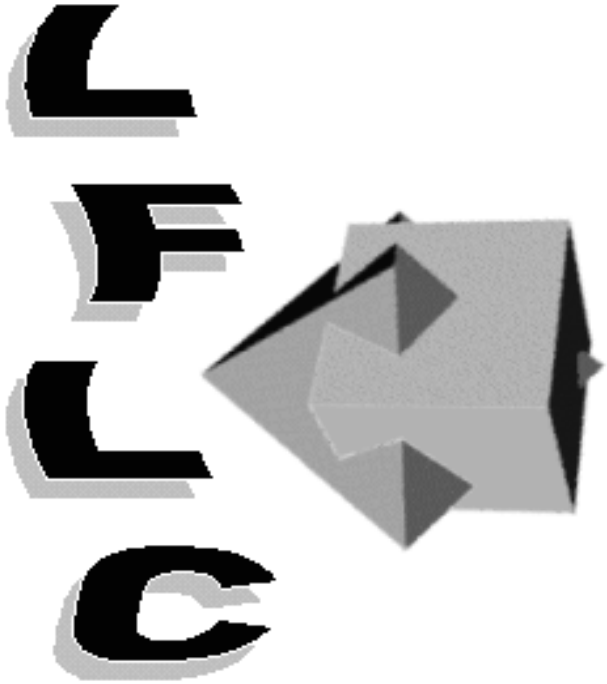
LFLC 2000
Linguistic Fuzzy Logic Controller



Ústav pro výzkum a aplikace
fuzzy modelování
Ostravská Univerzita v Ostravě

Obsah

1 Popis LFLC 2000 **4**

1.1 Fuzzy IF-THEN pravidla 5

1.1.1 Příklad 5

1.1.2 Funkcionální výklad 6

1.1.3 Jazykové výrazy 7

1.2 Přibližná dedukce 8

2 Práce s jazykovým popisem **8**

2.0.1 Otevření nebo vytvoření nového jazykového popisu 8

2.1 General — obecné nastavení jazykového popisu 9

2.1.1 Name of unit — název jednotky 9

2.1.2 Type of inference — typ inference 9

2.1.3 Příklad 10

2.1.4 Defuzzication — defuzzifikace 12

2.1.5 Description — popis 13

2.2 Input and output variables — Vstupní a výstupní proměnné . . . 14

2.2.1 Editování výrazů 15

2.2.2 User - uživatelské 15

2.2.3 Standard 17

2.2.4 Fuzzy numbers — fuzzy čísla 17

2.3 Rules — pravidla 18

2.3.1 Evaluační jazykové výrazy 19

2.3.2 Editování pravidel 20

2.3.3 Příklad 20

2.3.4 Třídění pravidel 21

2.4 Input/Output (Vstup/Výstup) 21

2.4.1 Funkce Compute 22

2.4.2 Automatické generování jazykového popisu 22

2.4.3 Příklad 23

3 Testování jazykového popisu **24**

3.1 Levá horní část 24

3.2 Levá dolní část 24

3.3 Pravá horní část 25

3.4 Pravá dolní část 25

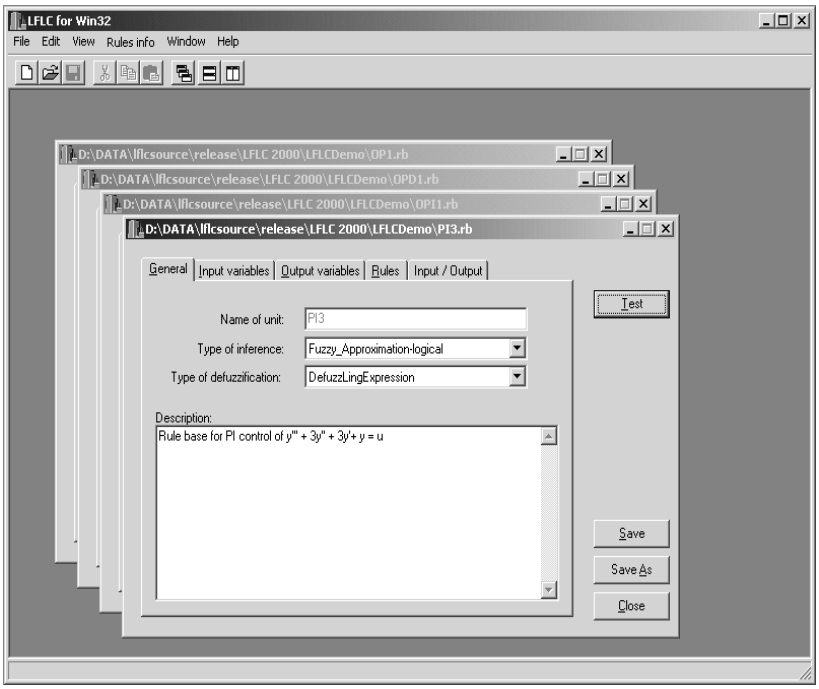
4 Využití v jiných aplikacích **25**

4.1 LFLC 2000 COM Server 25

4.2 LFLC 2000 MATLAB/Simulink klient 26

4.3 LFLC 2000 MATLAB klient 27

1 Popis LFLC 2000



LFLC 2000 (Linguistic Fuzzy Logic Controller) je specializovaný software, jenž je založen na teorii fuzzy množin a fuzzy logice a umožňuje odvozovat závěry na základě nepřesného popisu určité situace pomocí jazykově formulovaných *fuzzy IF-THEN (JESTLIŽE-PAK) pravidel.* Charakteristické pro tento

software je to, že umožňuje pracovat se skutečnými jazykově definovanými pravidly, které tvoří jazykový popis procesu, rozhodovací nebo klasifikační situace. Uživatel tedy může pracovat pouze s výrazy přirozeného jazyka, aniž by byl nucen uvažovat o způsobu, jak je definovat. Počítač se chová jakoby „partner" rozumějící přirozenému jazyku svého uživatele.

V dalším výkladu budeme fuzzy množinou A na univerzu U (značíme jako $A \subseteq U$) rozumět speciální funkci

$$A : U \longrightarrow [0,1]$$

přičemž hodnoty $A(x) \in [0,1]$ jsou stupně příslušnosti prvku $x \in U$ do fuzzy množiny A. Stupeň příslušnosti můžeme chápat jako *stupeň pravdivosti* vyjadřující míru *pravdivosti* tvrzení, že prvek $x \in U$ *náleží* do fuzzy množiny A.

