Indefinite Information with Certainty Factors for Object-Oriented Databases

Haechull Lim*
Kyhyun Um**
Youngbae Park***

*Dept. of Computer Science, Hong-Ik University, Mapo-Gu, Seoul, 121-791, Korea
**Dept. of Computer Science, Dongguk University, Chung-Gu, Seoul, 100-715, Korea
***Dept. of Computer Science, Myung-Ji University, Yongin, Kyunggi-Do, 449-728, Korea

ABSTRACTS

This paper presents an approach to integrate the concepts of indefinite information with certainty factor in an object-oriented framework. In this work, the concept of traditional indefinite information is extended to include certainty factors so that indefiniteness can be measured. Indefinite information with certainty factors is classified into two categories: certainty factors of association and certainty factors of object. Each of these categories is applied to each type of object: scala value, rule, and operation, etc. Instances with indefinite information with certainty factors including definite object is shown to be stored in G-relation. Various types of answers generated when query is issued are defined. Several rules for the computation of certainty factors based on Fuzzy Set theory are suggested to compute resultant certainty factor for the answer in query processing.

1. Introduction

When database contains only partial information which users want to know, we call this information incomplete information.

The notion of information incompleteness seems to be inherent in the domain of databases and many researchers have been progressed upon theoretical backgrounds. However, very little has been done toward modelling backgrounds and practically using them.

Resultantly, present database products provide little or no support for information incompleteness, though the situation when data are incomplete is quite common. Especially, since information is scattered in the areas of military, police, marketing, medical diagnosis, and weather forecasting, and since databases in those areas may be only a partial model of the world which it is intended to represent, it is necessary to extend the scope of information what databases contain, and to use it in information society.

Up to now, many researchers including J. Minker[Mink84], W. Lipski, Jr[Lips81], and T. Imielinski[Imie84], etc., have developed methods to handle situations where a database does not contain all the information a user would like to know exactly. But, their researches are primarily based on the relational, or slightly modified relational model, which is widely used in developing database management systems for its simplicity and mathematical backgrounds.

On the other hand, for the past decade, several object-oriented (O-O) semantic data models have been introduced as potential alternatives for modeling many advanced database applications. For example, Smalltalk[GoI83], ORION[Ban87], and OSAM*[Su90], etc., are worthy of close attention to capture much more of the semantics of application domains and to overcome the limitations of the traditional record-oriented data models. Especially, OSAM* of these O-O semantic data models allows the user to model the complex structures and relationships among objects that exist in the application world, so he/she can represent ideally the semantics of an application in a database.

Thus far, O-O semantic association model has assumed that all the information about objects considered is complete in spite of the fact that information can be acquired by direct observation or by the interaction with some outside agent. This information is potentially very vague or indefinite in nature. More importantly, the user of the object-oriented database can not simply wait for it to stabilize in some final and complete form since this may never happen. For such reasons, it is necessary to be accomplished to formulate some preliminary ideas on how to integrate incomplete information concepts into O-O semantic association model.

In this paper, we extend the concepts of the object to include indefinite information with certainty factor

* Research supported by the Korea Science and Engineering Foundation grant as a post doctoral program
so that indefiniteness can be measured quantitively. We classify indefinite information with certainty factor into two categories, i.e., association C.F and object C.F, which are applied to each group of object, scalar value, rule, and operation, etc. Instances with indefinite information with C.F are shown to be store in G-relation. For queries against indefinite information, various types of answers are defined. Also, C.F. computation rules based on fuzzy Set theory are suggested to compute resultant C.F of such answers in query processing.

It has a number of distinct features:

First, we expand the traditional indefinite information to indefinite information with C.F so that each operand of indefinite information can be measured in terms of its possibility that it may be true. This concept integrates definite and indefinite information into information with C.F, and provides more concrete information than simple indefinite information. Second, we classify C.F which can occur in objects into association C.F and object C.F according to the characteristics of object. Third, we extend the scope of indefinite object which can be handled as intra-attribute indefiniteness, inter-attribute indefiniteness within one object, and inter-class indefiniteness within one object. Finally, we devise C.F. computation rules based on Fuzzy Set theory, so that C.F. of answers can be computed.

This paper is organized as follows. Section 2 gives the general concepts and other research works about indefinite information with C.F. The object-oriented view of databases is described. In Section 3, we represent the integration of indefinite information with certainty factor into object in U-U semantic association model. In Section 4, mention about query processing method is suggested to compute resultant certainty factor of answer in query processing. A summary and further researches are given in Section 5.

2. Incomplete information & object-oriented view of databases.

When database contains only partial information which users want to know, we call this information incomplete information. Generally, incomplete information is classified into three types. The first is "null value" at special symbol '0' that contains one or more existential quantifiers[Codd75]. Many researchers have studied null values in the scope of "exists as an attribute value but unknown what value in domain". The second type of incomplete information is a numerical interval of ['a, b'] that contains a value with range rather than an atomic value, for example, anyone's height is greater than 'a' and less than 'b'-centimeters, but not exactly known. W. Lipski, Jr.[Lips81] has developed this type of incomplete information based on mathematical backgrounds. The third type of incomplete information is indefinite information that contains type of 'a v b', which means that a is true or b is true[Min84]. Indefinite information occurs in real application world and is more worthwhile than simple "unknown values" when one of a few information is certainly true. We limit the scope of incomplete information to indefinite information. Instead, the concept of indefinite information dealt with in this paper is extended to include C.F, which measures the certainty of object, which ranges from 0(ignorance) to 1(completely true).

