Categorical Description of Dynamic Fuzzy Logic Programming Language*

Xiaofen Han
School of Computer Science and Technology, Soochow University, 215006 ,Suzhou ,China
064227064004@suda.edu.cn

Fanzhang Li
School of Computer Science and Technology, Soochow University, 215006,Suzhou, China
lfzh@suda.edu.cn

***Abstract*—First, this paper gives the relative basic theory of dynamic fuzzy logic, and introduces the basic data types of dynamic fuzzy logic. Then we introduce the basic data types of dynamic fuzzy logic and the abstract syntax of dynamic fuzzy logic programming language. At last, this paper gives the categorical description of dynamic fuzzy logic programming language according to categorical theory, and some examples are also given to demonstrate the theory put forward in the paper. The achievement of this paper will lay the foundation for the further development of dynamic fuzzy logic programming language.**

***Index Terms*—dynamic fuzzy logic; dynamic fuzzy logic programming language; category;**

I. INTRODUCTION

A program has two aspects: syntax and semantics. Syntax is the program structure and the constitutive rules of the program. Semantics focuses on the explanation of syntax domain. That is to say, semantics is the meanings of program. We should know the meaning of each component in order to use the programming language accurately and effectively. Moreover, the effect after executing the component by computer should be consistent with its meaning. Formal semantics can meet the demand for accurate explanations.

There are four major approaches to provide a formal semantics of programming languages: operational semantics, denotational semantics, algebra semantics, and axiomatic semantics[1,2].

The basic principle of operational semantics is simulating the execution of the program by an abstract machine such as automata. Operational semantics loosely corresponds to interpretation, although again the "implementation language" of the interpreter is generally a mathematical formalism. Operational semantics may define an abstract machine, and give meaning to phrases by describing the transitions they induce on states of the machine. Alternatively, as with the pure lambda calculus, operational semantics can be defined via syntactic transformations on phrases of the language itself.

Denotational semantics formalizes the semantics of computer systems by constructing mathematical objects (namely denotations or meanings) which express the semantics of these systems. Denotational semantics interprets the denotation (meaning) of a computer program as a function that maps input domain into output domain. The input domain and the output domain are all called domain for short. Denotational semantics loosely corresponds to compilation, although the "target language" is usually a mathematical formalism rather than another computer language. For example, denotational semantics of functional languages often translates the language into domain theory.

The idea of axiomatic semantics is: given a certain method first, we can prove a property whether it is true under the given premise. Axiomatic semantics makes no distinction between a phrase's meaning and the logical formulas that describe it; its meaning is exactly what can be proven about it in some logic. The canonical example of axiomatic semantics is Hoare logic.

Algebraic semantics of a programming language is a form of axiomatic semantics based on algebraic laws for describing and reasoning about program semantics in a formal manner. The idea of algebraic semantics is combining the logic system which describes the semantics and all the models together. These models satisfy the logic system. In the mean time, the set of the models is considered as an algebraic system so that we can study the relation between them.

The basic theory of algebraic semantics includes categorical theory, diagram category, type algebra theory and so on. Category is put forward by Eilenberg and Maclane in 1945, and it is the further abstraction of abstract structures such as group, loop, domain etc. It focuses on the relation between objects rather than the inside structure of objects. Many concepts in formal methods can be explained more distinctly in categorical theory, so categorical theory is considered as a powerful mathematical tool in computer science. It is used in functional programming language, polymorphic type systems, concurrency models etc. Algebraic semantics has been developed from abstract data type(ADT).ADT is a very important concept in programming language, because it combines data objects with the operations on them and it changes the viewpoint that just regards the data type as

* The research of this paper is supported by National Nature Science Foundation under Grant No 60775045

a set of some objects. Therefore, ADT make programming language process the data under the uniform framework of value set and operations, and ADT can distinguish the character of data types from their implementation details.

Based on the fuzzy sets put forward in 1965 and fuzzy logic put forward in 1973 by L.A.Zadeh, Fanzhang Li professor put forward the concept of dynamic fuzzy sets(DFS) and dynamic fuzzy logic(DFL)[3,4], then he gave the basic problems to design a DFL programming language[5,6], and gave a framework of this language which was developed and perfected in[7]. This paper continues to do the research on this subject and tries to give the categorical description of DFL programming language.

The arrangement of this paper is as follows: section II includes dynamic fuzzy data theory, the basic data types of DFL programming language and the abstract syntax of it; section III gives the dynamic fuzzy abstract data type and the categorical description of DFL programming language. Some examples are also given to explain and analyze the definitions and theories put forward in this paper; section IV concludes the overall presentation.

II. INTRODUCTION OF DYNAMIC FUZZY LOGIC

A. Dynamic fuzzy data theory

In order to introduce the dynamic fuzzy character we need a special set D=[0,1]×[←,→], where "←"denotes good or advance direction of dynamic change and "→"indicates bad or back direction of dynamic change.

Pact 1 The character of the data that is "dynamic" and "fuzzy" is called Character of Dynamic Fuzzy.

Pact 2 The data which holds character of dynamic and fuzzy are called dynamic fuzzy data.

The set of dynamic fuzzy data is called dynamic fuzzy set , and it is defined as follows:

Definition 1 Let a mapping be defined over domain U

$$(\overleftarrow{A},\overrightarrow{A}):(\overleftarrow{U},\overrightarrow{U})\to[0,1]\times[\leftarrow,\rightarrow],(\overleftarrow{u},\overrightarrow{u})\mapsto(\overleftarrow{A}(\overleftarrow{u}),\overrightarrow{A}(\overrightarrow{u}))$$

written as $(\overleftarrow{A},\overrightarrow{A})=\overleftrightarrow{A}$ or $\overleftrightarrow{A}$, then we call $(\overleftarrow{A},\overrightarrow{A})$ the Dynamic Fuzzy Set of $(\overleftarrow{U},\overrightarrow{U})$, DFS is short for Dynamic Fuzzy Set. Moreover, $(\overleftarrow{A}(\overleftarrow{u}),\overrightarrow{A}(\overrightarrow{u}))$ is defined as the membership degree of the membership function to $(\overleftarrow{A},\overrightarrow{A})$.