Pomocná literatura užitečná pro pochopení principů tohoto software:

Reference

[1] Novák, V.: **Fuzzy množiny a jejich aplikace.** SNTL, Praha 1986 and 1990.

[2] Novák, V., Perfilieva I. and J. Močkoř: **Mathematical Principles of Fuzzy Logic**. Kluwer, Boston/Dordrecht 1999.

[3] Novák, V.: **Základy fuzzy modelování.** BEN, Praha 2000.

1.1 Fuzzy IF-THEN pravidla

Hlavní úlohou LFLC 2000 je práce (návrh a testování) s jazykovými popisy. Jazykové popisy jsou množiny fuzzy IF-THEN pravidel ve tvaru

$$\mathsf{IF}\ X_1\ \mathsf{is}\ \mathcal{A}_1\ \mathsf{AND}\ \cdots\ \mathsf{AND}\ X_n\ \mathsf{is}\ \mathcal{A}_n\ \mathsf{THEN}\ Y\ \mathsf{is}\ \mathcal{B},$$

kde $\mathcal{A}_1, \ldots, \mathcal{A}_n$ a $\mathcal{B}$ jsou jisté predikáty charakterizující proměnné $X_1, \ldots, X_n$ a Y. Pravidla jsou často specifikovány jazykově. Software umožňuje pracovat se předem definovanými jazykovými výrazy. Uživatel však může definovat své vlastní výrazy.

1.1.1 Příklad

Příkladem fuzzy IF-THEN pravidla je

IF *překážka je blízko* AND *rychlost vozidla je vysoká*
THEN *brzdná síla je velmi velká*.

"Překážka", "rychlost" and "brzdná síla" jsou proměnné zatímco "blízko", "vysoká" a "velmi velká" jsou výrazy vágně charakterizující význam proměnných.

Část pravidla nacházející se před THEN se nazývá *antecedent* a část za *sukcedent*. Proměnné $X_1, \ldots, X_n$ nazýváme *vstupní(input)* nebo *nezávislé* proměnné. Proměnná Y se nazývá *výstupní(output)* nebo *závislá* proměnná.

Množina fuzzy IF-THEN pravidel společně tvoří *jazykový popis*

$$\begin{aligned} \mathcal{R}_1 &:= \mathsf{IF}\ X_1 \ \mathsf{is}\ \mathcal{A}_{11}\ \mathsf{AND}\ \cdots\ \mathsf{AND}\ X_n\ \mathsf{is}\ \mathcal{A}_{1n}\ \mathsf{THEN}\ Y\ \mathsf{is}\ \mathcal{B}_1 \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ \mathcal{R}_m &:= \mathsf{IF}\ X_1 \ \mathsf{is}\ \mathcal{A}_{m1}\ \mathsf{AND}\ \cdots\ \mathsf{AND}\ X_n\ \mathsf{is}\ \mathcal{A}_{mn}\ \mathsf{THEN}\ Y\ \mathsf{is}\ \mathcal{B}_m \end{aligned}$$

Kromě sofistikovaných nástrojů pro tvorbu jazykového popisů poskytuje software LFLC 2000 několik druhů mechanismů přibližné dedukce, pomocí nichž lze nalézt závěr na základě jazykového popisu. Volba dedukčního mechanismu záleží na požadavcích uživatele a výběru zásadního přístupu k *interpretaci IF-THEN pravidel*.

(i) *Logický výklad* – pravidla jsou chápána jako jazykově charakterizované logické implikace.

(ii) *Funkcionální výklad* – pravidla jsou chápána jako části popisu funkční závislosti v nějakém modelu.

Logický výklad

Je založen na předpokladu, že fuzzy IF-THEN pravidla jsou skutečné jazykově charakterizované logické implikace. Poznamenejme, že logický výklad fuzzy IF-THEN pravidel dohromady s níže popsanou fuzzy logickou dedukcí je příznačný pro LFLC 2000, a proto (pokud je nám známo) nemá konkurenta v jiných softwarovych systémech pro fuzzy modelování.

1.1.2 Funkcionální výklad

Je tradičnější interpretací fuzzy IF-THEN pravidel. V tomto případě je jazykový popis charakterizací funkční závislosti, jejíž existence je předpokládána, ale která není známa přesně. Výsledkem inference na základě zadaného jazykového popisu je nějaká funkce aproximující tuto závislost.

Jazykovému popisu jako celku je přiřazena jedna z normálních forem:
disjunktivní normální forma

$$\bigvee_{j=1}^{m} (A_{j1}(x_1) \wedge \cdots \wedge A_{jn}(x_n) \wedge B_j(y)),$$

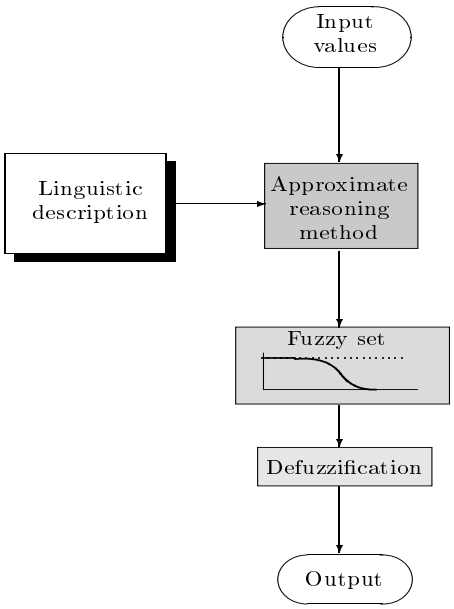
nebo *konjunktivní normální forma*

$$\bigwedge_{j=1}^{m}((A_{j1}(x_1) \wedge \cdots \wedge A_{jn}(x_n)) \Rightarrow B_j(y)).$$

1.1.3 Jazykové výrazy

Výrazy $\mathcal{A}_1, \ldots, \mathcal{A}_n$, $\mathcal{B}$ uvnitř fuzzy IF-THEN pravidel mohou být buď evaluační jazykové výrazy nebo jednoduše označení nějakých pojmů. V prvním případě je hlavní myšlenkou vytvořit počítač, který „rozumí“ popisům formulovaným v přirozeném jazyce a jehož chování odpovídá jejich smyslu. Uživatel pracuje s jazykovými výrazy stejně, jakoby používal přirozený jazyk a může předpokládat, že jim počítač „rozumí“ i bez bližší specifikace odpovídajících fuzzy množin (samozřejmě, že tyto fuzzy množiny lze modifikovat, pokud si to přejeme).