On the other hand, the U-U semantic association model provides a conceptual basis for uniformly capturing the semantics and inter-relationship among objects in the application world. These objects can be physical objects, abstract things, or events. Objects are grouped together into object classes (entity object class : E-class, domain object class : D-class) based on some common semantic properties they share. Object in the application world that are accessed independently are modeled by E-class. Each object of a U-U semantic association model is represented by a unique object identifier(ID). A D-class, on the other hand, models the structure and the behavior of objects that primarily serve as descriptive data of some other objects. An association of a class captures the semantic and structural relationship between the class and some other class(es). In order to give a better description of the U-U representation, we shall briefly describe a simple application domain using the U-U semantic association model [Su89] as shown in Figure 1 which is used in the remainder of this paper.

In OSAM*, object classes are graphically represented as nodes and associations among object classes are represented as links. The resulting diagram is called the Semantic diagram or S-diagram. E-classes and D-classes are represented in the S-diagram as rectangular and circular nodes, respectively.

There are five types of associations in OSAM*: three of these association types appear in Figure 1, namely, Aggregation(A), Generalization(G) and Interation(I), two of these are also recognized in several other semantics and U-U data models. A class can have several types of links and more than one link of each type emanating from it. As an example, in Figure 1, the E-class PERSON has two types of links: aggregation links connecting PERSON to D-classes sn# and name, and Generalization links to the E-class STAFF, FACULTY, and STUDENT(i.e., STAFF, FACULTY, and STUDENT are subclasses of the superclass PERSON). An aggregation link represents an attribute and has same name as the class it connects to unless specified others. An E-class with an interaction association which is specific in OSAM*, is connected to its constituent classes through links, which are grouped and labeled with an I in the S-diagram.

The links represent attributes of the E-class, and the constituent E-classes serve as the domains for the attributes. For any object in this E-class, the corresponding values of these attributes indicate the objects from the constituent domains which have entered into an interaction. For example, in Figure 1, ADVISING is an E-class that models an interaction between
two E-classes FACULTY and STUDENT. Each instances of ADVISING contains a fact that a specific FACULTY advises a specific STUDENT. A detailed description of the OSAM* model can be found in [Su89].

Next, the general syntax and semantics of OQL( Object-Oriented Query Language) [Alas89] are described, which was designed as an example of a new query model for O-O database and which was implemented for OSAM*. A query block in OQL consists of a CONTEXT clause and an operation clause. A CONTEXT clause has two optional subclauses: a WHERE subclause and a SELECT subclause. This structure is shown below.

CONTEXT association pattern expression WHERE conditions
SELECT object classes and/or attributes
OPERATION(S) object class(es)

In the CONTEXT clause, the user specifies a desired subdatabase pattern types of interest. A linear association pattern expression has the form 'A[intra-class conditions] op B[inter-class conditions] op ... ' where 'op' is one of the association pattern operators. The conditions that can be specified in the WHERE subclause are inter-class comparison conditions, which are comparisons between some attributes of two classes. The SELECT subclause operatoes on the subdatabases returned by the CONTEXT clause and its optional WHERE subclause to produce a new subdatabase and/or derived attributes. The operation clause specifies a set of messages(operation names) to be sent to the classes of the subdatabase.

3. Integration of indefinite information with C.F.

As we have introduced in section 2, objects can be physical objects, abstract things, or events. Objects are grouped together into object classes based on some common semantic properties they share. Objects in an O-O database are represented independently of their descriptive properties and their associations with other objects. In consideration of simple domain object class, objects can be grouped into three categories. The first one is scalar value such as string, character, integer, real, and boolean, etc.. The second one is operation such as function, procedure, and compute, etc.. The third one is rule which has the functions of deductivity and integrity constraint etc., and has IF and THEN clauses. On the other hand, a composite domain object class can be defined in terms of other class including Set, Vector, Matrix, and Ordered-set. We consider three kinds of simple D-class and a set type of composite D-class with respect to the concepts of indefinite information with C.F.

Before extending the concepts of object, we give a formal description of the concepts of indefinite information and that of C.F. for the general form of object.

[DEF 1] General form of object with indefinite information with C.F. is as the following:

\[ \text{val}_1(C_{F1}) \lor \text{val}_2(C_{F2}) \lor \text{val}_3(C_{F3}) \]

, where \( \text{val}_i, i=1,...,k \), is an object.

\( C_{Fi}, i=1,...,k \), is the certainty factor of \( \text{val}_i \).

\( 0 < C_{Fi} < 1, \sum C_{Fi} = 1 \)

\( \lor \) is 'logical OR'.

The number of operands in the expression of object is called 'the degree of indefiniteness'.