Definition 2 A declarative sentence which holds dynamic fuzzy character is called dynamic fuzzy (DF) proposition. It is usually denoted by capital letters A, B, C, etc. Generally speaking, a dynamic fuzzy proposition can not be absolutely true or false. We just consider its degree of truth or falseness concerning the character of dynamic fuzzy, which is called dynamic fuzzy true or false degree.

Definition 3 A dynamic fuzzy proposition logic truth or falseness can be defined by dynamic fuzzy number $(\overleftarrow{a},\overrightarrow{a})\in[0,1]\times[\leftarrow,\rightarrow]$. It is dynamic fuzzy true or false degree of a proposition. It is usually denoted by letters such as $(\overleftarrow{a},\overrightarrow{a})$, $(\overleftarrow{b},\overrightarrow{b})$, $(\overleftarrow{c},\overrightarrow{c})$ and so on.

In logic system the calculus of proposition is just the true of false calculus. In this paper we consider that the dynamic fuzzy proposition equals to its true or false value. We have some definitions as follows.

Definition 4 A dynamic fuzzy proposition can be regarded as a variable valued on interval $[0,1]\times[\leftarrow,\rightarrow]$. The variable is dynamic fuzzy variable, which is usually denoted by lowercase letters.

The operations of dynamic fuzzy variable $(\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{y},\overrightarrow{y})\in[0,1]\times[\leftarrow,\rightarrow]$ are defined as follows.

Note: $(\overleftarrow{x},\overrightarrow{x})=\overleftarrow{x}$ or $\overrightarrow{x}$, max $(\overleftarrow{x},\overrightarrow{x})\overset{\Delta}{=}\overrightarrow{x}$, min $(\overleftarrow{x},\overrightarrow{x})\overset{\Delta}{=}\overleftarrow{x}$

(1) Negation"—".

For example, express Negation of $(\overleftarrow{x},\overrightarrow{x})$ as $\overline{(\overleftarrow{x},\overrightarrow{x})}$ and we can obtain that $\overline{(\overleftarrow{x},\overrightarrow{x})}\overset{\Delta}{=}((\overleftarrow{1}-\overleftarrow{x}),(\overrightarrow{1}-\overrightarrow{x}))$;

(2)Disjunction""∨".

For example, $(\overleftarrow{x},\overrightarrow{x})\vee(\overleftarrow{y},\overrightarrow{y})\overset{\Delta}{=}\max((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{y},\overrightarrow{y}))$;

(3)Conjunction "∧".

For example, $(\overleftarrow{x},\overrightarrow{x})\wedge(\overleftarrow{y},\overrightarrow{y})\overset{\Delta}{=}\min((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{y},\overrightarrow{y}))$;

(4)Condition "→".

For example,

$$(\overleftarrow{x},\overrightarrow{x})\to(\overleftarrow{y},\overrightarrow{y})\Leftrightarrow(\overleftarrow{x},\overrightarrow{x})\vee(\overleftarrow{y},\overrightarrow{y})\overset{\Delta}{=}\max((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{y},\overrightarrow{y}))$$;

(5)Bi-condition "↔".

For example,

$$\begin{aligned}&(\overleftarrow{x},\overrightarrow{x})\leftrightarrow(\overleftarrow{y},\overrightarrow{y})\\&\Leftrightarrow((\overleftarrow{x},\overrightarrow{x})\to(\overleftarrow{y},\overrightarrow{y}))\wedge((\overleftarrow{y},\overrightarrow{y})\to(\overleftarrow{x},\overrightarrow{x}))\\&\Leftrightarrow((\overline{(\overleftarrow{x},\overrightarrow{x})}\vee(\overleftarrow{y},\overrightarrow{y}))\wedge((\overline{(\overleftarrow{y},\overrightarrow{y})}\vee(\overleftarrow{x},\overrightarrow{x}))\\&\Leftrightarrow\min(\max((\overline{(\overleftarrow{x},\overrightarrow{x})},(\overleftarrow{y},\overrightarrow{y})),\max(\overline{(\overleftarrow{y},\overrightarrow{y})},(\overleftarrow{x},\overrightarrow{x})))\end{aligned}$$

Let $(\overleftarrow{x},\overrightarrow{x})P$ and $(\overleftarrow{y},\overrightarrow{y})Q$ be two DYNAMIC FUZZY propositions. Then $\overline{(\overleftarrow{x},\overrightarrow{x})P}$, $(\overleftarrow{x},\overrightarrow{x})P\vee(\overleftarrow{y},\overrightarrow{y})Q$ and $(\overleftarrow{x},\overrightarrow{x})P\to(\overleftarrow{y},\overrightarrow{y})Q$ are dynamic fuzzy propositions.

Definition 5 Dynamic fuzzy proposition logic calculus formula is defined as follows:

(1) A single dynamic fuzzy proposition variable is a well-formed formula.

(2) If $(\overleftarrow{x},\overrightarrow{x})P$ is a well-formed formula, then $\overline{(\overleftarrow{x},\overrightarrow{x})P}$ is also a well-formed formula.

(3) If $(\overleftarrow{x},\overrightarrow{x})P$ and $(\overleftarrow{y},\overrightarrow{y})Q$ are well-formed formulas, then $(\overleftarrow{x},\overrightarrow{x})P\vee(\overleftarrow{y},\overrightarrow{y})Q$, $(\overleftarrow{x},\overrightarrow{x})P\wedge(\overleftarrow{y},\overrightarrow{y})Q$, $(\overleftarrow{x},\overrightarrow{x})P\to(\overleftarrow{y},\overrightarrow{y})Q$ and $(\overleftarrow{x},\overrightarrow{x})P\leftrightarrow(\overleftarrow{y},\overrightarrow{y})Q$ are all well-formed formulas.