Ve druhém případě nejsou výrazy $\mathcal{A}$ a $\mathcal{B}$ explicitně formulovány v přirozeném jazyce, ale jsou pouze označením (zadanými uživatelem) fuzzy množin reprezentujících významy určitých pojmů v mysli uživatele. Fuzzy množiny mohou být modifikovány tak, aby se dosáhlo nejlepšího možného přiblížení k uživatelově představě bez ohledu na jazykový význam použitých výrazů.



Obrázek 1: Obecné schéma přibližné dedukce

1.2 Přibližná dedukce

Jazykový popis umožňuje přibližnou dedukci, což je procedura pomocí níž z nějakých informací vyvozujeme závěr. Obecné schéma přibližné dedukce je znázorněno na obrázku 1.

Vstupní hodnoty jsou konkrétní hodnoty vstupních proměnných, které jsou později využity vybranou metodou přibližné dedukce spolu s jazykovým popisem k určení výstupní fuzzy množiny. Ve většině případů potřebujeme pouze jednu konkrétní hodnotu, která by měla být odvozena z výsledné výstupní fuzzy množiny. Toto odvození je realizováno prostřednictvím nějaké defuzzifikační metody.

2 Práce s jazykovým popisem

2.0.1 Otevření nebo vytvoření nového jazykového popisu

Vyberte si jednu z následujících možností:

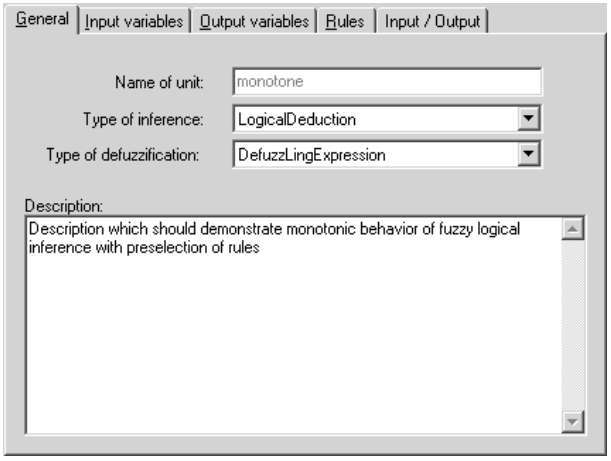
File → *Open* nebo stlačte **Ctrl O**

File → *New* nebo stlačte **Ctrl N**

a vyberte soubor (jehož přípona je .rb). Následně se otevře okno zobrazující úplné informace o jazykovém popisu, jenž je v tomto souboru uložen. Okno má následující Tab-záložky:

- General — obecné charakteristiky jazykového popisu.
- Input variables — vstupní proměnné.
- Output variables — výstupní proměnná.
- Rules — jazykově formulovaná pravidla.
- Input/Output — záložka pro učení jazykového popisu z dat modelu

2.1 General — obecné nastavení jazykového popisu



Na této Tab-záložce se nastavují následné obecné charakteristiky jazykového popisu.

2.1.1 Name of unit — název jednotky

Specifikuje jméno jazykového popisu. Toto jméno je identické se jménem souboru, ve kterém je popis uložen.

2.1.2 Type of inference — typ inference

Zadáním typu inference určíme způsob interpretace IF-THEN pravidel a zároveň i metodu přibližné dedukce. Máme k dispozici následující možnosti:

(a) *Logical deduction — logická dedukce* V tomto případě je jazykový popis přeložen na množinu logických implikací ve tvaru

$$\begin{aligned}&\mathbf{A}_{1,X_1,\ldots,X_n} \Rightarrow \mathbf{B}_{1Y},\\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\ &\mathbf{A}_{\mathbf{m}X_1,\ldots,X_n} \Rightarrow \mathbf{B}_{m,Y},\end{aligned}$$

kde $\mathbf{A}_{j,X_1,\ldots,X_n}$, $\mathbf{B}_{j,Y}$ jsou formální fuzzy reprezentace (můžeme na ně nahlížet jako na speciální formule) odpovídající jazykovým výrazům.

Zadané vstupní hodnoty $x_{10},\ldots,x_{n0}$ se následně transformují na *nejvhodnější* formuli, například na $\mathbf{A}_{k,X_1,\ldots,X_n}$. Tuto formuli lze chápat jako *percepci* zadané hodnoty. Pak se provede odvození pomocí pravidla modus

ponens ve tvaru

$$\frac{\mathbf{A}_{k,X_1,\ldots,X_n},\ \ \mathbf{A}_{k,X_1,\ldots,X_n} \Rightarrow \mathbf{B}_{m,Y}}{\mathbf{B}_{m,Y}},$$

jehož výsledkem je formule $\mathbf{B}_{m,Y}$. Ta se transformuje na fuzzy množinu $B_m \subseteq V$. Tuto fuzzy množinu nakonec defuzzifikujeme, abychom získali výstupní hodnotu y_0. Logická dedukce by se měla používat především s evaluačními jazykovými výrazy a jednou z defuzzifikačních metod DEE nebo SDEE.

Poznámka: Skutečná procedura je nepatrně pozměněna oproti výše popsané, neboť jak vstup $\mathbf{A}_{k,X_1,\ldots,X_n}$ tak i výstup $\mathbf{B}_{m,Y}$ mohou být mírně modifikovány.

2.1.3 Příklad

Uvažujme jazykový popis

$$R_1 : \mathsf{IF}\ X \ \mathsf{is}\ \textit{velmi malé}\ \mathsf{THEN}\ Y \ \mathsf{is}\ \textit{velmi malé}$$
$$R_2 : \mathsf{IF}\ X \ \mathsf{is}\ \textit{malé}\ \mathsf{THEN}\ Y \ \mathsf{is}\ \textit{malé}$$

Nechť interval $[0,1]$ je jazykovým kontextem proměnných X, Y . Například hodnoty 0.22 a 0.09 jsou obě malé, avšak 0.09 je zároveň velmi malá, zatímco 0.22 není. Nechť zadaná vstupní hodnota $x_0 = 0.09$. Na základě zadaného jazykového popisu předpokládáme, že výstupní hodnota y_0 bude velmi malá, tedy něco kolem $y_0 = 0.09$ (vzhledem k pravidlu R_1). Podobně očekáváme pro vstupní hodnotu $x_0 = 0.22$ malou, ale ne velmi malou výstupní hodnotu y_0, tedy něco kolem $y_0 = 0.22$. Toto chování je umožněno logickou dedukcí a defuzzifikační metodou DEE (SDEE). Pro lepší pochopení principu tohoto druhu přibližné dedukce je připraven jazykový popis MONOTONE.rb. Doporučujeme, abyste s ním provedli různé experimenty.

