University of Ostrava

Institute for Research and Applications of Fuzzy Modeling

The software package LFLC 2000 — its specificity, recent and perspective applications

Antonín Dvořák, Hashim Habiballa, Vilém Novák, Viktor Pavliska

Research report No. 43

2002

Submitted/to appear:

Computers in Industry

Supported by:

University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
30. dubna 22, 701 03 Ostrava 1, Czech Republic

tel.: +420-69-6160 233 fax: +420-69-6120 478
e-mail: antonin.dvorak@osu.cz

The software package LFLC 2000 — its specificity, recent and perspective applications

Antonín Dvořák, Hashim Habiballa, Vilém Novák, Viktor Pavliska
University of Ostrava
Institute for Research and Applications of Fuzzy Modeling
30. dubna 22,
701 03 Ostrava 1, Czech Republic
e-mail: {Antonin.Dvorak,Hashim.Habiballa,Vilem.Novak,Viktor.Pavliska}@osu.cz

April 18, 2002

Keywords: Fuzzy logic, approximate reasoning, fuzzy control

1 Introduction

The core of applications of soft computing methods in the practice lays in the use of approximate reasoning methods. These are very general techniques, which can be applied in control, decision making, classification, pattern recognition and elsewhere, provided that a description of the given system (situation) is at disposal. The latter is supposed to be based on using set(s) of rules, each of which characterizes a very specific detailed behaviour of the system and may be expressed imprecisely. Imprecision raises from various sources — too large complexity, insufficient precise information, presence of human factor, necessity to spare time or money, etc. Very often, combination of more such factors is present.

The necessary theory is already sufficiently developed and thus, it is possible to realize it in software. There already exist a lot of software packages aimed at applications of approximate reasoning, possibly joined with (fuzzy) neural networks.

In this paper, we will present a software which differs from all packages known to the authors by consistent realization of linguistic aspects of fuzzy logic applied in fuzzy logic deduction. The software is called Linguistic Fuzzy Logic Controller (LFLC) and it exists now in two versions. The old one, LFLC 1.5 is written in Borland Pascal under MS-DOS. We have obtained a lot of experiences with it including several real applications. The new version, LFLC 2000 is written in C++ under Windows and it is fully object oriented system, which is now joined with Matlab/Simulink to enable simulation of wide class of systems.

In the next section we will briefly overview some of the theoretical background standing behind the software. Further, we provide a short description of the main features, representation of the data and outline the inner structure. Finally, we mention applications and outline intentions for future development.

2 Theoretical background

2.1 Linguistic descriptions and their elaboration

The theoretical background of LFLC lays in formal fuzzy logic in broader sense (FLb), which is an extension of that in narrow sense (FLn) (for the detailed detailed presentation of both see [11]). The theory provides elaboration of the semantics of part of natural language, which consists of the so called *evaluating* and *conditional linguistic expressions.* The former are expressions such as "small, roughly medium, very big", etc. The latter are the well known fuzzy IF-THEN rules gathered into sets called *linguistic descriptions*,

$$\mathcal{R}_1 := \textsf{IF } X \textsf{ is } \mathcal{A}_1 \textsf{ THEN } Y \textsf{ is } \mathcal{B}_1$$
$$\dots\dots\dots\dots\dots\dots\dots\dots$$
$$\mathcal{R}_m := \textsf{IF } X \textsf{ is } \mathcal{A}_m \textsf{ THEN } Y \textsf{ is } \mathcal{B}_m$$

where $\mathcal{A}_j, \mathcal{B}_j$ are the mentioned evaluating linguistic expressions. Let us remark that they characterize property of objects, which are denoted by variables X and Y. They represent, e.g. values of temperature, pressure, price, etc.

The interpretation of fuzzy IF-THEN rules depends on the purpose for which they are to be used. In general, we speak about *approximate reasoning*, which is finding a conclusion on the basis of imprecise linguistic description. There are two fundamental possibilities:

- *Linguistically based fuzzy logical deduction*, i.e. finding a formal conclusion when we treat the fuzzy IF-THEN rules as linguistically characterized logical implications.
- *Approximation of function*, i.e. finding a function which approximates some hidden function, whose existence is guessed but we have only imprecise information about it.

2.2 Fuzzy approximation

If the goal is approximation of some function then each expressions of the form 'X is $\mathcal{A}$' (this is called the *evaluating linguistic predication*) is assigned some formula $A(x)$ of predicate fuzzy logic. The whole linguistic description can be then assigned one of two special formulas called the disjunctive and conjunctive normal forms.