(4) Symbol strings connected by proposition variable connections and embraces in (1), (2) and (3) for finite times.

Principal DFL formulas are as follows:

(1) Idempotent Law

$$(\overleftarrow{x},\overrightarrow{x})A\vee(\overleftarrow{x},\overrightarrow{x})A=(\overleftarrow{x},\overrightarrow{x})A$$

$$(\overleftarrow{x},\overrightarrow{x})A\wedge(\overleftarrow{x},\overrightarrow{x})A=(\overleftarrow{x},\overrightarrow{x})A$$

(2) Law of Commutation

$$(\overleftarrow{x},\overrightarrow{x})A\vee(\overleftarrow{y},\overrightarrow{y})B=(\overleftarrow{y},\overrightarrow{y})B\vee(\overleftarrow{x},\overrightarrow{x})A$$

$$(\overleftarrow{x},\overrightarrow{x})A \wedge (\overleftarrow{y},\overrightarrow{y})B = (\overleftarrow{y},\overrightarrow{y})B \wedge (\overleftarrow{x},\overrightarrow{x})A$$

(3) Law of association

$$(\overleftarrow{x},\overrightarrow{x})A \vee ((\overleftarrow{y},\overrightarrow{y})B \vee (\overleftarrow{c},\overrightarrow{c})C) = ((\overleftarrow{x},\overrightarrow{x})A \vee (\overleftarrow{y},\overrightarrow{y})B) \vee (\overleftarrow{c},\overrightarrow{c})C$$

$$(\overleftarrow{x},\overrightarrow{x})A \wedge ((\overleftarrow{y},\overrightarrow{y})B \wedge (\overleftarrow{c},\overrightarrow{c})C) = ((\overleftarrow{x},\overrightarrow{x})A \wedge (\overleftarrow{y},\overrightarrow{y})B) \wedge (\overleftarrow{c},\overrightarrow{c})C$$

(4) Absorption Law

$$(\overleftarrow{x},\overrightarrow{x})A \vee ((\overleftarrow{y},\overrightarrow{y})B \wedge (\overleftarrow{x},\overrightarrow{x})A) = (\overleftarrow{x},\overrightarrow{x})A$$

$$(\overleftarrow{x},\overrightarrow{x})A \wedge ((\overleftarrow{y},\overrightarrow{y})B \vee (\overleftarrow{x},\overrightarrow{x})A) = (\overleftarrow{x},\overrightarrow{x})A$$

(5) De Morgan rule

$$\overline{(\overleftarrow{x},\overrightarrow{x})A \wedge (\overleftarrow{y},\overrightarrow{y})B} = \overline{(\overleftarrow{x},\overrightarrow{x})A} \vee \overline{(\overleftarrow{y},\overrightarrow{y})B}$$

$$\overline{(\overleftarrow{x},\overrightarrow{x})A \vee (\overleftarrow{y},\overrightarrow{y})B} = \overline{(\overleftarrow{x},\overrightarrow{x})A} \wedge \overline{(\overleftarrow{y},\overrightarrow{y})B}$$

(6) Operational rule of Constant

$$\mathrm{A} \vee (\overleftarrow{x},\overrightarrow{x})A = \mathrm{A}$$

$$A \wedge (\overleftarrow{x},\overrightarrow{x})A = (\overleftarrow{x},\overrightarrow{x})A$$

B. Basic data types of DFL programming language

A data type is defined as a data set and its operation sets holding certain attributes. According to the definition, the basic data types of DFL programming language are DFS and some operation sets on DFS. From the definition of DFS we can see that the basic principium of DFS is a mapping from a denotable set to $D=[0,1]\times[\leftarrow,\rightarrow]$. Thus the basic data types of DFL programming language are mappings from the basic data types of advanced programming languages to D. They can be denoted as follows in detail [8]:

(1) The dynamic fuzzy integer data type(DFInt) is represented as $((\overleftarrow{i},\overrightarrow{i}),(\overleftarrow{d},\overrightarrow{d}))$,where $i \in$ Integer , $d \in D$.

(2) The dynamic fuzzy real data type (DFReal) is represented as $((\overleftarrow{r},\overrightarrow{r},),(\overleftarrow{d},\overrightarrow{d}))$, where $r \in$ Real, $d \in D$.

(3) The dynamic fuzzy char data type (DFChar) is represented as $((\overleftarrow{c},\overrightarrow{c}),(\overleftarrow{d},\overrightarrow{d}))$, where $c \in$ Char, $d \in D$.

(4) The dynamic fuzzy Boolean data type (DFBool) is represented as $((\overleftarrow{b},\overrightarrow{b}),(\overleftarrow{d},\overrightarrow{d}))$, where $b \in$ Boolean, $d \in D$.

Here, Integer is the classical integer data set; Real is the classical real data set; Char is the classical character data set and Boolean is the classical Boolean data set.

We will not focus on the operations of dynamic fuzzy data types, so the readers can get more information from references[9-11].

C. Abstract syntax of DFL programming language

The syntactic structure of dynamic fuzzy logic programming language given in this paper is consistent with reference [7], so the programming language designed in this paper is as same as the common programming language in principle. It is composed by assigned statements, conditional statements, repetitive statements, input statements and output statements and so on. But they are a bit different in the form and content. The important difference between DLF programming language and the classical programming language is that DLF programming language can deal with the dynamic fuzzy data. In order to do this, we can follow the guarded commands proposed by Dijkstra, whose characteristic is introducing nondeterminism to a program. Each statement executed possibly is provided with a guarded condition.