(b) *Fuzzy approximation with conjunctions — fuzzy aproximace s konjunkcemi*

Toto je velmi známá Mamdani-Assilianova metoda. Antecedent každého pravidla je zadán jako fuzzy množina $A_1 \times \cdots \times A_n \subseteq U_1 \times \cdots \times U_n$ a sukcedent jako fuzzy množina $B \subseteq V$. Disjunktivní normální forma příslušná k jazykovému popisu je transformována na fuzzy relaci

$$R = R_1 \cup \cdots \cup R_m,$$

kde každé R_j je fuzzy relace získaná z $A_{j1}, \ldots, A_{jn}$ a B_j, $j = 1, \ldots, m$ pomocí formule

$$R_j(x_1, \ldots, x_n, y) = A_{j1}(x_1) \wedge \cdots \wedge A_{jn}(x_n) \wedge B_j(y),$$

jelikož fuzzy IF-THEN pravidla charakterizují nějakou funkční závislost a tedy mohou být chápány jako konjunkce.

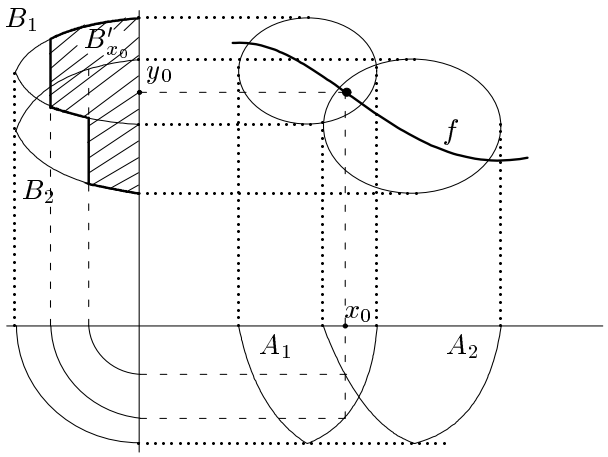
Nyní uvažujme vstupní hodnoty $x_1 = x_{10}, \ldots, x_n = x_{n0}$. Výstupní fuzzy množina je projekcí fuzzy relace R podle formule

$$B_{x_{10},\ldots,x_{n0}}(y) =$$

$$R(x_{10}, \ldots, x_{n0}, y) = \bigvee_{j=1}^{m} (A_{j1}(x_{10}) \wedge \cdots \wedge A_{jn}(x_{n0}) \wedge B_j(y)). \quad (1)$$

Cílem je aproximovat nějakou funkci f ukrytou ve fuzzy relaci R. Tvar fuzzy množin A_j and B_j a defuzzifikační metoda by měly být modifikovány tak, aby co nejlépe odpovídaly funkci f. Nejlepšího chování fuzzy aproximace s konjunkcemi dosáhneme tak, že fuzzy množiny specifikujeme jako fuzzy čísla a použijeme defuzzifikační metodu Center of Gravity (metoda těžiště).

Celou situaci dobře popisuje následující obrázek.



(c) *Fuzzy approximation with implications — fuzzy aproximace s implikacemi*

Utéto metody jsou pravidla interpretována jako logické implikace, avšak jejím cílem je aproximace funkce.

Antecedent každého pravidla je určen fuzzy množinou $A \underset{\sim}{\subseteq} U$ a sukcedent fuzzy množinou $B \underset{\sim}{\subseteq} V$. Konjunktivní normální forma přiřazená jazykovému popisu se transformuje na fuzzy relaci

$$R = R_1 \cap \cdots \cap R_m,$$

kde každé R_j je fuzzy relací získanou z $A_{j1}, \ldots, A_{jn}$ a B_j, $j = 1, \ldots, m$ pomocí formule

$$R_j(x, y) = (A_{j1}(x_1) \wedge \cdots \wedge A_{jn}(x_n)) \rightarrow B_j(y),$$

kde $\rightarrow$ je Łukasiewiczova implikace (vypočítaná pomocí formule $a \rightarrow b = \min(1, 1-a+b), a, b \in [0,1]$). Pro dané vstupní hodnoty $x_1 = x_{10}, \ldots, x_n = x_{n0}$ se výstupní fuzzy množina spočte jako projekce fuzzy relace R podle formule

$$B_{x_{10},\ldots,x_{n0}}(y) =$$

$$R(x_{10}, \ldots, x_{n0}, y) = \bigwedge_{j=1}^{m} (A_{j1}(x_{10}) \wedge \cdots \wedge A_{jn}(x_{n0}) \rightarrow B_j(y)). \quad (2)$$

Praktický význam této metody je spíše okrajový, protože je poměrně obtížné stanovit správný tvar fuzzy množin a defuzzifikační metodu, abychom dosáhli uspokojivého výsledku.

Poznamenejme, že ve všech případech může být daný jazykový popis použit ke generování nějaké funkce

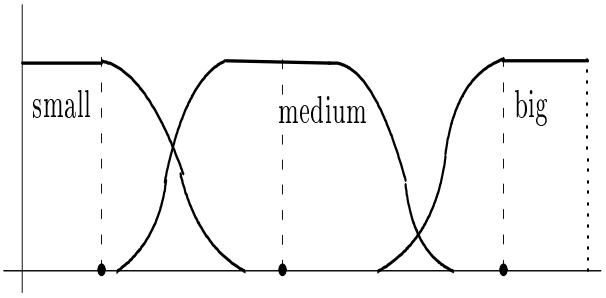
$$g : U_1 \times \cdots \times U_n \longrightarrow V.$$

2.1.4 Defuzzication — defuzzifikace

Určuje způsob defuzzifikace fuzzy množiny, tj. výsledku přibližné dedukce. Máme k dispozici následující defuzzifikační metody:

(a) *Simple defuzzification of Evaluating Expressions (SDEE) — jednoduchá defuzzifikace evaluačních výrazů*

Tato metoda je založena na předpokladu, že fuzzy množiny interpretující evaluační jazykové výrazy jsou ve tvaru S a Π křivek. Defuzzifikační metoda závisí na konkrétním typu fuzzy množiny. Defuzzifikovanou hodnotou je okraj jádra fuzzy množiny, pokud odpovídá fuzzy množině typu „small“ (malý) nebo „big“ (velký) a těžiště fuzzy množiny, jestliže tato odpovídá typu „medium“ (střední) (viz. obrázek níže).