For example, if we specify 'a(0.3) \lor b(0.7)' as an object, it means 'a' may be true with the reliability of 0.3 or 'b' may be true with the reliability of 0.7. The degree of indefiniteness is 2 in this case, 'a \lor b \lor c' means that 'a' or 'b' or 'c' is true with the same probability, i.e., a(1/3) \lor b(1/3) \lor c(1/3). In this case, the degree of indefiniteness is 3. 'a(0.4)' means that 'a' may be true with certainty factor 0.4 or 'a' may not be true with the certainty factor of 0.6. It has the same meaning as 'a(0.4) \lor \neg a(0.6)'. If we specify 'a' as an object, it means 'a' is definitely true with C.F = 1. It has the same meaning as 'a(1) \lor \neg a(0)' which means that 'a' is true with C.F = 1 or 'a' is not true with C.F = 0.

With this general form, we can extend the scope of an object dealt with from simple indefinite value or value with certainty factor to four kinds of objects, i.e., indefinite value with C.F(ex. a(0.3) \lor b(0.7)), indefinite value(ex. a \lor b \lor c), definite value with C.F(ex. a(0.4)), definite value(ex. a). We also recognize the fact that the latter three kinds of objects are the special types of indefinite value with C.F.

This concept expands the traditional indefinite information of object to indefinite information with C.F so that each operand of indefinite information can be measured in terms of its degree of belief that it may be true. From now on, we collectively call these four kinds of object described above as 'indefinite object with C.F'.

Indefiniteness with C.F occurred in object-oriented database is classified into two categories. The first one is object indefiniteness with C.F that represents the certainty that an object itself may be included as a member of E-class or D-class. The second one is association indefiniteness with C.F which represents the certainty that an object in an E-class may have an association with another object in its constituent E-class or D-class. All the objects in E-class or D-class have one object indefiniteness with C.F and one or more association indefiniteness with C.F. For simplicity, we abbreviate object indefiniteness with C.F as 'object C.F', and association indefiniteness with C.F as 'association C.F'.

We explain how to integrate each category of indefiniteness with C.F into each type of object. As described earlier, object can be grouped into scalar
value, rule, and operation.

In case of scalar value, it can be divided into two kinds: one is an OID representing the entity of object in E-class, another is a scalar type of object in D-class. Each of these can have an object C.F and an association C.F respectively.

Figure 2 shows an extensional diagram corresponding to the subdatabase of Figure 1. An entity object (OID='01') in FACULTY E-class has 0.6 as its object C.F, which means the object with OID=01 has 80% as the degree of belief that this object may be a member of FACULTY. Also, the object (OID=11) in STUDENT E-class has '1' as its object C.F, which means that the entity object (OID=11) is definitely a member of STUDENT. There exists an entity object (OID='21', C.F=0.9) in ADVISING E-class that models an interaction between two E-classes, i.e. STUDENT and FACULTY. C.F=0.9 of OID=21 is an object C.F. It means there exists an object(OID=21) that represents the fact that OID=01 in FACULTY may advise OID=11 in STUDENT with C.F=0.9. Also, an entity object can have associations with other entity objects according to the types of association. Therefore, OID can have association C.F. As an example, C.F=0.9 in object with OID=21 is an object C.F in ADVISING E-class, and concurrently, it becomes an association C.F for OID=11 and 01 in STUDENT and FACULTY E-class respectively.

An E-class can be defined to have an aggregation association with other object classes. This association defines a set of attributes for the defined class which serves as domains for these attributes respectively. In Figure 2, a scale value of '0.5' is an attribute value of 'major' with a member of entity object(OID=01) in FACULTY E-class, and it is also a member of D-class of 'major'. Since '0.5' is definitely a member of 'major' D-class, the C.F of object is 1. The association C.F of '0.5' is 0.7. Therefore the certainty that '0.5' may be an attribute value of 'major' associated with entity object '01' is 0.7.

Next, we extend the concept of rule specification to integrate the concept of indefiniteness. Rules are specified in OQL(Object-Oriented Query Language) [Alas89] which was designed as an example of a new query model for 0-0 database. A deductive-rule in OQL has an If-Then structure as belows:

IF CONTEXT association pattern expression
WHERE conditions
THEN (subdatabase-id(classes)) : (attribute := expression)

The IF clause of a deductive rule may contain an OQL expression that identifies certain association patterns in the database. In the CONTEXT clause, the user specifies a desired subdatabase by specifying its intensional pattern and extensional pattern types of interest(Both are specified in the association pattern expression). A linear association pattern expression has the form 'A[intra-class conditions] op B[intra-class conditions]'. Each operator separates two E-classes that are directly associated in a schema. The intra-class conditions enclosed in brackets following a class are optional and are expressed in the form of predicates that involve the descriptive attributes of that class. The conditions that can be specified in the WHERE subclause are inter-class comparison conditions between some attributes of two classes. The THEN clause of a rule derives either a new subdatabase or the values of a derived attribute.