The *disjunctive normal form* is

$$\mathrm{DNF}(x,y) := \bigvee_{j=1}^{m} (A_j(x) \mathbin{\pmb{\wedge}} B_j(y)). \tag{1}$$

In this case, each rule is assigned a conjunction of formulas $A_j(x)$ and $B_j(y)$ and all of them are joined by disjunction. We speak also about *functional interpretation* of the linguistic description.

The alternative possibility is the *conjunctive normal form*

$$\mathrm{CNF}(x,y) := \bigwedge_{j=1}^{m} (A_j(x) \Rightarrow B_j(y)) \tag{2}$$

in which each rule is assigned an implication between the formulas $A_j(x)$ and $B_j(y)$ and all of them are joined by conjunction. In this case, the interpretation of the rules is *logical.* However, still the main goal is fuzzy approximation of a function.

Both formulas (1) and (2) correspond to certain fuzzy relations after the following assignment. Let us consider a couple of sets[†)]

$$\mathcal{V} = \langle U, V \rangle \tag{3}$$

which will be taken as a model. In the practice, we usually consider U, V to be some intervals of real numbers representing, e.g. temperatures, angles, prices, etc. Furthermore, each formula $A_j(x)$ is assigned a fuzzy set $A_{\mathcal{V},j} \underset{\sim}{\subset} U$ and $B_j(y)$ is assigned a fuzzy set $B_{\mathcal{V},j} \underset{\sim}{\subset} V$, $j = 1, \dots, m$. Then the disjunctive normal form (1) is assigned a fuzzy relation $R_{DNF,\mathcal{V}} \underset{\sim}{\subset} U \times V$ given by the membership function

$$R_{DNF,\mathcal{V}}(u,v) = \bigvee_{j=1}^{m} (A_{\mathcal{V},j}(u) \wedge B_{\mathcal{V},j}(v)), \tag{4}$$

and the conjunctive normal form (2) is assigned a fuzzy relation $R_{CNF,\mathcal{V}} \underset{\sim}{\subset} U \times V$ given by the membership function

$$R_{CNF,\mathcal{V}}(u,v) = \bigwedge_{j=1}^{m} (A_{\mathcal{V},j}(u) \to B_{\mathcal{V},j}(v)) \tag{5}$$

where $a \to b = \min(1, 1 - a + b)$, $a, b \in [0,1]$ is the so called Łukasiewicz implication[‡)]. Note that (4) is the well known Mamdani-Assilian formula used in most applications of fuzzy control.

[†)]In fact, the problem is more complex since we must precisely specify the language, the structure and assignments of all symbols from the language. For the purpose of this paper, we simplify the explanation. The interested reader is referred to [11, 12, 13].

[‡)]Alternatively, it can be any residuation operation based on some continuous t-norm.

Now, let some value $u_0 \in U$ be given (i.e. this is some precise measurement of, say temperature, on the basis of which we should find a proper control action). Then using (4) or (5) we derive a fuzzy set $B_{u_0} \underset{\sim}{\subset} V$ with the membership function

$$B_{u_0} = \left\{ {}^{R(u_0,v)}/v \mid v \in V \right\}. \tag{6}$$

The result of this kind of elaboration of fuzzy IF-THEN rules is an approximating function $f^A : U \longrightarrow V$ given by the formula

$$f^A(x) = \mathrm{DEF}(B_x), \qquad x \in U \tag{7}$$

where DEF is a defuzzification function. In the case just described, one of the best possible defuzzifications is Center of Gravity Method. More about approximation properties can be found in [12, 13].

2.3 Linguistically based fuzzy logical deduction

The most specific feature of LFLC is the possibility to realize a *fuzzy logical deduction* when the rules are interpreted as *linguistically characterized* logical implications.

2.3.1 Linguistic feature

The LFLC system works with the so called evaluating linguistic expressions (possibly with signs) which have the general form

$$\langle \text{linguistic modifier} \rangle \langle \text{atomic term} \rangle \tag{8}$$

where ⟨atomic term⟩ is one of the words "small, medium, big", or "zero" (possibly also arbitrary symmetric fuzzy number) and ⟨linguistic modifier⟩ is an intensifying adverb such as "very, roughly", etc.

The linguistic modifiers in (8) are of two basic kinds, namely those with narrowing and extending effect. *Narrowing* modifiers are, for example, "extremely, significantly, very" and *widening* ones are "more or less, roughly, quite roughly, very roughly". We will take these modifiers as canonical. Note that narrowing modifiers make the meaning of the whole expression more precise while widening ones do the opposite. Thus, "very small" is more precise than "small", which, on the other hand, is more precise than "roughly small".