(b) *Defuzzification of Evaluating Expressions (DEE) — defuzzifikace evaluačních výrazů*

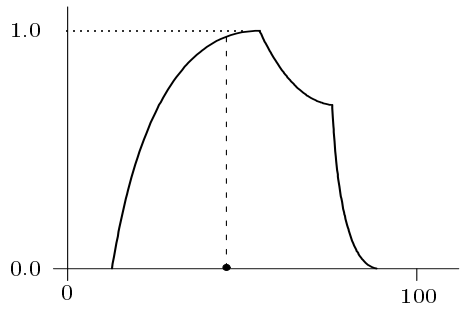
Metoda je modifikací metody SDEE; hodnota defuzzifikace se určí v okolí okraje jádra S-fuzzy množiny.

(c) *Center of Gravity Method (COG) — metoda těžiště*

Toto je klasická metoda daná formulí

$$\mathrm{DEF}(A) = \frac{\sum_{x_i \in U} A(x_i) x_i}{\sum_{x_i \in U} A(x_i)}.$$

Výsledek je znázorněn na následujícím obrázku.



(d) *Modified Center of Gravity Method (MCOG) — modifikovaná metoda těžiště*

Je modifikací předcházející metody, v níž se bere v úvahu pouze horní část fuzzy množiny.

(e) *Mean of Maxima Method (MOM) — metoda středu maxim*

Toto je další klasická metoda, která vezme za hodnotu defuzzifikace průměr ze všech hodnot s maximálním stupněm příslušností.

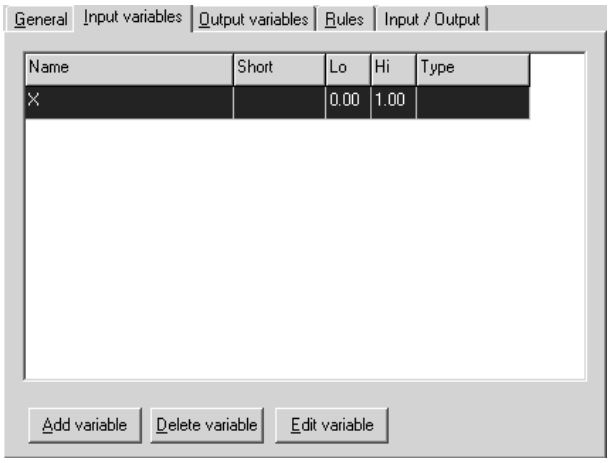
(f) *Smooth Defuzzification of Linguistic Expression* (SDLE — hladká defuzzifikace evaluačních výrazů)

Metoda vychází z teorie Hladké logické dedukce, která aplikuje Fuzzy transformaci (F-transformaci) na výstupy obdržené užitím Logické dedukce. Připomeňme, že výsledkem logické dedukce je po částech spojitá funkce. Nicméně, průběh výstupu z SDLE je hladký a spojitý, což je zaručeno vlastnostmi F-transformace.

2.1.5 Description — popis

V tomto okně si uživatel může zapsat poznámky k jazykovému popisu.

2.2 Input and output variables — Vstupní a výstupní proměnné



Tato Tab-záložka je určena k zadání proměnných použitých v jazykovém popisu. Ty jsou rozděleny na *input variables* (vstupní proměnné), což jsou proměnné vyskytující se v antecedentu IF-THEN pravidel, a *output variable* (výstupní proměnná) vyskytující se v sukcedentu.

Každá proměnná je charakterizovaná následujícími atributy.

- *Name* (Jméno) proměnné. Kromě celého jména je také možno definovat jeho zkratku, která se pak používá v záhlaví IF-THEN pravidel.
- *Upper* (horní) a *Lower* (dolní) ohraničení — specifikace intervalu (univerza), na kterém jsou definovány fuzzy množiny interpretující jazykové výrazy. Pokud používáme evaluační jazykové výrazy, hovoříme o *jazykovém kontextu*, protože tato ohraničení znamenají nejmenší a největší uvažovanou hodnotu v daném kontextu. Poznamenejme, že ve fuzzy regulaci se používá jiný název – *scaling*.

  Při zadání kontextu máme dvě možnosti. Buďto specifikujeme symetrický interval $[-u, u]$, kde u je nějaké reálné číslo, nebo stanovíme nějaký interval u_1, u_2, přičemž u_1, u_2 jsou reálná čísla splňující podmínku $u_1, u_2 \geq 0$. Symetrický interval se používá především ve fuzzy regulaci.
- *Discretization* (diskretizace univerza). Tento atribut určuje počet bodů univerza, v nichž se vypočítávají funkce příslušnosti. Čím je toto číslo vyšší, tím je výpočet přesnější, avšak na úkor výpočetní složitosti. Jestliže je specifikován symetrický kontext, potom diskretizace musí být liché číslo, abychom měli k dispozici přesný střed.

Předchozí atributy lze specifikovat pro každou proměnnou stlačením tlačítka **Edit variable** nebo **Add variable**. Označenou proměnnou můžeme smazat stlačením tlačítka **Delete variable**.

Toto okno také umožňuje editovat výrazy použitelné pro všechny proměnné ve fuzzy IF-THEN pravidlech. Stisknutím **Edit expressions** aktivujeme editaci pravidel.

Připomeňme, že proměnné ve fuzzy logice jsou jazykové. To znamená, že každá proměnná má hodnoty, které jsou obecně jazykové výrazy jako „malý“, „velmi velký“, „zhruba střední“ atd. V software LFLC 2000 je možno používat evaluační jazykové výrazy definované předem a také výrazy zadané uživatelem. Nastavení evaluačních jazykových výrazů je v souladu s jazykovou analýzou a psychologickým výzkumem. Mohou však být modifikovány, pokud je to třeba.

Jinou možností je zadání vlastních *uživatelských výrazů*. V tomto případě je na uživateli, aby si sám nastavil tvary funkcí příslušnosti a přiřadil jim jména.