In this part we give the syntactic structure of the DFL programming language following the guarded commands. In order to introduce dynamic fuzzy sets to the program, we need the special set $D=[0,1]\times[\leftarrow,\rightarrow]$. For each variable of the program, there exists an element belonging to D to denote its membership degree. We only provide the abstract syntax of our language for convenience.

p::=d ; s

d::=I:t|d';d''

t::= DFInt| DFReal| DFChar| DFBool

s::= skip|abort|ide:=e|(s; s)|if g fi|do g od

e::= -e| $\left(\left(\overleftarrow{n},\overrightarrow{n}\right),\left(\overleftarrow{d},\overrightarrow{d}\right)\right)$ |e0 op e1|(e) | $(\overleftarrow{x},\overrightarrow{x})$ | DFInt|DFReal

op::=+|-|*|/|%

g::=b→s|(g0□g1)

b::= $((\overleftarrow{true},\overrightarrow{true}),(\overleftarrow{d},\overrightarrow{d}))\,|\,((\overleftarrow{false},\overrightarrow{false}),(\overleftarrow{d},\overrightarrow{d}))$ |b0 Bop b1|e0 Rel e1

Bop::=&|||!=

Rel ::==|>|<|>=|<=|!=

Here, "abort" denotes that it can not arrive to the terminate state from any initial state.

$((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d})) \in Var$ is a variable, where

$$Var = \{((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d}))\,|\,(\overleftarrow{x},\overrightarrow{x}) \in \mathrm{DFInt}\} \cup \{((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d}))\,|\,(\overleftarrow{x},\overrightarrow{x}) \in \mathrm{DFReal}\};$$

$(\overleftarrow{n},\overrightarrow{n})$ is a constant; $(\overleftarrow{d},\overrightarrow{d}) \in D$,and it demotes the membership degree;

III. CATEGORICAL DESCRIPTION OF DFL PROGRAMMING LANGUAGE

A. Dynamic fuzzy abstract data type

An abstract data type is an important concept in algorithm design and program design. It is a set of data values and associated operations that are precisely specified independent of any particular implementation. So, the operations on a data type depend on the concrete representation of the data type. On the one hand, the operations on a data model use the data variables as operation objects, or operation result, or both of them. On the other hand, if the concrete representation of the data type is determined and the operations have been implemented, then operation efficiency is determined[2，12].

Definition 6 A dynamic fuzzy abstract data type can be represented as $A=(S,\Sigma,E)$, where:

S is the limited set of data types;

Σ is the limited set of the objects over S;

E is the limited set of axioms.

It can be seen that a dynamic fuzzy abstract data type consists of type set (the basic syntax components), operation set (the combination relation between the operation objects, namely the operations on the data types) and the limited set of axioms. From the analysis of DFL, we know that every dynamic fuzzy variable has a degree of dynamic fuzzy and the value of the degree will be modified during the execution of the program. The modifying principle usually meets T modular and S modular operation [4].

Example 1

Let $ADT = (S, \Sigma, E)$, where

$S = \{DFInt, DFBool\}$, $\Sigma = \{+, -, \times, /, =, >\}$:

sort DFInt,DFBool

operations

$+$: DFInt,DFInt, $\rightarrow$ DFInt

$-$: DFInt,DFInt, $\rightarrow$ DFInt

$\times$: DFInt,DFInt, $\rightarrow$ DFInt

$/$: DFInt,DFInt, $\rightarrow$ DFInt

=:DFInt,DFInt $\rightarrow$ *DFBool*

>:DFInt,DFInt $\rightarrow$ *DFBool*

axioms for all

$((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})),((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) \in DFInt$,

$((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) = ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})) \bullet ((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y}))$

with:

(1) $(\overleftarrow{z},\overrightarrow{z}) = (\overleftarrow{x},\overrightarrow{x}) \bullet (\overleftarrow{y},\overrightarrow{y})$

(2) $(\overleftarrow{d_z},\overrightarrow{d_z}) = t((\overleftarrow{d_x},\overrightarrow{d_x}),(\overleftarrow{d_y},\overrightarrow{d_y}))$, where the modifying principle meets certain T modular operation and $\bullet$ can be $+, -, \times, /$ etc.

(3)For

$((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) = f_{=}(((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})))$,

if $(\overleftarrow{x},\overrightarrow{x}) = (\overleftarrow{y},\overrightarrow{y})$,then $(\overleftarrow{z},\overrightarrow{z}) = (\overleftarrow{true},\overrightarrow{true})$,

$(\overleftarrow{d_z},\overrightarrow{d_z}) = t((\overleftarrow{d_x},\overrightarrow{d_x}),(\overleftarrow{d_y},\overrightarrow{d_y}))$.

(4) for

$((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) = f_{>}(((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})))$,

if $(\overleftarrow{x},\overrightarrow{x}) > (\overleftarrow{y},\overrightarrow{y})$,

then $(\overleftarrow{z},\overrightarrow{z}) = (\overleftarrow{true},\overrightarrow{true})$, $(\overleftarrow{d_z},\overrightarrow{d_z}) = t((\overleftarrow{d_x},\overrightarrow{d_x}),(\overleftarrow{d_y},\overrightarrow{d_y}))$.

B. Categorical description of DFL programming language

This paper is based on the dynamic fuzzy logic, so we can consider the execution of the whole algorithm program as the transition between abstract states. The abstract states are objects of a category, and the transition of the states is the morphisms between the objects. The difference between the traditional category and dynamic fuzzy category is that the dynamic fuzzy category has membership function. Therefore, in this section, we focus on the membership function processing of dynamic fuzzy category.

