$,$\forall X_i \in X$, $\exists Y_i \in Y$ in other words $\exists o_j \in C$, $o_j \neq o$: $o_j(X) =_{k'} o(X) \Rightarrow o_j(Y) \neq_{k'} o(Y)$ so *o* also does not satisfy $X \rightsquigarrow_{k'} Y$,$\forall k' \in Z^+$: $k' > k$ where *X* is a clear key.

Corollary 4.2: Given Object $o \in C$, $A = \{A_1, A_2,..., A_m\}$ is the attribute set of class *C*. Let $X,Y, Z \subseteq A$. We have:

i) If object *o* satisfies constraint $X \rightsquigarrow_k Y$ level *k* then *o* also satisfies constraint $XZ \rightsquigarrow_k YZ$ level $k \in Z^+$.

ii) If object *o* satisfies constraints $X \rightsquigarrow_k Y$ and Y $\rightsquigarrow_k Z$ with level *k* then *o* also satisfies constraint $X \rightsquigarrow_k Z$ with level $k' \in Z^+$: $k' > k$.

Proof.

i) Because object *o* satisfies constraint $X \rightsquigarrow_k Y$ level *k*, then $\forall o_i \in C$, $o_i \neq o$: $o_i(X) =_k o(X) \Rightarrow o_i(Y) =_k o(Y)$. Furthermore, with $o_i(XZ) =_k o(XZ)$ and $o_i(X) =_k o(X)$ implies: $o_i(Z) =_k o(Z)$ (*).

From $o_i(Y) =_k o(Y)$ and $o_i(Z) =_k o(Z)$ implies: $o_i(YZ) =_k o(YZ)$ (**). Thus, from (*) and (**) we have $\forall o_i \in C$, $o_i \neq o$: $o_i(XZ) =_k o(XZ) \Rightarrow o_i(YZ) =_k o(YZ)$ or we say that object *o* satisfies constraint $XZ \rightsquigarrow_k YZ$ level $k \in Z^+$.

ii) Because object *o* satisfies constraints $X \rightsquigarrow_k Y$ and Y $\rightsquigarrow_k Z$ level *k* should be $\forall o_i \in C$, $o_i \neq o$: $o_i(X) =_k o(X) \Rightarrow o_i(Y) =_k o(Y)$ (*) and $o_i(Y) =_k o(Y) \Rightarrow o_i(Z) =_k o(Z)$ (**). From (*) and (**) we have $\forall o_i \in C$, $o_i \neq o$: $o_i(X) =_k o(X) \Rightarrow o_i(Z) =_k o(Z)$ or we say that object *o* satisfies constraint $X \rightsquigarrow_k Z$ at level $k \in Z^+$.

V. CONCLUSION

In the process of designing fuzzy data structures, determining the types of data constraints is an important issue. For example, the type of constraints on the attribute value domain is the basis for determining fuzzy keys, the constraints between attributes are the basis for determining fuzzy functional dependencies, an important basis in building fuzzy standard forms to minimize data redundancy. The constraints on the attribute value domain aim to ensure the determination of data according to a certain rule specified when designing the database and ensure the correctness of the data of an object in the fuzzy object class. Researching and proposing algorithms related to these types of data constraints is meaningful in terms of theory as well as application design in practice. Some issues related to special types of constraints are researched and presented in the following works.

REFERENCES

[1]. Nguyen Kim Anh, "Normalizing object-oriented database schema" , Journal of Computer Science and Cybernetics, 9 (2), (2003), 125-130.

[2]. Ho Cam Ha, Vu Duc Quang, "Fuzzy function dependencies in fuzzy object-oriented databases", Journal of HNUE, 7, pp 23-31, 2011.

[3]. Nguyen Cong Hao, "Fuzzy functional dependencies based on hedge algebras", International Journal of Computer Technology and Applications, Vol (6) 6, 2015, pp 1052-1059.

[4]. Nguyen Cong Hao, "Fuzzy functional dependency with linguistic quantifiers base on hedge algebra", Journal on Information and Communications Technology, 22, (2), (2009), 87-93.

[5]. Nguyen Cong Hao, "Fuzzy Normal Forms an Approach hedge Algebra", Journal on Information and Communications Technology, (17), (2008), 101-107.

[6]. Nguyen Cong Hao, Le Thi My Le, "Data dependencies in fuzzy object oriented databases model based on hedge algebras", Annales Uni. Sci. Budapest. Sect. Comp, Vol 44 (2015), pp. 165-182.

[7]. Nguyen Cong Hao, Truong Thi My Le, "Fuzzy object-oriented database model base d on hedge algebra", Journal of Computer Science and Cybernetics, 20, (3), (2012), 129-140.

[8]. Nguyen Cat Ho, Le Xuan Vinh, Nguyen Cong Hao, "Unify data and establish similarity relation in linguistic databases base on hedge algebra", Journal of Computer Science and Cybernetics, 25 (4), (2009), 314-332.

[9]. ZM Ma, "Advances in Fuzzy Object-Oriented Databases: Modeling and Applications", Idea Group Publishing, 2004.

[10]. Cristina-Maria Vladarean, "Extending object-oriented databases for fuzzy information modeling", SC WATERS Romania SRL, Romai J., 2 (1) (2006), 225-237.