We deal with only the type of 'definite value with C.F', not 'indefinite value with C.F'. Rule objects, like the scalar value, can also have object C.F and association C.F. The object C.F in rule measures the reliability that each rule becomes a member of domain class which that rule belongs to. But with respect to association C.F, various types of C.F should be considered since the structure of the rule is more complex than that of scalar value.

Certainty factors that can exist in rule consist of those in rule itself, in IF-clause, and in Then-clause. C.F in rule itself represents the fact that the rule as well as its associated data may be uncertain. Each rule has an attenuation, a number from 0 to 1 which indicates its certainty. The C.F of a rule itself influences on determining the C.F of the value of a new subdatabase or the value of derived attributes in its THEN clause. Since each attribute value referred in the IF-clause may have a C.F, a C.F also appears in IF-clause. This type of C.Fs influences to compute how much the conditions in IF-clause satisfy that rule. This C.F has different values for different objects. We can classify the C.F in IF-clause into implicit C.F and explicit C.F. The implicit C.F does not appear in IF-clause explicitly because it represents the association C.F of object whose instance appears in G-relations. Instead, when an object is applied in executing rule, this C.F is revealed by the C.F of object which takes part in associationship between classes. Explicit C.F appears in IF-clause explicitly with the shape of variable in intra-class condition or in inter-class condition. This C.F is substituted into real C.F of object when an object is applied. There C.Fs in IF-clause are computed according to simple rule proposed in Fuzzy Set theory introduced in later section. This computed C.F measures how much related object satisfies the condition, and influences on computing the C.F of a new subdatabase or the values of newly derived attributes in Then-clause.

C.F can also appear in Then-clause. That type of C.Fs is the C.F of object which new subdatabase or derived attributes directly refer. As a result, three kinds of C.Fs, i.e. C.F of rule itself, C.F in IF-clause, and in Then-clause, determine the C.F of each object in the new subdatabase or that of derived attribute. This is called a derived C.F which is derived from the C.F in rule itself, the C.F in IF-clause, and the C.F in Then-clause. This resultant C.F(derived C.F) is the association C.F of two categories of C.F. In Figure 2, C.Fs 0.54 and 0.65 of R1, R2 are the association C.Fs of R1, R2,
The syntax of complete structure of rule in the rule detail. C.F of the object value is described in Section 4 in structure of a rule containing the trigger condition. The body. The C.F of trigger condition specifies the Then clauses in rule. Explicit C.Fs in If-clause are X
condition of If-clause, is associated with STUDENT 'LEONE' satisfies the
Z=O. 64.

We exemplify a rule as:

' If the major of an advisor is 'D.B' and
his/her interest area is 'D.C, then there is a
tendency that his/her student's interest area is the
same as that of advisor with 84% as the degree of
belief'.

This rule is translated into the following using OQL.

R:O.8: IF CONTEXT FACULTY[major='DB(x)' and
    fa-int='D.C(Y)'] * ADVISING * STUDENT
THEN STUDENT[st-int(UJ1 =
    FACULTY[fa-int(Z))]

The C.F of the rule itself is 0.8, which shows the
attenuated level of relationship between If and
Then clauses in rule. Explicit C.Fs in If-clause are X
and Y which may have different values according to each
object. Implicit C.Fs in If-clause may occur in the association
with objects in FACULTY class and objects in
STUDENT class. Let a faculty SMITH'S major be
'D.B(1.0)' and his/her interest area be 'D.C(0.8)' and
he may advise STUDENT 'LEONE' with C.F=0.9. Since
explicit C.Fs in the If-clause are X=1.0, Y=0.8,
respectively, and implicit C.Fs in the If-clause is
0.9, C.F in If-clause becomes MIN(MIN(1.0,0.8),0.9)=0.8
according to the Fuzzy Set theory. This shows
that the reliability that his ADVISOR 'SMITH'
associated with STUDENT 'LEONE' satisfies the
condition of If-clause, is 0.8. Since SMITH'S
interest area is 'D.C(0.8)', C.F in Then-clause is
Z=0.64.

As a result. STUDENT 'LEONE'S interest area is
'D.C(U)' . Value 'U' can be computed as follows:

U = 0.8 x MIN(MIN(1.0, 0.8) , 0.9) x 0.8 = 0.512

The computational method to obtain the resultant
C.F of the object value is described in Section 4 in
detail.

We also extend the simple rule to complete the
structure of a rule containing the trigger condition.
The syntax of complete structure of rule in the rule
specification language of OSM* is as follows:

TRIGGER-COND(trigger-specification)
rule-body not-exist-expression
if-then-else-rule
cardinality-constraints:
CORRECTIVE-ACTION action-specification

The certainty Factor in the complete structure of a
rule is similar to that in the simple rule. There are
two kinds of C.Fs with respect to trigger condition,
i.e., the C.F of trigger condition and the C.F of rule
body. The C.F of trigger condition specifies the
attenuation in its reliability between the relationship of
the TRIGGER-COND clause and the rule-body clause.
The C.F of rule-body clause is the composition of C.Fs
occurring in the simple rule described above, and which
is similar to the C.Fs of a nested rule.