The meaning of each linguistic expression $\mathcal{A}$ has two parts: the *intension* $\mathrm{Int}(\mathcal{A})$ and *extension* $\mathrm{Ext}(\mathcal{A})$. Intension is a formal characterization of the meaning on the level of formal syntax and can be understood as a fuzzy set of special formulas[†)]. Furthermore, we suppose some model is given, which can be seen as a set U (usually an interval of real numbers). Then the extension of $\mathcal{A}$ is some fuzzy set of elements $\mathrm{Ext}(\mathcal{A}) \underset{\sim}{\subset} U$, which is determined by its intension $\mathrm{Int}(\mathcal{A})$.

Let us stress that the extensions of the evaluating expressions are fuzzy sets of the form of the so called S- and Π-curves, as is depicted on Figure 1. More on the formal theory of evaluating linguistic expressions can be found in [9, 11].

2.3.2 Fuzzy logical deduction

Unlike fuzzy approximation, where we dealt with fuzzy sets in a model (i.e. on the level of semantics), logical deduction must proceed on syntax. Instead of the detailed formal description, we will demonstrate the behaviour of logical deduction on an example.

Let us consider a linguistic description consisting of two rules:

$$\mathcal{R}_1 := \textsf{IF } X \textsf{ is small AND } Y \textsf{ is small THEN } Z \textsf{ is big}$$
$$\mathcal{R}_2 := \textsf{IF } X \textsf{ is big AND } Y \textsf{ is big THEN } Z \textsf{ is small.}$$

These rules are assigned intensions $\mathrm{Int}(\mathcal{R}_1), \mathrm{Int}(\mathcal{R}_2)$, which can schematically be written as

$$\mathrm{Int}(\mathcal{R}_1) = (\mathbf{Sm}_x \wedge \mathbf{Sm}_y) \qquad \Rightarrow \mathbf{Bi}_z \tag{9}$$
$$\mathrm{Int}(\mathcal{R}_2) = (\mathbf{Bi}_x \wedge \mathbf{Bi}_y) \qquad \Rightarrow \mathbf{Sm}_z. \tag{10}$$

Furthermore, let X, Y, Z be interpreted in a model which will consist of three sets $U = V = W = [0, 1]$. Then small values are some values around 0.3 (and smaller) and big ones some values around 0.7 (and

†) They have the form $\mathbf{A} := \{{}^{a_t}/A_x[t] \mid t \in M\}$ where $A(x)$ is a formula, M is a set of constants and a_t is an evaluation of the instance $A_x[t]$. For the details — see [11].

bigger). Of course, given the input, e.g. $X = 0.3$ and $Y = 0.25$ then we expect the result $Z \approx 0.7$ due to the rule $\mathcal{R}_1$. Similarly, for $X = 0.75$ and $Y = 0.8$ we expect the result $Z \approx 0.25$ due to the rule $\mathcal{R}_2$.

The value 0.3 is represented in the formal system by a certain intension $\mathbf{Sm}'_x$ and similarly, the value 0.25 is represented by $\mathbf{Sm}'_y$.

Then the inference rule of modus ponens is applied on $\mathbf{Sm}'_x$, $\mathbf{Sm}'_y$ and the implication (9). The result is the intension $\mathbf{Bi}'_z$. The latter is to be interpreted as some fuzzy set $B' \underset{\sim}{\subset} W$.

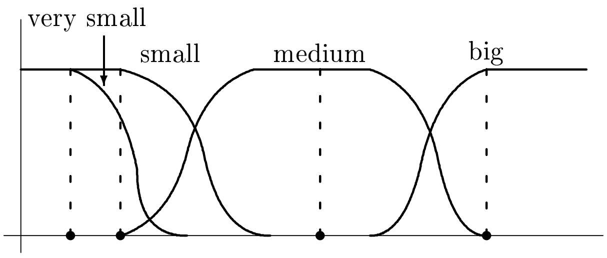


Figure 1: Form of fuzzy sets corresponding to the meaning of the evaluating linguistic expressions and the DEE defuzzification.

To obtain one concrete value, the resulting fuzzy set B' should further be defuzzified. However, we deal with evaluating linguistic expressions, whose interpretation has always one of the three possible forms depicted on Figure 1. Therefore, standard defuzzification methods such as COG do not work properly. Instead, we have developed a special method, which we call Defuzzification of Evaluating Expressions (DEE). This method classifies first the type of the membership function and then decides the defuzzification, as is depicted on Figure 1. There are two versions of the DEE method, namely *simple* which first classifies the resulting fuzzy sets in types "small", "medium" and "big" and then defuzzifies it using Last of Maxima, Center of Gravity or First of Maxima methods, respectively. The second one uses a sophisticated algorithm to choose a value close to these dependingly on the specific shape of the membership function.