2.2.1 Editování výrazů

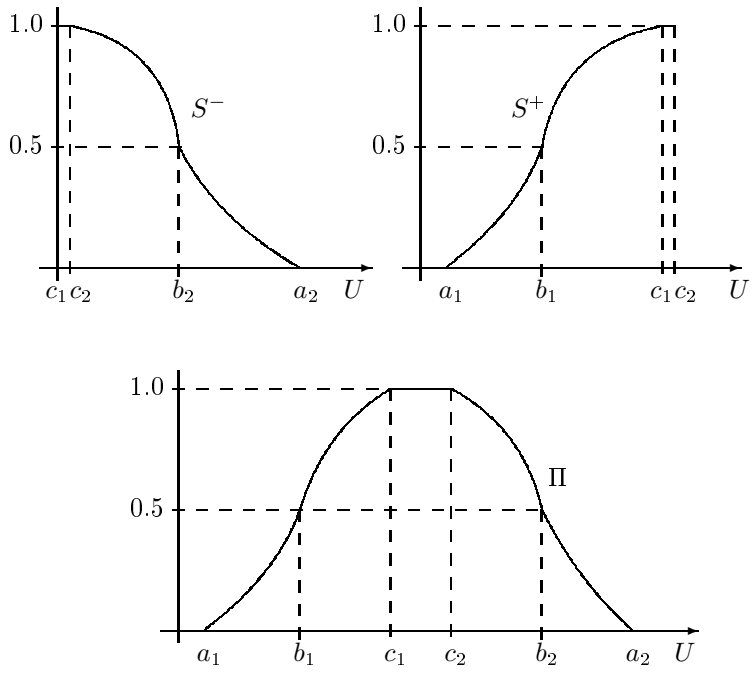
Stlačením **Edit expressions** se otevře okno pro editaci jazykových výrazů používaných k charakterizaci hodnot dané proměnné. Obsahuje dvě tab-záložky.

2.2.2 User - uživatelské

Zde zadáváme tvary funkcí příslušnosti jazykových výrazů definovaných výrazů. Tyto tvary jsou obecně dány funkcí

$$F(x,a_1,b_1,c_1,c_2,b_2,a_2)=\begin{cases}0, & x\le a_1 \text{ nebo } x\ge a_2\\ \frac{1}{2}\left(\frac{x-a_1}{b_1-a_1}\right)^2, & a_1<x<b_1\\ 1-\frac{1}{2}\left(\frac{c_1-x}{c_1-b_1}\right)^2, & b_1\le x<c_1\\ 1-\frac{1}{2}\left(\frac{x-c_2}{b_2-c_2}\right)^2, & c_2<x<b_2\\ \frac{1}{2}\left(\frac{a_2-x}{a_2-b_2}\right)^2, & b_2\le x<a_2\\ 1, & c_1\le x\le c_2,\end{cases}$$

Pomocí této funkce je možné vytvářet tři *standardní typy funkcí příslušnosti* většinou uvažované v teorii fuzzy množin, a to S^+, S^- a Π. První dva typy jsou speciálními případy třetího. Pokud nastavíme $a_1=b_1=c_1$, dostaneme fuzzy množinu typu S^-. Nastavením $c_2=b_2=a_2$ získáme fuzzy množinu typu S^+. Všechny tři fuzzy množiny jsou základem pro vyjádření významu standardních jazykových výrazů a jsou zobrazeny na následujícím obrázku:



Tuto funkci je možné zjednodušit na trapezoidální (danou parametry a_1, c_1, c_2, a_2) nebo pouze trojúhelníkovou ($c_1 = c_2$).

K nastavení některé z předchozích možností je nutné stlačit jedno z tlačítek **Add Quadratic**, **Add Trapezoid**, **Add Trangular**. Nejprve je třeba zadat jméno výrazu, které bude dále použito při zadání jazykového popisu v záložce Rules.

Tvar fuzzy množin lze nastavit buďto graficky, přesunutím libovolného parametru (malého čtverečku na křivce) pomocí myši, nebo explicitně v následujících sloupcích:

Left support	a_1
Left Equilibrium	b_1
Left Kernel	c_1
Right Kernel	c_1
Right Equilibrium	b_1
Right support	a_2

Již existující fuzzy množiny je možné kopírovat užitím kláves **Ctrl C** a **Ctrl V**. K tomu je třeba označit myší číslo fuzzy množiny na levé straně tabulky, což označí všechny její parametry. Následným stlačením **Ctrl C** přemístíme parametry do clipboardu. Stlačením **Ctrl V** docílíme překopírování výrazu. Para-

metry fuzzy množin lze kopírovat také z jedné proměnné do druhé. To můžeme provést dvěma způsoby:

- Označte jednu nebo více fuzzy množin (pomocí klávesy **Shift**), stlačte **Ctrl C**, přejděte na jinou proměnnou a stlačte **Ctrl V**.
- Všechny fuzzy množiny se automaticky kopírují z proměnné specifikované v nabídce "Copy from variable", která se zobrazí po stlačení tlačítka **Add variable**.

Již vytvořené fuzzy množiny mohou být smazány tlačítkem **Delete**.

Zajímavou možností, rozšířenou především ve fuzzy regulaci, je speciální funkce **Add Uniform**. Výsledkem je vytvoření n trojúhelníkových fuzzy množin (n zadává uživatel), které rovnoměrně pokryjí univerzum. Typický počet fuzzy množin je 7. Fuzzy množinám se pak při symetrickém kontextu většinou přiřadí názvy *negative big* (NB), *negative medium* (NM), *negative small* (NS), *zero* (ZE), *positive small* (PS), *positive medium* (PM) and *positive big* (PB).

Aktivní fuzzy množina je zobrazena červeně a lze ji měnit výše popsaným způsobem. Další fuzzy množiny jsou zobrazeny šedě, aby bylo viditelné rozvržení všech možných hodnot.

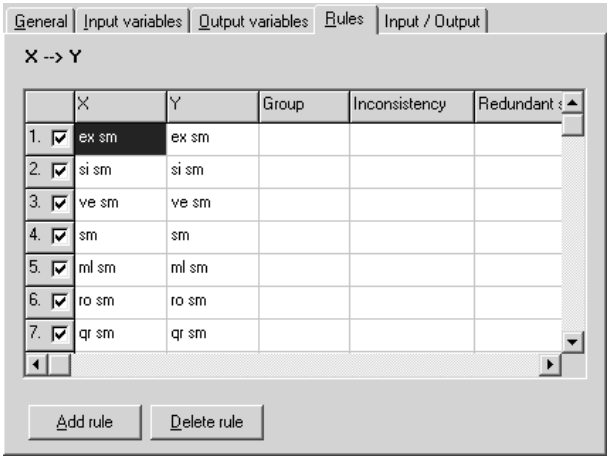
2.2.3 Standard

Tato tab-záložka slouží k zobrazení významů standardních jazykových výrazů, které jsou modelovány pomocí standardních funkcí příslušnosti. Zatím je není možné modifikovat.

2.2.4 Fuzzy numbers — fuzzy čísla

Fuzzy číslo je matematickým modelem nepřesných čísel, například "kolem 20", "velmi zhruba 100", "zhruba 300". Takové výrazy jsou velmi časté a lze konstatovat, že přesná čísla se v praxi téměř nepoužívají. Fuzzy čísla jsou modelována pomocí standardních funkcí příslušnosti.