Definition 7 Let $(\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}) \in D\mathrm{F}(U)$, we call $(\overleftarrow{A},\overrightarrow{A}) \cup (\overleftarrow{B},\overrightarrow{B})$, $(\overleftarrow{A},\overrightarrow{A}) \cap (\overleftarrow{B},\overrightarrow{B})$ the union set and intersection set of $(\overleftarrow{A},\overrightarrow{A})$, $(\overleftarrow{B},\overrightarrow{B})$ respectively. $(\overleftarrow{A},\overrightarrow{A})^c$ is the complement set of $(\overleftarrow{A},\overrightarrow{A})$ .Membership functions are:

$$((\overleftarrow{A},\overrightarrow{A}) \cup (\overleftarrow{B},\overrightarrow{B}))(u) = (\overleftarrow{A},\overrightarrow{A})(u) \vee (\overleftarrow{B},\overrightarrow{B})(u)$$
$$\stackrel{\Delta}{=} \max((\overleftarrow{A},\overrightarrow{A})(u),(\overleftarrow{B},\overrightarrow{B})(u))$$
$$((\overleftarrow{A},\overrightarrow{A}) \cap (\overleftarrow{B},\overrightarrow{B}))(u) = (\overleftarrow{A},\overrightarrow{A})(u) \wedge (\overleftarrow{B},\overrightarrow{B})(u)$$
$$\stackrel{\Delta}{=} \min((\overleftarrow{A},\overrightarrow{A})(u),(\overleftarrow{B},\overrightarrow{B})(u))$$
$$(\overleftarrow{A},\overrightarrow{A})^c(u) = 1 - (\overleftarrow{A},\overrightarrow{A})(u)$$
$$\stackrel{\Delta}{=} (\overleftarrow{1} - \overleftarrow{A}(\overleftarrow{u}), \overrightarrow{1} - \overrightarrow{A}(\overrightarrow{u}))$$

（Here $u \stackrel{DF}{=} (\overleftarrow{u},\overrightarrow{u})$ ）

Definition 8 Let S=$\{s_i | i \in I\}$ be a finite set, where I is a finite set of subscript, that is to say I is a set of integers. s_i is a dynamic fuzzy data type. $O = \{O_j \mid j \in J\}$ is a finite set, where J is a finite set of subscript, and O_j is a operation based on dynamic fuzzy data. Therefore, operation O_j can be represented as:

$$O_j : s_1 \times s_2 \times \cdots \times s_k \rightarrow s_{k+1} \qquad (1)$$

Here, $(s_1, s_2, \cdots, s_k)$ is the rank of O_j, and it is called full rank together with the result s_{k+1}.The type of each operation is described by its full rank. Each s_i is independent, and they can be the same in full or partially, or they are different.

The ordered pair $\mathrm{DF}\Sigma =< S, O >$ is called a dynamic fuzzy Σ – algebra ($\mathrm{DF}\Sigma$ for short).

Definition 9 Suppose that $\mathrm{DF}\Sigma_1 =< S_1, O_1 >$ and $\mathrm{DF}\Sigma_2 =< S_2, O_2 >$ are two $\mathrm{DF}\Sigma$ .The morphism σ from $\mathrm{DF}\Sigma_1$ to $\mathrm{DF}\Sigma_2$ is (f, g) ,where f is a morphism from S_1 to S_2 ,and g is a morphism from O_1 to O_2 .

The definition of category is based on the traditional set and class, while the objects we should process are dynamic and fuzzy, so a dynamic fuzzy category can be constructed by modifying the definition of the traditional category[2,13,14]:

Definition 10 A dynamic fuzzy category

$\mathrm{DF}C = \{objDFC, MorDFC, dom, cod, \circ\}$,

here:

（1）Object elements: *objDFC* is the dynamic fuzzy class of all the object elements and the object elements are a dynamic fuzzy data type;

（2）Morphism elements: *MorDFC* is the class of all the dynamic fuzzy morphism elements and the dynamic fuzzy morphism element is morphism between the objects.

（3） dom : $MorDFC \rightarrow objDFC$ is a function. If $f \in MorDFC$, then dom(f) is the domain of f;

（4） cod : $MorDFC \rightarrow objDFC$ is a function. If $f \in MorDFC$, then cod (f) is the codomain of f ;

（5） $\circ$: $MorDFC \times MorDFC \rightarrow MorDFC$ is a function. It is a compound calculation based on DFC.

Besides, the following conditions should be satisfied:

1. Satisfy the associative law: if $f \circ g$ and $h \circ f$ are defined, then $h \circ (f \circ g) = (h \circ f) \circ g$.

2. Existence conditions of identical morphism: for arbitrary object element $(\overleftarrow{A},\overrightarrow{A})$, there only exits one morphism element $l_A \in MorDFC$ which satisfies the following conditions:

（1） $dom(l_A) = cod(l_A) = (\overleftarrow{A},\overrightarrow{A})$.

（2） if $dom(f) = (\overleftarrow{A},\overrightarrow{A})$,then $f \circ l_A = f$.

（3） if $cod(f) = (\overleftarrow{B},\overrightarrow{B})$,then $l_B \circ f = f$.

3. For arbitrary ordered pair $((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B})) \in MorDFC$,

$\mathrm{Hom}((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}))$ represents the class of all the dynamic fuzzy morphism elements from $(\overleftarrow{A},\overrightarrow{A})$ to $(\overleftarrow{B},\overrightarrow{B})$.

Theorem 1 All the dynamic fuzzy sets and the dynamic fuzzy morphisms between the sets construct a dynamic fuzzy category **DFSet**.

Proof.

(1) The objects of **DFSet** are dynamic fuzzy sets.

(2) The morphisms between objects are the morphisms between dynamic fuzzy sets.

(3) We can validate the combinability of the morphisms according to the combinability of mappings between sets.

(4) The identical mapping from a set to itself is the identical morphism from an object to itself.

Theorem 2 We take all the dynamic fuzzy data types as objects, and take the morphism defined in definition 7 as the dynamic fuzzy morphism between objects, then the entire of dynamic fuzzy data types and their morphism can construct a dynamic fuzzy category DFSig .

Proof.