In our case, when the input is $X = 0.3$ and $Y = 0.25$ then both values correspond to "small" and thus, with respect to the rule $\mathcal{R}_1$, the resulting linguistic corresponds to "big" and thus, after its interpretation in the model and defuzzification using the DEE method, we obtain the result $Z \approx 0.7$, i.e. a value being intuitively big. In other words, we obtain the result which, on the basis of the form of the given rules, should be expected. Similarly, the input values $X = 0.75$ and $Y = 0.8$ would lead to the value $Z \approx 0.25$ due to the rule $\mathcal{R}_2$.

To summarize: in the case of fuzzy approximation, we form the special formulas DNF or CNF on the level of syntax, interpret them in some model and then find the approximation on the level of semantics only. In the case of linguistically based fuzzy logical deduction we interpret the rules on the level of syntax, transform measurement also on this level, realize formal logical deduction and then interpret the result in some model.

3 Purpose of LFLC

Recall that the original idea of the fuzzy controller proposed by L. A. Zadeh [15] and E. H. Mamdani [8] is to translate knowledge of the human control operator into mathematical description in a way to mimic a successful course of his/her control. Since such knowledge is usually expressed using evaluating linguistic expressions, the main problem consists in translation of the linguistic expressions into mathematical form.

The main purpose of LFLC software system is the design, testing and learning of linguistic descriptions. These descriptions can be further used in control, decision support and other applications (see Section 7). We can distinguish the following main tasks realized by LFLC:

- **Design of linguistic descriptions:** LFLC allows to use various pre-defined linguistic expressions (e.g. *small*, *about* 5, *more or less medium* etc.) It has also means for analysis of several properties of linguistic descriptions (sorting, detection of identical or inconsistent IF-THEN rules etc.)

- **Design and modification of user expressions:** In addition to pre-defined linguistic expressions the user can also define and use his own expressions in situations, when standard ones are for some reasons insufficient.
- **Testing of inference over designed linguistic descriptions:** LFLC allows the user to visualize the behavior of the linguistic description for various (crisp) observations. He/she can select various inference and defuzzification methods (cf. Section 2), see projections with respect to individual variables or three-dimensional control surface. There is also information about IF-THEN rules fired and most suitable linguistic expressions assigned to crisp values from input or output intervals.
- **Learning of linguistic description from experimental data:** LFLC implements two methods for learning of linguistic description from data sets. The first method is based on the ability to find proper evaluating linguistic expression corresponding to the given value. The resulting linguistic description should be used for finding conclusions using logical deduction.

  The second possibility is based on the theoretical results published in [12, 13] and it enables to find a linguistic description interpreted by DNF with the prescribed accuracy of approximation of the data understood as specification of some function.

4 Important data structures and algorithms

The LFLC software system is implemented in C++programming language with full use of object-oriented methodology. In the following we describe the most important data structures used for the implementation of linguistic descriptions and related notions. In this section we use the common C++terminology such as *class*, *method*, *instance* etc.

- ESyntax
- ESyntaxRB
- CFuzzySet
  - CFuzzyParamSet
    - CFuzzyTriangSet
    - CFuzzyTrapezoidSet
    - CFuzzyQuadraticSet
  - CFuzzyDiscreteSet
    - CCompressedFuzzyDiscreteSet
  - CFuzzyACutSet
  - CFuzzyFuncDefSet
- CSettings
  - CLinCSettings
  - CBasicSettings
    - CTriangSettings
    - CLinguisticSettings
- C1ArgOper
  - CNegation
  - CModifier
- C2ArgOper
  - CMinTNorm
  - CMaxTConorm
  - CLukasiewiczTNorm
  - CLukasiewiczImplication
  - CHamacherTNorm
  - CFrankTNorm
- CFuzzyVariable
- CRuleBase
- CValue
  - CLinCValue
  - CFuzValue
- CAlphaCut
- CUniverse

Figure 2: List and structure of main classes

Figure 4 shows the hierarchy of the main classes of LFLC. There are classes for representation of fuzzy sets (`CFuzzySet` and its derivatives), classes which allow assignments of fuzzy sets to the linguistic expressions (`CLinguisticSettings`), classes for representation of operations on truth degrees and induced operations on fuzzy sets (`C1ArgOper`, `C2ArgOper` and their derivatives), class which represents an overall linguistic description `CRuleBase` and several others auxiliary classes.

There are three important concepts, namely the fuzzy set, the semantical rule which assigns fuzzy sets to linguistic expressions and, finally, the linguistic description. Each of these concepts has a corresponding counterpart in the implementation, namely the class. These classes `CFuzzySet`, `CLinguisticSettings` and `CRuleBase` are not, however, on the same level of generality. Instances of the `CFuzzySet` are members of `CLinguisticSettings` and, similarly, instances of the `CLinguisticSettings` are members of `CRuleBase`.