2.3 Rules — pravidla



Tato záložka umožňuje editovat fuzzy IF-THEN pravidla vytvářející jazykový popis. Záložka je rozdělena do následujících sloupců:

- *Number of the rule*
  Na začátku každého řádku zobrazujícího pravidlo je zaškrtávací okénko společně s jeho pořadovým číslem. Slouží k aktivaci/deaktivaci daného pravidla. Jestliže je pravidlo neaktivní, není smazáno, ale nepoužívá se při inferenci. To umožňuje provádět různé experimenty, při nichž sledujeme vliv určitých pravidel na výsledek inference bez nutnosti jejich smazání.

- Dále pak následují sloupce pro každou *nezávislou* (antecedent) a *závislou* (sukcedent) proměnnou ,v nichž se pomocí výrazů přirozeného jazyka vytvářejí jednotlivá pravidla tvořící celý jazykový popis (viz editování pravidel).

- *Group* Tento sloupec obsahuje informaci o přítomnosti shodných pravidlech (pokud existují).

- *Inconsistency* Informuje o inkonzistenci pravidel, tj. o pravidlech, která mají stejný antecedent ale odlišný succedent.

- *Redundancy* Dále jsou k dispozici dva sloupce (pro antecedent a succedent), kde může uživatel zjistit, zda je pravidlo přebytečné či nikoli. Přitom může nastat jedna ze dvou následujících situací:

  (a) Sukcedent obou pravidel je stejný, ovšem antecedent prvního je užší než antecedent druhého. V tomto případě je první pravidlo nepotřebné a může být z jazykového popisu odstraněno.

  Uvažujme například dvě pravidla

RuleNo	X_1	X_2	Y
1	sm	bi	me
2	ve sm	si bi	me

Pravidlo 2 je přebytečné, protože když X_1 je velmi malé a X_2 je podstatně velké potom X_1 je také malé a X_2 je také velké. Jelikož obě pravidla mají stejný sukcedent, první pravidlo zastane i úlohu druhého.

(b) Antecedent obou pravidel je stejný, ale sukcedent prvního je užší než sukcedent druhého. Jelikož druhé pravidlo je přesnější, je první pravidlo přebytečné.

Příkladem jsou následující dvě pravidla

RuleNo	X_1	X_2	Y
1	sm	bi	bi
2	sm	bi	ve bi

Pravidlo 1 je přebytečné, protože velmi velké Y je přesnější než velké Y.

Rozhodnutí, zda by mělo být pravidlo odstraněno či nikoliv není provedeno automaticky, ale zůstává na uživateli.

Jazykový popis je definován prostřednictvím dostupných jazykových výrazů. V našem případě to jsou evaluační jazykové výrazy (standardní) nebo výrazy definované uživatelem.

2.3.1 Evaluační jazykové výrazy

Struktura evaluačních jazykových výrazů, které se používají v IF-THEN pravidlech, je následující:

$$[\langle\text{znaménko}\rangle]\langle\text{jazykový modifikátor}\rangle\langle\text{základní výraz}\rangle$$

Znaménko je + nebo −. Užívají se, pokud má příslušná proměnná symetrický kontext.

Základní výrazy jsou

výraz	zkratka	překlad
small	sm	malý
medium	me	střední
big	bi	velký

Jazykové modifikátory jsou

modifikátor	zkratka	překlad
extremely	ex	výrazně
significantly	si	značně
very	ve	velmi
rather	ra	spíše
more or less	ml	více méně
roughly	ro	zhruba
quite roughly	qr	spíše zhruba
very roughly	vr	velmi zhruba

Složitější výrazy mohou být vytvořeny spojováním jednodušších výrazů spojkami *and* (a) a *or* (nebo).

Můžeme použít také fuzzy čísla. Zapisují se následujícím způsobem:

$$\text{about}\langle\text{číslo}\rangle,$$

kde ⟨číslo⟩ je jakékoli číslo patřící do jazykového kontextu dané proměnné.

2.3.2 Editování pravidel

Předchozí výrazy mohou být použity kdekoli ve fuzzy IF-THEN pravidlech. Uživatel má k dispozici dvě možnosti zadání výrazů v pravidlech. Může ručně vepsat zkratky výrazů do odpovídajících řádků záložky **Rules**. Může si také zobrazit všechny výrazy a modifikátory tak, že z podmenu **View** hlavního menu zvolí odpovídající možnost. Výrazy pak vkládá jednoduchým stisknutím levého tlačítka myši na příslušné položky těchto menu.

2.3.3 Příklad

Uvažujme jazykový popis se dvěma nezávislými proměnnými *teplota* a *tlak*. Závislou proměnnou je *poloha* (regulovaného ventilu). Předpokládáme, že univerzum všech proměnných je symetrické. Nechť jsou dána následující pravidla:

IF *teplota* is *positive small* and *tlak* is *negative big*
THEN *poloha* is *positive medium*
IF *teplota* is *negative very small* and *tlak* is *positive*
significantly big THEN *poloha* is *positive roughly small*

Tento popis lze zapsat v editovacím okně pravidel následujícím způsobem:

teplota	tlak	poloha
+sm	− bi	+me
−ve sm	+si bi	+ro sm

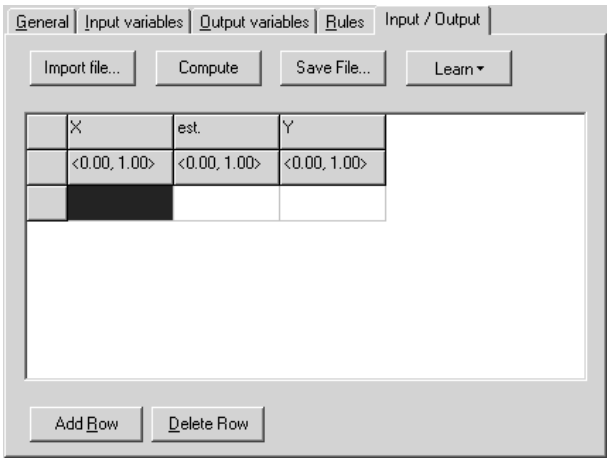
Pravidla je možné také vyjímat, kopírovat a vkládat. Toho můžeme dosáhnout označením jednoho nebo více pravidel (pomocí klávesy **Shift**) a kliknutím na číslo pravidla. Kopíruje se stlačením kláves **Ctrl C**. K vyjmutí slouží **Ctrl X** a vkládá se **Ctrl V**.