For example, if we specify the following
constraint, 'After updating the interest area of
FACULTY, Rl should be executed with C.F = 0.7'

Rule RO(0.7)
Trigger-cond(After Update(FACULTY[fa-int])
EXECUTE Rl
END RO

Rl(0.6): IF CONTEXT FACULTY * ADVISING * STUDENT
THEN STUDENT[st-int(Y) = FACULTY[fa-int(X)]

The C.F of trigger condition is 0.7 and the C.F of
rule body would be the resultant C.F of executing Rl.
Since a rule can contain a sub-rule in its THEN clause
recursively, its C.F also can be represented and
computed recursively.

As the third type of object, an operation, like
scalar value and rule, can also have both object C.F
and association C.F. object C.F in an operation
measures the reliability that each operation becomes
the member of domain class which that operation
belongs to. Since the domain class which an operation
belongs to may have associations with other classes, an
operation type of object also have a association C.F.
In Figure 2, the function Fl has I as its object C.F
since we assume that each operation defined by the user
and/or the database administration becomes definitely a
member of the domain class which that operation belongs
to. C.F of an operation itself is the association C.F,
since the reliability of an entity object associated
with the specified operation depends on the C.F of
operation itself. We see the function Fl has 0.8 as a
association C.F in Figure 2.

Next, we extend the concepts of set value in OSM*
to integrate the concepts of indefinite information
with C.F. Since the set value is a composite domain
class in OSM*, we simply define the set value as
follows:

[DEF 2] General form of set value of object is
as the following:

let el, e2, . . . , ek be objects defined in [Def 1].
Then, S is the set value of object with indefinite
information with C.F.

S={el, e2, . . . ,ek}

where el = vali(CFi), i=1, . . . , k, and vali, CFi are
defined in [Def 1]. If, for all i, j = 1, . . . , k, CFj =
CFi, then S = {valj, val2, . . . , valk}(CFi).

For example, (a(0.3), b(0.7)) is a set value of an
object consisting of two elements. On the other hand,
(a(0.3) V b(0.7)) is a set value consisting of
one element. (a(0.7), b(0.7)) can be abbreviated as
(a, b)(0.7).

Set value like scalar value in simple D-class can
have object C.F and association C.F. But, since we
regard a set value as an atomic value, we apply object C.F and association C.F to set value in the same way as to scalar value.

To store the instances with definite and indefinite information with C.F, we use the G-relation from [Su89]. A G-relation provides a tabular representation of an E-class and its instances. Each E-class has a corresponding G-relation. The G-relations corresponding to the E-classes in Figure 1 are shown in Figure 3. A G-relation is similar to a relation in the relational model, but the values of an attribute in a G-relation do not have to be atomic. Additionally, the structure of an attribute in a G-relation can itself be a G-relation, i.e., a G-relation can contain other G-relations nested to an arbitrary level to represent complex objects. Furthermore, a G-relation recognizes different types of attributes.

From FACULTY and ADVISING in the G-relation in Figure 3, we can see four types of scalar values. The value 'D.B' is a definite value, '0.S(0.7)' is a definite value with C.F, '13 V 14' as a STUDENT attribute value of OID=23 is an indefinite value, C.(A.0.2) as the major attribute value of OID=03 and C.(A.0.8) as the fa-int attribute value of that object represent indefinite value with C.F. It means that OID 03's major is 'C.(A.0.2)' or its fa-int is 'C.(A.0.8)'. Also, we can extend the scope of indefinite object in forms of intra-attribute indefiniteness, inter-attribute indefiniteness within one object, and inter-class indefiniteness within one object, '13 V 14' in ADVISING is an example of intra-attribute indefiniteness. To represent inter-attribute indefiniteness within one object in the G-relation of FACULTY, we introduce a 'fac' attribute whose type of domain is rule. By specifying the exclusiveness of a value between two attributes, major and fa-int in 'fac' attribute, we can represent inter-attribute indefiniteness within one object. It means that the value of major with OID=03 is 'C.(A.0.2)' or its fa-int is 'C.(A.0.8)'.

Objects in sat-int and sc-up attributes in the STUDENT G-relation are a type of rule, not a type of scalar value. It means that STUDENT's interesting area and desired occupation are represented not in the form of scalar value directly, but in the form of deduced values obtained by executing rule. In reality, in Figure 3 and 4, we can see that the value of STUDENT's interest area is influenced by his/her ADVISOR's interest area through rule R1, R2, and values of STUDENT's desired occupation is influenced by his/her ADVISOR's major or interest area.

In Figure 4, the meaning of rule R1 is that STUDENT's interest area depends on his/her ADVISOR'S interest area with C.F = 0.6. Rule R2 means that if an ADVISOR's major is 'D.B' and his/her interest area is 'D.C', then the interest area of his/her STUDENT is the same as the interest area of ADVISOR with 80% as the degree of belief. The meaning of R3 is that if the major of one's ADVISOR is 'D.B' or the interest area of ADVISOR is 'D.C', and his/her STUDENT's interest area is 'D.C', then the desired occupation of that STUDENT would be 'S.A' with 70% as the degree of belief.