From definition 10, it can be seen that the proof of dynamic fuzzy category DFSig should satisfy the following conditions:

（1）Satisfy associative law;

（2）The existence of identical morphism;

Obviously, the identical morphism exits and the compound calculation result of morphism is still morphism, because:

If $\sigma_1 \circ \sigma_2$ and $\sigma_2 \circ \sigma_3$ are defined, then

$$\begin{aligned}\sigma_1 \circ (\sigma_2 \circ \sigma_3) &= (f_1, g_1) \circ ((f_2, g_2) \circ (f_3, g_3)) \\ &= (f_1, g_1) \circ (f_2 \circ f_3, g_2 \circ g_3) \\ &= (f_1 \circ f_2 \circ f_3, g_1 \circ g_2 \circ g_3) \\ &= ((f_1 \circ f_2) \circ f_3, (g_1 \circ g_2) \circ g_3) \\ &= (\sigma_1 \circ \sigma_2) \circ \sigma_3\end{aligned}$$

So, the associative law is satisfied. The theorem is proved.

According to the traditional notation, the following notations are equivalent:

$f \in DFC((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}))$, $(\overleftarrow{A},\overrightarrow{A}) \xrightarrow{f} (\overleftarrow{B},\overrightarrow{B})$, $f : (\overleftarrow{A},\overrightarrow{A}) \to (\overleftarrow{B},\overrightarrow{B})$ .Therefore we can know that the dynamic fuzzy morphism element can be considered as the ordered pair of object elements. For ordered pair $< (\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}) >$, obviously,

$dom(< (\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}) >) = (\overleftarrow{A},\overrightarrow{A})$,

$cod(< (\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}) >) = (\overleftarrow{B},\overrightarrow{B})$,

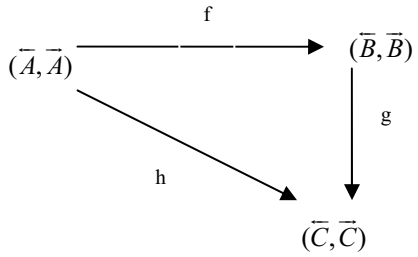
$< (\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}) > \circ < (\overleftarrow{C},\overrightarrow{C}),(\overleftarrow{A},\overrightarrow{A}) >$

$=< (\overleftarrow{C},\overrightarrow{C}),(\overleftarrow{B},\overrightarrow{B}) >$.

Commute, the character of category, applies to the dynamic fuzzy category. Therefore,

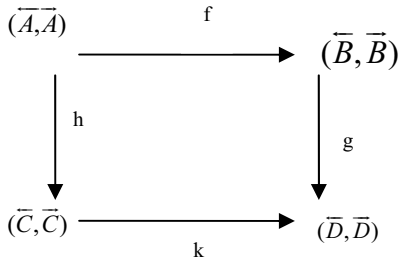
$h = g \circ f$,

or $(\overleftarrow{A},\overrightarrow{A}) \xrightarrow{f} (\overleftarrow{C},\overrightarrow{C}) = (\overleftarrow{A},\overrightarrow{A}) \xrightarrow{f} (\overleftarrow{B},\overrightarrow{B}) \xrightarrow{g} (\overleftarrow{C},\overrightarrow{C})$ .we can see it in graph 1:



Graph 1

While the commute of the graph 2 is embodied in $g \circ f = k \circ h$:



Graph 2

Here, a dynamic fuzzy morphism is the morphism between objects of dynamic fuzzy data types. We call it dynamic fuzzy morphism for short in order to distinguish it from the morphism based on traditional sets and classes.

Example 2

Let $\left((\overleftarrow{G},\overrightarrow{G}),\prec\right)$ is a partial ordering set, $\prec$ is the partial relation over $(\overleftarrow{G},\overrightarrow{G})$. According to the definition of dynamic fuzzy category, we can construct a dynamic fuzzy category $DFC = \left((\overleftarrow{G},\overrightarrow{G}),\prec,dom,cod,\circ\right)$, where $(\overleftarrow{G},\overrightarrow{G})$ is a class of all the object elements, and $\prec$ is the dynamic fuzzy morphism of DFC . Let $((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d}))$, $((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d},\overrightarrow{d}))$ and $((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d},\overrightarrow{d}))$ be elements of $(\overleftarrow{G},\overrightarrow{G})$

(1) $dom :\prec \to (\overleftarrow{G},\overrightarrow{G})$,

i.e.

$dom(< ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d},\overrightarrow{d})) >) = ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d}))$.

(2) $cod :\prec \to (\overleftarrow{G},\overrightarrow{G})$,

i.e.

$cod(< ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d},\overrightarrow{d})) >)$

$= ((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d},\overrightarrow{d}))$.

(3) $\circ :\prec \times \prec \to \prec$, i.e.

$< ((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d},\overrightarrow{d})),((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d},\overrightarrow{d})) >$

$\circ < ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d},\overrightarrow{d})) >$.

$=< ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d},\overrightarrow{d})),((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d},\overrightarrow{d})) >$

We give two concrete examples as follows.

Example 3

$(\overleftarrow{G},\overrightarrow{G}) = \{((\overleftarrow{i},\overrightarrow{i}),(\overleftarrow{d},\overrightarrow{d})) \mid \text{i} \in \text{Integer}, \ \text{d} \in \text{D}\}$ .That is to say $(\overleftarrow{G},\overrightarrow{G})$ is the class of dynamic fuzzy integer data. $\left(\left(\overleftarrow{a,a}\right),\left(\overleftarrow{d},\overrightarrow{d}\right)\right) \le \left(\left(\overleftarrow{b,b}\right),\left(\overleftarrow{d},\overrightarrow{d}\right)\right)$ denotes $\prec$. Therefore we can get a dynamic fuzzy category.

Example 4

Let $(\overleftarrow{G},\overrightarrow{G}) = \{((\overleftarrow{i},\overrightarrow{i}),(\overleftarrow{d},\overrightarrow{d})) \mid \text{i} \in \text{Integer}, \ \text{d} \in \text{D}\}$. $\prec$ represents that $\left(\left(\overleftarrow{a,a}\right),\left(\overleftarrow{d},\overrightarrow{d}\right)\right)$ can be divided exactly by $\left(\left(\overleftarrow{b,b}\right),\left(\overleftarrow{d},\overrightarrow{d}\right)\right)$, notating $\left(\left(\overleftarrow{a,a}\right),\left(\overleftarrow{d_a},\overrightarrow{d_a}\right)\right)\%\left(\left(\overleftarrow{b,b}\right),\left(\overleftarrow{d_b},\overrightarrow{d_b}\right)\right)=\left(\left(\overleftarrow{0,0}\right),\left(\overleftarrow{d,d}\right)\right)$ .Thus we can also get a dynamic fuzzy category.