4.1 Representation of fuzzy sets

The class `CFuzzySet` is the basis for representation of all types of fuzzy sets and implements their two common properties, namely the *context* and the *membership function*. Therefore it contains as its member an instance of the class `CUniverse` which is designed for representation of the context (or universe) of (generally multidimensional) fuzzy set. Further, it has the method `GetMembDeg` that returns membership degree for every point from the context. This method is pure, no particular membership function is defined in the class `CFuzzySet` itself. It has to be defined by classes derived from it.

There are four basic types of fuzzy sets derived from `CFuzzySet` which can be distinguished by implementation of the membership function:

- *Discrete fuzzy sets*, represented by the class `CFuzzyDiscreteSet`, define fuzzy sets on discretized universe which are widely used especially in inference, defuzzification and other computational routines. There is also special implementation of discrete fuzzy set `CCompressedFuzzyDiscreteSet` designed for better memory usage, where the zero membership values lying outside of the support of the fuzzy set are not stored in memory.
- *Parametric fuzzy sets* implemented in the classes `CFuzzyQudraticSet`, `CFuzzyTriangleSet` and `CFuzzyTrapezoidalSet` are most often used as extensions of the evaluating linguistic expressions.
- Fuzzy sets with membership function represented by means of *α-cuts* (see, e.g. [7]) implemented by class `CFuzzyAlphaCutSet`. Each α-cut is represented by class `CAlphaCut`.
- *General fuzzy sets* with membership function defined using arbitrary function given by user are implemented in a class `CFuzzyFuncDefSet`.

Operations on fuzzy sets are implemented by classes derived from `C1ArgOper` and `C2ArgOper`. Classes derived from `C1ArgOper` implement *prefix* operators:

- `CModifier` represents modifier of Zadeh's type (e.g. power).
- `CNegation` represents negation.

The subclasses of `C2ArgOper` implement as methods various types of intersections, unions, implications and differences of fuzzy sets. In this way e.g. intersection have no predefined t-norm, but it is possible to use any t-norm implemented by some subclass of `C2ArgOper`.

4.2 Representation of semantical rule

The semantical rule which assigns extensions (fuzzy sets) to evaluating linguistic expressions is represented by the derivatives of the class `CSettings`. It is a pure class which contains the extensions of atomic evaluating expressions (array of instances of classes derived from `CFuzzyParamSet`) and meanings of modifiers (array of instances of class `CModifier`) on appropriate context (class `CUniverse`), i.e. all the necessary information needed for the computation of the meaning of linguistic expressions (see Section 2). Derived classes have to implement the fundamental method `GetMeaning` that should return appropriate extension of a given linguistic expression. Derived classes have the following structure:

- `CBasicSettings` – base class for the situations when atomic terms are parametrically defined families of fuzzy sets:
  - `CLinguisticSettings` – meanings of atomic terms are fuzzy sets with quadratic membership function,
  - `CTriangSettings` – triangular membership functions,
  - `CTrapezoidSettings` – trapezoidal membership functions.
- `CLinCSettings` – meaning is linear combination of input variables, this class is used for evaluation of the output in Takagi-Sugeno models [14].

4.3 Representation of linguistic description

The linguistic description is represented by the class `CRuleBase` and the overall structure of that class is shown in Figure 3. It contains arrays of antecedent and succedent fuzzy variables (that corresponds to inputs and outputs of fuzzy controller), and pointers to inference and defuzzification routines. The intended purpose of the classes is the following:

- `CFuzzyVariable` represents one fuzzy variable and an appropriate row from linguistic description, which is an array of instances of the class `CValue`.