As we have described above, there are many types of C.F in the rule, and it is very complex to identify which C.F influences to compute the C.F of derived new attributes and how C.Fs are related with each other. We show C.F relationship structure in rules in the form of diagram. There are three kinds of C.Fs in rule as we described earlier and each kind of C.Fs corresponds to a rectangle. The C.F of a rule itself is stored in C.F in condition clause in which is divided into two parts, the upper part corresponds to implicit C.F, lower part corresponds to explicit C.F respectively, and C.F in action clause is in C.F. The resultant C.F of the rule is in the shape of circle. Many C.Fs exist in each rectangle in the form of constant, variable, and expression. Figure 5(a) shows a C.F relationship structure for rules R1 - R4 in Figure 4.

There can be dependencies between rules because it may be necessary to execute another rule to determine whether the condition clause of a rule is satisfied or not, or to determine which rule needs to be selected for specified derived attribute. For example, in Figure 4, to determine whether st-int = 'D.C(2)' in R3 is satisfied or not for a certain object, it is necessary to execute R1, R2 before executing R3. Therefore, the resultant C.F of a rule(R1 or R2 in Figure 4) should be passed to the condition-clause of referring rule(R3 in Figure 4). Figure 5(b),(c) illustrate this dependency. If rules are applied in parallel, average or maximum value of C.F is selected, and if rules are illustrated as serial. C.Fs are multiplied and the resultant C.F is substituted to the variable of C.F of the next applied rule. The detailed computing rule of C.F is described in the next section.

4. Query Processing

4.1 Answer types

In this section, we show how to process queries and which type of answer is generated if queries are input. Queries are classified into three types according to its function: addition, deletion, and retrieval. We only deal with a retrieval operation and other kinds of operations remain as a further research. We choose OQL as sample queries.

To list the types of answers suggested in this paper, let ai, bi, ci(i=1, ...,k) be attribute values of objects a*, b*, c* respectively, and CFi, CFj, CFI(i=1, ...,k) be the certainty factors of attribute values ai, bi, ci. Then, there are three kinds of answer types as follows:

(1) definite answer: ai
   this type of answer is same as traditional answer.

(2) answer with certainty factor: ai(CFI)
   For example, FACULTY 'DOLL' 's major is '0.S(0.7)'.
3. indefinite answer with certainty factor
   - single attribute: \( a_i(CF) \lor b_i(CF) \)
     For example, the name of \( STUDENT \) whose \( FACULTY \) is \( 'CLAYTON' \) is \( 'DEBOIS(0.5) \lor SMITH(0.5)' \).
   - group attribute: \( [a_i(CF), a_j(CF)](CF^a) \lor [b_i(CF), b_j(CF)](CF^b) \)
     For example, the name and the interest area of student who is advised by \( FACULTY \) 'CLAYTON' is \( [DEBOIS, C.A(0.6)](0.5) \lor [SMITH, A.1(0.4)](0.5) \).
     This type of answer can be generated when more than two attributes are specified for output. And, the meaning of this example is as follows: certainty factor '0.5' attached to bracket means the reliability that each student is advised by faculty 'CLAYTON' and certainty factors attached to each attribute values are represented as the reliability that each attribute value may be true. And so, the student whom \( FACULTY \) 'CLAYTON' advises is 'DEBOIS' or 'SMITH'. If the object 'DEBOIS' is true, his/her name and interest area are 'DEBOIS' and 'C.A(0.6)' respectively, or if the object 'SMITH' is true, his/her name and interest area are 'SMITH' and 'A.1(0.4)'.

4.2 Applying Fuzzy Set theory

To answer the query, it is necessary to combine C.Fs to derive new C.F as the basic operations of a query are performed. We only translate some basic operations performed on Fuzzy Sets to apply C.Fs in 0-0 database because the definition of operations on Fuzzy Set theory has generally been accepted since Lofti Zadeh first introduced that in 1962. We call this as 'operations on certainty factor (OCF)'.

Let \( A, B \) be set of facts, predicates, or rules, and \( P(A) \) be the C.F that \( A \) may be true.

Operations on certainty factor.
1. The intersection of \( A \) and \( B \) is denoted by \( A \cap B \) and is defined by
   \[ P(A \cap B) = \min(P(A), P(B)) \]
2. The union of \( A \) and \( B \) is denoted by \( A \cup B \) and is defined by
   \[ P(A \cup B) = \max(P(A), P(B)) \]
   A justification of the choice of \( \max \) and \( \min \) was given by Bellman and Gertz(1973).
3. The complement of \( A \) is denoted by \( \bar{A} \) and is defined by
   \[ P(\bar{A}) = 1 - P(A) \]
4. The product of \( A \) and \( B \) is denoted by \( A \cdot B \) and is defined by
   \[ P(A \cdot B) = P(A), P(B) \]
5. The bounded sum of \( A \) and \( B \) is denoted by \( A \oplus B \) and is defined by
   \[ P(A \oplus B) = \min(1, P(A) + P(B)) \]
   where '\( \oplus \)' is the arithmetic sum.
6. The bounded difference of \( A \) and \( B \) is denoted by \( A \ominus B \) and is defined by
   \[ P(A \ominus B) = \max(0, P(A) - P(B)) \]
   where '\( \ominus \)' is the arithmetic difference.