Let * be a binary operation over dynamic fuzzy relation, then

$$((\overleftarrow{A},\overrightarrow{A}) * (\overleftarrow{B},\overrightarrow{B}))(\overleftarrow{z},\overrightarrow{z})$$

$$= \bigvee_{(\overleftarrow{z},\overrightarrow{z})=(\overleftarrow{x},\overrightarrow{x})*(\overleftarrow{y},\overrightarrow{y})} ((\overleftarrow{A},\overrightarrow{A})(\overleftarrow{x},\overrightarrow{x}) \wedge (\overleftarrow{B},\overrightarrow{B})(\overleftarrow{y},\overrightarrow{y}))$$

where $(\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}) \in DF(R)$ .Specially, we call it expand addition, expand subtraction, expand multiplication, expand division when $*$ *is* $+,-,\times,/$.

Example 5

Let DFInt represent dynamic fuzzy integer data, and DFReal is dynamic fuzzy real data. $((\overleftarrow{A},\overrightarrow{A}) * (\overleftarrow{B},\overrightarrow{B}))(\overleftarrow{z},\overrightarrow{z})$ is the binary operation between $(\overleftarrow{A},\overrightarrow{A})$ and $(\overleftarrow{B},\overrightarrow{B})$. Then we can obtain a category:

(1)The set of the dynamic fuzzy objects can be notated as: $objC = \{DFInt, DF\,\text{Re}al\}$.

(2)For arbitrary objects $(\overleftarrow{A},\overrightarrow{A})$ and $(\overleftarrow{B},\overrightarrow{B})$, there exits a class $H((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}))$, whose element is $f_*(((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})))$, where $*$ is one of the following operations:

for

$((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})),((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) \in DFInt,$

$((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) = ((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})) * ((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y}))$

with:

① $(\overleftarrow{z},\overrightarrow{z}) = (\overleftarrow{x},\overrightarrow{x}) * (\overleftarrow{y},\overrightarrow{y})$

② $(\overleftarrow{d_z},\overrightarrow{d_z}) = t((\overleftarrow{d_x},\overrightarrow{d_x}),(\overleftarrow{d_y},\overrightarrow{d_y}))$, where the modifying principle meets certain T modular operation and $*$ can be $+,-,\times,/$ etc.

③ For

$((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) = f_=(((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})))$,

if $(\overleftarrow{x},\overrightarrow{x}) = (\overleftarrow{y},\overrightarrow{y})$,then $(\overleftarrow{z},\overrightarrow{z}) = (\overleftarrow{true},\overrightarrow{true})$,

$(\overleftarrow{d_z},\overrightarrow{d_z}) = t((\overleftarrow{d_x},\overrightarrow{d_x}),(\overleftarrow{d_y},\overrightarrow{d_y}))$.

④ For

$((\overleftarrow{z},\overrightarrow{z}),(\overleftarrow{d_z},\overrightarrow{d_z})) = f_>(((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})))$,

if $(\overleftarrow{x},\overrightarrow{x}) \succ (\overleftarrow{y},\overrightarrow{y})$,

then $(\overleftarrow{z},\overrightarrow{z}) = (\overleftarrow{true},\overrightarrow{true})$, $(\overleftarrow{d_z},\overrightarrow{d_z}) = t((\overleftarrow{d_x},\overrightarrow{d_x}),(\overleftarrow{d_y},\overrightarrow{d_y}))$.

(3) Obviously, we can prove that

morphism $f_*(((\overleftarrow{x},\overrightarrow{x}),(\overleftarrow{d_x},\overrightarrow{d_x})),((\overleftarrow{y},\overrightarrow{y}),(\overleftarrow{d_y},\overrightarrow{d_y})))$ satisfies the associative law and compound calculation is shown as follows:

$(\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}),(\overleftarrow{C},\overrightarrow{C}) \in objC,$

$H((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B})) * H((\overleftarrow{B},\overrightarrow{B}),(\overleftarrow{C},\overrightarrow{C}))$

$\to H((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{C},\overrightarrow{C}))$.

Definition 11 Let

$\text{DF}C = \{objDFC, MorDFC, dom, cod, \circ\}$ is a dynamic fuzzy category, then the opposite category can be defined as:

$\text{DF}C^{op} = \{obj\text{DF}C, Mor\text{DF}C, cod, dom, \circ\}$,

From the definition, we can see that:

（1）Object elements: the morphism sets of $\text{DF}C^{op}$ is the same as DFC , i.e., it is the dynamic fuzzy class of all the morphism elements;

（2）Morphism elements: the object sets of $\text{DF}C^{op}$ is the same as DFC , i.e., it is the dynamic fuzzy class of all the object elements;

（3） dom and cod : the dom and cod of $\text{DF}C^{op}$ can be obtained by exchanging the dom and cod of DFC .

Definition 12 If $(\overleftarrow{A},\overrightarrow{A})$ is an object element of DFC , for arbitrary object $(\overleftarrow{B},\overrightarrow{B})$ of DFC ,there only exits one morphism $f : (\overleftarrow{A},\overrightarrow{A}) \to (\overleftarrow{B},\overrightarrow{B})$ (i.e. $H((\overleftarrow{A},\overrightarrow{A}),(\overleftarrow{B},\overrightarrow{B}))$ only has one morphism element), we call $(\overleftarrow{A},\overrightarrow{A})$ is the initial object element of DFC .