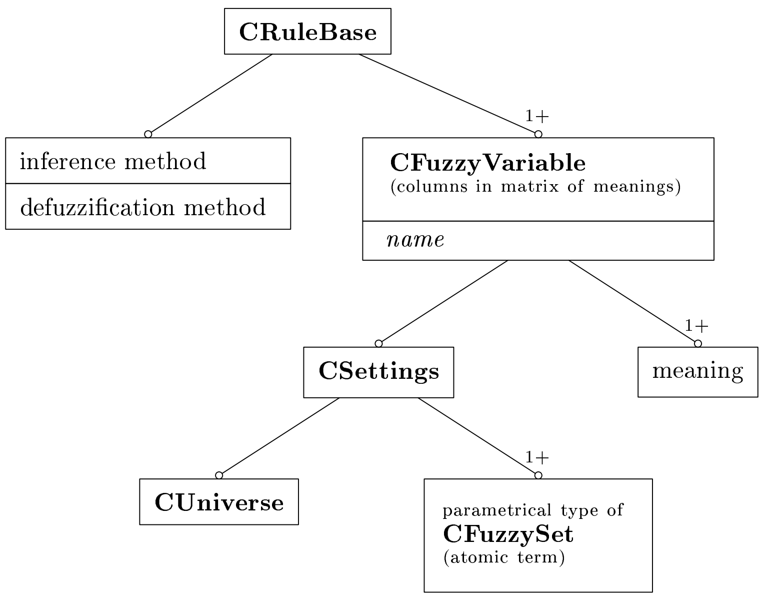


Figure 3: Structure of **CRuleBase**

- `CValue` represents the linguistic expression as a character string and also the extension of the linguistic expression (the corresponding fuzzy set). There are two derived classes:
  - `CLinCValue` represents the meaning of succedent expression in Takagi-Sugeno model (cf. [14]) in the form of array of real numbers — coefficients of linear combination of input variables.
  - `CFuzValue` represents the extension of linguistic expression by means of the instance of the class `CFuzzyDiscreteSet`.

4.4 Inference and defuzzification

The LFLC software system implements various inference and defuzzification methods. For thorough discussion, see Section 2 and book [11]. It allows the user to compare different combinations of them and, consequently, types of approximate reasoning mechanisms. The inference methods are (cf. Section 2):

- *Fuzzy logical deduction* (see [4]).
- *Fuzzy approximation* using *disjunctive normal form* (the linguistic description is used to characterize vaguely some crisp function; see [12, 13]).
- *Fuzzy approximation* using *conjunctive normal form* (the linguistic description is also used to characterize a crisp function, but IF-THEN are rules interpreted as implications).

Defuzzification methods implemented in LFLC are:

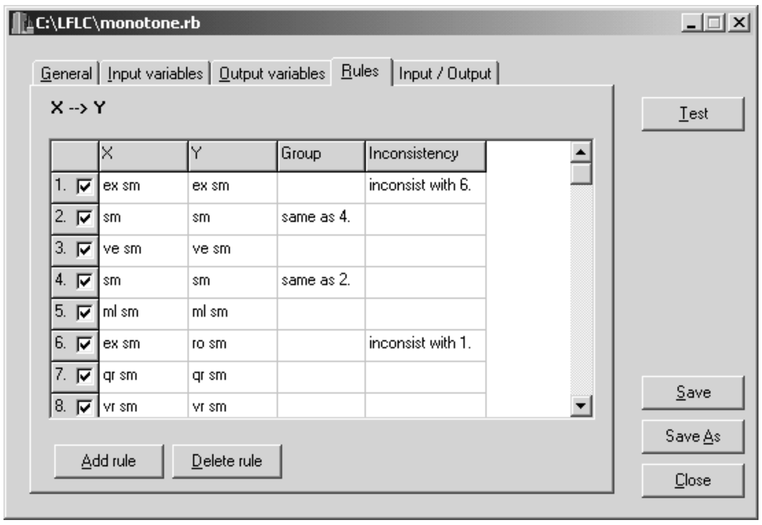
- *Center of Gravity* (COG),
- *Mean of Maxima* (MOM),
- *Modified Center of Gravity* (ModCOG),
- *Defuzzification of Linguistic Expressions* (DEE),
- *Simple Defuzzification of Linguistic Expressions* (SDEE).

For the detailed discussion of the defuzzification methods, see [5]. Modified COG is described in [2]. Recall that the DEE method is a special defuzzification method designed for use in conjunction with Fuzzy Logical Deduction inference method (see also [4]).

5 User interface

The user interface of the LFLC application is designed under Win32 platform. It provides all of the tasks mentioned in Section 3 in standard manner of windows application.

The simplest way how to describe the user interface is to show some screenshots of the running application. Figure 4 shows the most important dialog window in which the user can edit rules of the linguistic description (rulebase). Note the indication that there are some duplicate or inconsistent rules.



X --> Y

	X	Y	Group	Inconsistency
1. ✓	ex sm	ex sm		inconsist with 6.
2. ✓	sm	sm	same as 4.	
3. ✓	ve sm	ve sm		
4. ✓	sm	sm	same as 2.	
5. ✓	ml sm	ml sm		
6. ✓	ex sm	ro sm		inconsist with 1.
7. ✓	qr sm	qr sm		
8. ✓	vr sm	vr sm		

Figure 4: Design of IF-THEN rules

Figure 5 contains the "test screen" dialog where the user can test the behavior of the inference on the basis of the designed linguistic description. On the left hand side the user can interactively enter the crisp inputs to the inference for each input variable. On the bottom of left side he/she can see the whole text version of the rulebase file. On the right hand side of the dialog window, the user can change the defuzzification and inference methods (cf. Section 4.4). Further he/she can see the output fuzzy set of the inference for the previously entered crisp inputs together with the defuzzified value. Next thing which he can there see is the number of fired rules in the inference. Last but not least item is the projection over one chosen variable.