7. If \( A_1, \ldots, A_k \) are some facts, predicates, or rules, and \( w_1, \ldots, w_k \) are nonnegative 'weights' adding up to unity, then a convex combination of \( A_1, \ldots, A_k \) is defined by
   \[ P(w_1, A_1 + \ldots + w_k, A_k) = w_1, P(A_1) + \ldots + w_k, P(A_k) \]
8. If \( A_1, \ldots, A_k \) are some facts, predicates, or rules, respectively, the Cartesian Product of \( A_1, \ldots, A_k \) is denoted by \( A_1 \times A_2 \times \ldots \times A_k \) and is defined by
   \[ P(A_1 \times A_2 \times \ldots \times A_k) = \min(P(A_1), P(A_2), \ldots, P(A_k)) \]

Next, we formulate rules to specify which kinds of OCF to apply according to each operation occurred in query processing.

(Rule1) The C.F resulting from combining C.Fs of objects in intra-class condition is obtained by
1. choosing minimum value of C.Fs in case of 'AND'
2. choosing maximum value of C.Fs in case of 'OR'
3. choosing complement value of C.Fs in case of 'NOT'
4. choosing minimum value of C.Fs in case of arithmetic \( +, -, \cdot, \div \)

(Rule2) The C.F resulting from combining C.Fs of object in inter-class condition is obtained by
1. choosing minimum value of C.Fs in case of 'association'
2. choosing minimum value of resultant C.F of objects and complement of C.F of association in case of 'not association'

(Rule4) The C.F of object in new subdatabase or derived attribute can be obtained by multiplying the C.F of original value into the resultant C.F of (Rule1) and (Rule3). This rule is derived from item 4 of OCF for the case of product.

In case of processing production rule described in Section 3, computation of C.F is same as above except the following: since the syntax of rule is similar to that of query, C.Fs in production rule have one additional type, i.e., C.F of rule itself. There might be a conflict to apply rules, i.e., two or more rules are applied to obtain values of the same derived attribute or the same subdatabase. And so, the following two rules are added to compute C.F in processing rules.

(Rule5) C.F of rule itself is multiplied into the resultant C.F of newly derived attribute value or new subdatabase value.

This rule is also derived from item 4 of OCF for the case of product.
(Rule 6) If the resultant objects for differently applied rules are the same, we choose the average value of C.F for resultant objects. If the resultant objects for differently applied rules are different from each other, we choose the maximum value of C.Fs for resultant objects since we use the resultant object which has the maximum value of C.F.

4.3 Sample Query Processing

In this section, we show what kind of answer can be generated and how to compute resultant C.F. We only concern about with respect to computing C.Fs and answer of C.F for resultant objects. If the resultant objects have the maximum value of C.F.

Ex1) Find the name of student whose advisor's major is not '0.S'.

This query can be translated into OQL as follows:

```oql
CONTEXT FACULTY[major <> '0.S'] + ADVISING + STUDENT
SELECT STUDENT[name]
DISPLAY
```

In Figure 3, since the value of major with OID='01' is '0.S(0.7)' in FACULTY G-relation, the C.F that his/her major is not '0.S' is 0.3(i.e. 1 - 0.7) by (Rule 1), and since this entity object is associated with the STUDENT object whose OID is '11' with C.F = 1 in ADVISING in Figure 3, and the resultant C.F with combining these two C.Fs is 0.3(i.e. 1 x MIN(1,0.3)) by (Rule 3). Since the name of OID='11' is 'MAXWELL' with C.F=1, the C.F of derived attribute is 0.3(i.e. 1 x MIN(1,0.3)) by (Rule 4). And so, the first answer is 'MAXWELL(0.3)'.

The next FACULTY's major is not '0.S' with C.F=1, and with same procedure as the first answer, the second answer is 'LEONE'. The C.F that the value of major with OID='03' is not '0.S' is 1.0, and since this object is associated with OID = 13 V 14 of STUDENT(i.e. the C.F of association is 0.5 respectively), the third answer is 'DEBOIS V SMITH' which is the same meaning as 'DEBOIS(0.5) V SMITH(0.5)'.

Next example shows how to compute C.F of the resultant object when it needs to apply rule to get the specified attribute value.

Ex2) Find the name and the desired occupation of each student.

This query can be translated into OQL as follows:

```oql
CONTEXT STUDENT
SELECT STUDENT[name], STUDENT[st-occp]
DISPLAY
```

The name of first student is 'MAXWELL' from Figure 1 and Figure 3. We apply R3 and R4 to get the value of st-occp in STUDENT. We find that C.F in rule itself is 0.7 from R3 in Figure 4 and 'MAXWELL''s advisor is 'DOLL' with C.F = 1 from Figure 3. Since neither FACULTY DOLL's major is 'D.B' nor his/her interest area is 'D.C', it does not satisfy the condition clause of R3 completely. If we substitute this value to the block diagram of R3 in Figure 5(a), we find that STUDENT MAXWELL's desired occupation is 'S.A(0.0)' to apply R4 in Figure 4, we substitute C.F in rule itself(i.e 0.6) to the block diagram of R4 in Figure 5(a). MAXWELL's advisor is found to be 'DOLL' with C.F = 1 as above. Since FACULTY DOLL's major is '0.S' with C.F = 0.7, if we substitute these values to the block diagram of R4 in Figure 5(a), STUDENT MAXWELL's desired occupation is 'C.E' with C.F = 0.42 by (Rule 5). If we choose the maximum value of C.F from S.A(0.0) and C.E(0.42) as resultant values, STUDENT MAXWELL's desired occupation is C.E(0.42). Figure 6(a) shows this.