Definition 13 If $(\overleftarrow{A},\overrightarrow{A})$ is an object element of DFC , for arbitrary object $(\overleftarrow{B},\overrightarrow{B})$ of DFC , there only exits one morphism $f : (\overleftarrow{B},\overrightarrow{B}) \to (\overleftarrow{A},\overrightarrow{A})$ （i.e. $H((\overleftarrow{B},\overrightarrow{B}),(\overleftarrow{A},\overrightarrow{A}))$ only has one morphism element） , we call $(\overleftarrow{A},\overrightarrow{A})$ is initial ultimate object element of DFC .

Definition 14 Let the dynamic fuzzy morphism of DFC be $f : A \to B$, then:

1. If for each $(\overleftarrow{C},\overrightarrow{C}) \in objC$ and all $r,s \in Hom((\overleftarrow{C},\overrightarrow{C}),(\overleftarrow{A},\overrightarrow{A}))$, there exits $f \circ r = f \circ s$,then f is a dynamic fuzzy constant morphism.

2. If f is a dynamic fuzzy constant morphism of $\mathrm{DF}C^{op}$, then f is a dynamic fuzzy coconstant morphism of $\mathrm{DF}C$.

3. If f is both a dynamic fuzzy constant morphism and a dynamic fuzzy zero morphism, then f is a dynamic fuzzy zero morphism.

Definition 15 Let E be a group of DFΣ equations, then E^* represents the set of all the $\Sigma-$algebra which satisfy E.

Definition 16 Let <DFΣ, E> is a description of abstract data type. We call <DFΣ, E> is a program representation. If axiom group is equation group E, we use E^+ to represent the entire equations derived by E. E^+ is called closed equation group. Therefore, if E is closed, then <DFΣ, E> is called a program, otherwise <DFΣ, E^+> is the program described by <DFΣ, E>.

Because we can construct *DFC* using DFΣ , we can continue the research on DFL programming language based on the categorical description of it. As we have said, we can regard the execution of the whole algorithm program as the transition between abstract states. The abstract states are objects of a category, and the transition of the states is the morphisms between the objects.

IV. CONCLUSIONS AND FUTURE WORK

A. Conclusions

Since Fanzhang Li proposed the DFL to process the dynamic fuzzy problems in 1996, he has obtained a series of achievements and application. But there is few language suitable for the dynamic fuzzy problems which is defined by clear semantic so far. Based on this, this paper gives the Categorical description of DFL programming language and some examples are also given, which could promote the development of DFL and DFL programming language.

B. Future work

Future work on dynamic fuzzy logic and DFL programming language is as follows:

（1）Though literature[5] has given the abstract semantics of dynamic fuzzy logic programming language, based on the categorical description given in this paper, the algebra semantics should be defined, and the language framework should be perfected ,too. The reliability and completeness should be validated.

（2）The categorical description of the language is not concrete enough, and the proof of some theorem needs to be more perfect.

（3）Dynamic fuzzy functor should be described based on dynamic fuzzy category.

REFERENCES

[1] Lu Ru-qian. Formal semantics of computer languages (in Chinese).Beijing: Science Press, 1992.12.

[2] Yanwen Qu. Foundations of formal semantics and formal specification. Beijing: Science Press, 1989.9－120

[3] Fanzhang Li: Dynamic Fuzzy Sets and Its Applications, KunMing: Yunnan Science and Technology Press, Mar .1997

[4] Fanzhang Li et al. Dynamic Fuzzy Logic and Its Applications [M]. KunMing: Yunnan Science and Technology Press，1997.11.

[5] Fanzhang Li et al. The Dynamic Fuzzy Data Model and Programming Language Design [J]. Computer Science, 1997,vol.24(12):49-53.

[6] Fanzhang Li, Jialiang Zheng. Programming Language Design of Dynamic Fuzzy Data[J]. Computer Science,1997(supplement):54-56.

[7]Xiaofang Zhao, Fanzhang Li. The Framework of DFL Programming Language. The Journal of Communication And Computer, 3(1):1－15.

[8] Xiaofang Zhao, Hui Fan, Xiaohua Liu. Data Types of DFL Programming Language. Fuzzy Systems and Knowledge Discovery, 2007.116-120.

[9] Fanzhang Li et al. The model study of Dynamic Fuzzy Logic, Computer Research and Development, 1998, vol.35(8):714-718.

[10] Fanzhang Li. The Dynamic Fuzzy Data Computing and Model design [J]. Computer Engineering,2001，vol.27(3):100-102.

[11] Fanzhang Li, Guiquan Liu, Yumei She. Introduction of Dynamic Fuzzy Logic [M]. KunMing: Yunnan Science and Technology Press, 2005.7.8－40.

[12] Michael Winter. Products in categories of relations. Journal of Logic and Algebraic Programming, Volume 76, Issue 1, May-June 2008, Pages 145-159

[13] Diego Cazorla, Fernando Cuartero, Valentín Valero, Fernando L. Pelayo, J. José Pardo. Algebraic theory of probabilistic and nondeterministic processes. The Journal of Logic and Algebraic Programming, Volume 55, Issues 1-2, March-April 2003, Pages 57-103

[14] Glynn Winskel(write), Guoxin Song, Zhiqing Shao(transtale).Formal semantics of programming language (in Chinese). Beijng: Machine Industry Press, 2004.1

Fanzhang Li is a Professor of school of Computer Science and Technology of Suzhou University. He mainly interested in Machine Learning, Dynamic Fuzzy Logic, Quantum Logic and Theory and Applications of multi-Agent etc. He is the director of Chinese Association for artificial intelligence, senior member of China Computer Society and the standing committee & the commissioner of eight councils of Theoretical Computer Science, Machine Learning and Artificial Intelligence & Pattern Recognition etc. At the same time he is also the member of many IEEE international conferences' program committee; He has published more than 150 papers in academic journals at home & abroad and 8 works; Moreover, he is the chief presenter of Dynamic Fuzzy Logic, Lie Group Machine Learning, Differential Machine Learning and Agent Pervasive Machine Learning.