6 Interface to other software systems

For communication with other software systems the standard COM object *RBaseCOM* is designed. It encapsulates the core methods of the LFLC package. Definition of its interface follows:

LoadFromFile(BSTR FileName) Loads rulebase with the specified *FileName*.

int NumInputVars() Returns the number of input variables from the loaded rulebase.

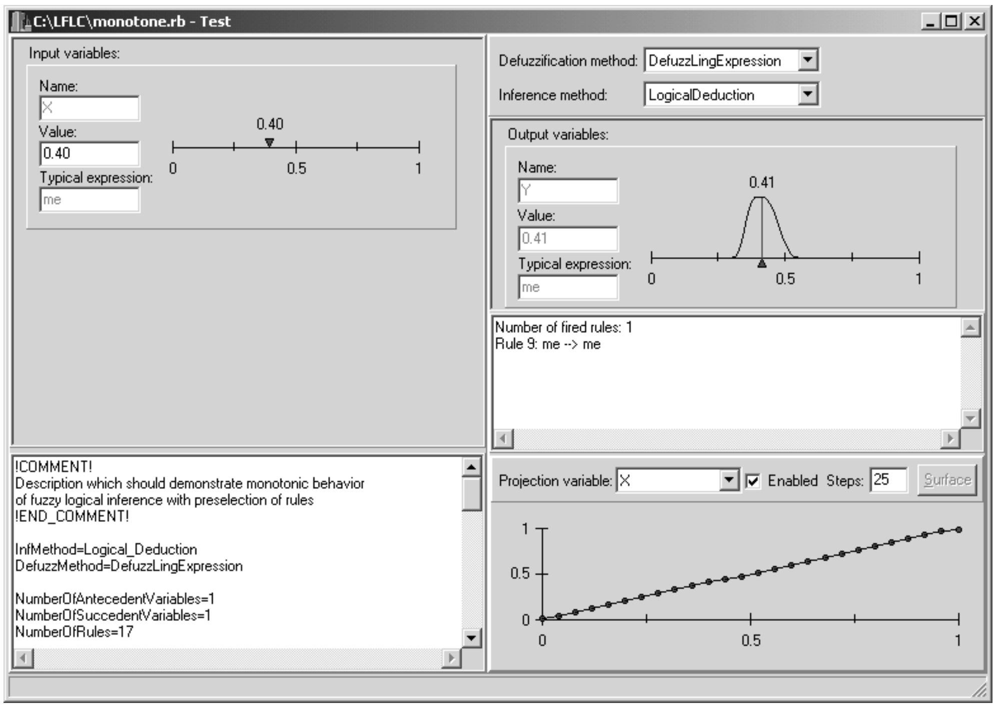
double HiBoundOfVar(int VarIndex) Returns high bound of the variable with *VarIndex*

double LoBoundOfVar(int VarIndex) Returns low bound of the variable with *VarIndex*

BSTR VarName(int VarIndex) Returns name of the variable with *VarIndex*

double Inference(TVariantInParam Inputs) Performs inference on *Inputs* values.

A Matlab S-Function for Simulink package has also been developed. This provides link with Matlab and thus enables simulation, e.g. of control of various kinds of processes using the LFLC system.


C:\LFLC\monotone.rb - Test

Input variables:

Name: X

Value: 0.40

Typical expression: me

Defuzzification method: DefuzzLingExpression

Inference method: LogicalDeduction

Output variables:

Name: Y

Value: 0.41

Typical expression: me

```
Number of fired rules: 1
Rule 9: me --> me
```

```
!COMMENT!
Description which should demonstrate monotonic behavior
of fuzzy logical inference with preselection of rules
!END_COMMENT!

InfMethod=Logical_Deduction
DefuzzMethod=DefuzzLingExpression

NumberOfAntecedentVariables=1
NumberOfSuccedentVariables=1
NumberOfRules=17
```

Projection variable: X ✓ Enabled Steps: 25

Figure 5: Testing of behavior of the designed linguistic description.

7 What applications can be realized

The LFLC software has wide abilities for applications. Let us stress that the possibility to use pure linguistic expressions have been appreciated in all cases. The user thinks only in linguistic terms and can forget about fuzzy sets. This feature tremendously increases the explicability of the linguistic descriptions, which are easily understood even long time after the application design is finished.