To get the next answer, we search the next STUDENT whose name is 'LEONE', and whose major is 'RICHARD' in Figure 3. We apply R3 and R4 to get the STUDENT's desired occupation. We substitute each C.F to the block of R3 in Figure 5(a), i.e. C.F in rule itself is 0.7, C.F of association between 'LEONE' and 'RICHARD' is 1, and FACULTY RICHARD's major is 'D.B' with C.F = 1 and his/her interest area is 'D.C' with C.F = 0.9. But, we have to apply R1 and R2 to decide whether STUDENT LEONE's interest area is 'D.C' or not with relationship structure in Figure 5(b). Related C.Fs are filled out in the block R1 of Figure 5(a). Resultantly, we find that the C.F for STUDENT LEONE's interest area to be 'D.C' is 0.54. R2 is executed with the same procedure as above, and the C.F for STUDENT LEONE's interest area to be 'D.C' is 0.54. From the result of applying R1 and R2, we obtain 0.594 as the average C.F, i.e. 0.594, which is substituted into the block of R3 in Figure 5(a). The resultant C.F of R3 is 0.4158. So, LEONE's desired occupation is S.A(0.4158) by rule R3. Next, we apply R4, and resultantly, the C.F for LEONE's desired occupation to be 'C.E' is 0, since it does not satisfy the condition clause of R4. We choose the answer from the attribute value which have the maximum value of C.F, i.e. 'S.A(0.42)'. Therefore, the second answer is [LEONE, S.A(0.42)]. Figure 6(b) shows this. Similar procedure is applied to find the third and last answer.

5. Summary & Further Research

In this paper, we have extended the concepts of the object to include indefinite information with C.F so that indefiniteness can be measured. Objects have been classified into three, i.e. scalar value, rule, and operation, and have classified indefinite information with C.F occurred in 0-0 databases into two categories, i.e. object C.F and association C.F. We have represented those three types of objects with two categories of indefiniteness with C.F. In addition, we have represented the set types of complex domain class with indefinite information with C.F. Various types of answers which can be generated with respect to indefiniteness with C.F were defined, and C.F
Computation rules based on Fuzzy Set theory were suggested to compute resultant C.F of answer in query processing.

As further researches, insertion, deletion, and update operations in OQL should be completely defined, and we should precisely determine and justify the strategy about what happens when these operations occur in consideration with indefiniteness with C.F. Computation rules to determine the resultant C.F of answer should be precisely defined and justified with objective point of view for all persons to accept using Fuzzy Set theory or any other statistics. Integration of object-oriented database and numerical interval which is a part of incomplete information should be considered.

**Figure 1. Sample University Schema**

**Figure 2. Extensional diagram of subschema of Figure 1.**

**Figure 3. G-relation of B-class shown in Figure 1.**

**Figure 4. Sample rules referred in Figure 3.**

**Figure 5. C.F relationship structure in rule.**

**R1(0.6):** IF CONTEXT FACULTY * ADVISING * STUDENT THEN STUDENT[st-inst(Y)] = FACULTY[fa-int(Z)]

**R2(0.8):** IF CONTEXT FACULTY[major = 'D.B(X)' and fa-int = 'D.C(Y)'] * ADVISING * STUDENT THEN STUDENT[st-inst(U)] = FACULTY[fa-int(Z)]

**R3(0.7):** IF CONTEXT FACULTY[major = 'D.B(X)' or fa-int = 'D.C(Y)'] * ADVISING * STUDENT [st-inst = 'D.C(Z)'] THEN STUDENT[st-occp = 'S.A(U)']

**R4(0.6):** IF CONTEXT FACULTY[major = 'O.S(X)'] * ADVISING * STUDENT THEN STUDENT[st-occp = 'C.E(Y)']

TRIGGER(BEFORE DISPLAY STUDENT[st-inst])
EXECUTE R1, R2

TRIGGER(BEFORE DISPLAY STUDENT[st-occp])
EXECUTE R3, R4

TRIGGER(BEFORE EXECUTE R3)
EXECUTE R1, R2
EX) Find the name and the desired occupation of each student

CONTEXT STUDENT
SELECT STUDENT[name], STUDENT[des-occ]
DISPLAY

1) MAXWELL, C.E(0.42)

2) LEONE, S.A(0.42)

3) SMITH, unknown

4) SMITH, unknown

5) DIBOS, unknown

Figure 6. Sample query processing

REFERENCES