The previous version, LFLC 1.5, has been used in several real applications, among which the most sophisticated was large scale application of fuzzy control in the Metallurgic plant Břidličná (see [10]). The problem was to control an aluminium smelting furnace. The biggest difficulty for control of the production was discontinuity of the process. This means that the sequence “filling the furnace — smelting — emptying” continuously repeats in regular periods of the length of approximately 8 hours. Moreover, the phase of smelting is extremely slow with great inertia and it is influenced by the latent heat in negative way. The main phase comes when the temperature of the melted aluminium reaches the technological level (approximately 740–800°C). Then it is important to keep it for about one hour. The proces is highly nonlinear and before using fuzzy control, several other techniques including adaptive control have been applied.

The applications of fuzzy control in Břidličná started in 1995. After good experiences with the first furnace fuzzy control, it was decided to apply it on the other four furnaces one by one, too. At present, the system works in the whole enterprise, which consists of five furnaces.

Except for pure fuzzy control (several tens of simulations with fuzzy control of various kinds of processes have been realized), LFLC can be applied also in multicriteria decision making. For example, several studies for its application in decision making of the bank concerning creditworthiness of its clients have been prepared. At present, a study for application in control of traffic junction is being elaborated.

8 Conclusion — further development

Further development of the software is supposed in several directions. The main goal is to enable to design hierarchical structures, which may consist of various kinds of units, each of them possibly realizing different inference mechanism. It is also necessary to complete the system by the possibility to form Takagi-Sugeno rules (they are already prepared in the kernel of the system).

A very important part with wide potential for applications is learning. Besides the second learning method noticed in Section 3, we suppose to implement learning abilities based on neural nets. We also plan to extend the linguistic power of the system, besides others by generalized (fuzzy) quantifiers, which would thus extend the applicability of the system to summarization of data (see [3]).

References

[1] Bělohlávek, R., A. Dvořák, D. Jedelský and V. Novák (1998) Object Oriented Implementation of Fuzzy Logic Systems. In: Camarinha-Matos, L.M. et al. (eds.): *Intelligent Systems for Manufacturing. Multi-Agent Systems and Virtual Organizations.* Kluwer, Dordrecht, 589–594.

[2] Dvořák, A. and D. Jedelský (1999) Defuzzification and Chaining of Rules in Hierarchical Fuzzy Systems. In: *Proceedings of 1999 Eusflat-Estylf Joint Conference*, 199-202.

[3] Dvořák, A. and V. Novák (2000) On the Extraction of Linguistic Knowledge in Databases Using Fuzzy Logic. In: H. L. Larsen et al. (Eds.): *Flexible Query Answering Systems. Recent Advances.* Physica-Verlag, Heidelberg-New York, 445-454.

[4] Dvořák A, Novák V (submitted) Fuzzy Logic Deduction with Crisp Observations. Soft Computing.

[5] Van Leekwijck W, Kerre EE (1999) Defuzzification: criteria and classification. *Fuzzy Sets and Systems*, **108**: 159-178.

[6] Kerre EE, De Cock M (1999) Linguistic Modifiers: An Overview. In: Chen G. et al. (eds) *Fuzzy Logic and Soft Computing.* Kluwer, Boston.

[7] Klir, G. J. and Bo Yuan (1995) *Fuzzy Sets and Fuzzy Logic. Theory and Applications.* Prentice Hall, Englewood Cliffs.

[8] Mamdani E.H. and Assilian S. (1975). An experiment in linguistic synthesis with a fuzzy logic controller, *Int. J. of Man-Machine Studies*, **7**: 1–13.

[9] Novák V (2001) Antonyms and Linguistic Quantifiers in Fuzzy Logic. *Fuzzy Sets and Systems*, **124**: 335–351.

[10] Novák V, Kovář J (2000) Linguistic IF-THEN Rules in Large Scale Application of Fuzzy Control. In: Da Ruan, Kerre EE (eds) *Fuzzy If-Then Rules in Computational Intelligence: Theory and Applications.* Kluwer, Boston, 223–241.

[11] Novák V, Perfilieva I, Močkoř J (1999) *Mathematical Principles of Fuzzy Logic.* Kluwer, Boston.

[12] Perfilieva I (2000) Fuzzy Relations, Functions, and Their Representation by Formulas. *Neural Network World*, **10**: 877–890.

[13] Perfilieva I (2001) Normal Forms for Fuzzy Logic Functions and Their Approximation Ability. *Fuzzy Sets and Systems*, **124**: 371–384.

[14] Takagi, T and M. Sugeno (1985) Fuzzy Identification of Systems and its Applications to Modeling and Control, *IEEE Transaction on System, Man, and Cybernetics*, **15**: 116–132.

[15] Zadeh, L.A. (1972). A Rationale for Fuzzy Control, *J. of Dynamical Systems, Measurement, an Control (trans. ASME. Ser. G.)*, **94**: 3–4.