V mnoha případech potřebujeme oddělit pravidla popisující kladné a záporné případy. Např. ve fuzzy regulaci potřebujeme hodnoty pod a nad žádanou hodnotou. Tyto pravidla se liší pouze znaménky u jednotlivých evaluačních výrazů. Práci si lze zjednodušit použitím *Invert sign* $+, -$. Invertovat znaménka je možné po označení příslušných pravidel (a případně i překopírování) a stlačení kláves **Ctrl I**. Celý výše uvedený postup může být proveden také prostřednictvím podmenu **Edit**.

2.3.4 Třídění pravidel

Pro lepší orientaci v jazykovém popisu lze pravidla setřídit. Třídění se provádí kliknutím na příslušný název proměnné. Pokud klikneme na více proměnných, dostaneme uspořádání v pořadí, jak jsme na proměnné kliknuli. Uspořádání se provádí vzhledem k hodnotě (od *small* k *big*) a ostrosti jazykového operátoru (od *extremely* k *very roughly*).

2.4 Input/Output (Vstup/Výstup)



V této záložce je možné pracovat s daty uloženými v textovém souboru. Máme dvě možnosti:

- Vypočítat hodnoty funkce g generované jazykovým popisem (viz. odstavec Typ inference).

- Učení (learning) jazykového popisu z dat (v externí databázi nebo z přímo zadaných hodnot).

2.4.1 Funkce Compute

Před použitím funkce *Compute* je třeba zadat jazykový popis. Pak můžeme importovat textový soubor stlačením **Import file**. Jeho struktura je

Text line 1	Var X_1	Var X_2	...	Var X_n	Var Y
Text line 2	xxxx	xxxx	xxxx	xxxx	xxxx
Data line 1	999.99	999.99	999.99	999.99	999.99
⋮	⋮	⋮	⋮	⋮	⋮
Data line m	999.99	999.99	999.99	999.99	999.99

Každý sloupec odpovídá jedné proměnné (včetně nezávislé). Jejich počet závisí na definici jazykového popisu a na definici proměnných na tab-záložkách *input, output variables*.

V importovaném souboru může být libovolný počet textových řádků, které jsou při načítání ignorovány. Data jsou desetinná čísla libovolné délky oddělena nejméně jednou mezerou. Jestliže soubor obsahuje více sloupců, než je počet definovaných proměnných v jazykovém popisu, potom se sloupce navíc ignorují.

V souboru obsažená nezávislá proměnná se používá pouze pro porovnání a její hodnoty jsou zobrazeny ve sloupci pojmenovaném „est“ (estimation — odhad). Sloupec Y je určen pro vypočtené hodnoty funkce g pro každý záznam (tj. každý řádek) importovaného souboru. Výpočet se provede po stlačení tlačítka **Compute**. Všimněte si, že vypočtené hodnoty by měly padnout do definovaného jazykového kontextu. Pokud tomu tak není, výsledkem je příslušná dolní nebo horní mez.

Importovaný soubor lze modifikovat. Je možné přídávat nové řádky tlačítkem **Add Row** a mazat **Delete Row** a uložit **Save File**. Uložený soubor má příponu '.out'.

2.4.2 Automatické generování jazykového popisu

Ještě před spuštěním této metody je nutné definovat všechny nezávislé proměnné i závislou proměnnou a jazykový kontext. Pak načteme soubor s daty pomocí tlačítka **Import file**. Počet sloupců souboru nesmí být menší, než je počet všech proměnných. Nyní lze aplikovat metodu automatického generování pravidel, tzv. metodu automatického učení **Learn/Linguistic learning**. Nová pravidla se vloží do jazykového popisu, který je uložen pod jménem, které je zadáno před spuštěním učící procedury.

Linguistic learning method

Je založena na metodě hledání typického evaluačního jazykového výrazu k dané hodnotě v daném jazykovém kontextu. Interval $[u_1, u_2]$ možných hodnot dané proměnné je *fuzzy* rozdělen na podintervaly „extremely small“, „very small“, „roughly small“ atd., podobně pro „medium“ a „big“. Jestliže daná hodnota padne do některého podintervalu, je tento chápán jako typický reprezentant odpovídající jazykové hodnoty.

Každá hodnota proměnné ve vstupním souboru je nahrazena jazykovým výrazem pomocí výše popsané metody. Výsledkem je nové jazykově charakterizované fuzzy IF-THEN pravidlo.

2.4.3 Příklad

Nechť proměnné X_1, X_2, Y mají kontexty $[1, 10]$, $[-5, 5]$, $[0, 1]$. Datový řádek je

$$1.7 \qquad -3.25 \qquad 0.83$$

Typické jazykové výrazy nalezené výše popsanou metodou jsou v následující tabulce:

kontext	hodnota	typický výraz
$[1, 10]$	1.7	very small
$[-5, 5]$	-3.25	roughly big
$[0, 1]$	0.83	big

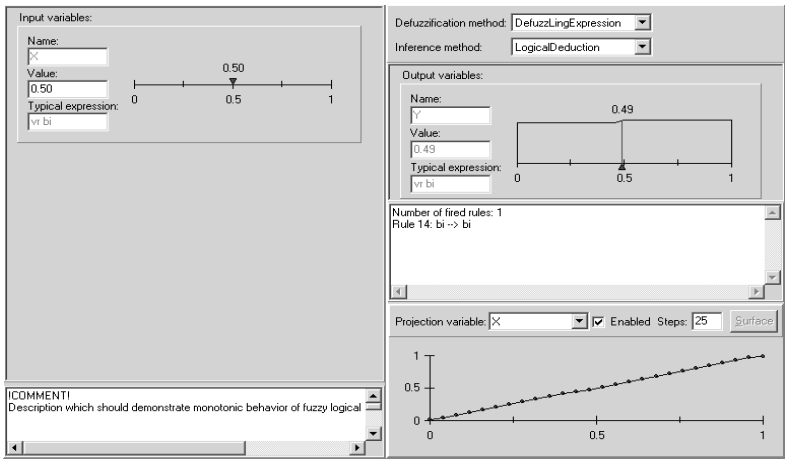
Nové pravidlo je

IF X_1 is *very small* AND X_2 is *roughly big* THEN Y is *big*.

Opakující se pravidla se automaticky odstraňují. Další redukce popisu může být provedena buď ručně nebo poloautomaticky pomocí tlačítek na tab-záložce Rules.

Další metody učení v současnosti připravujeme.

3 Testování jazykového popisu



Tato funkce umožňuje testovat chování vytvořeného jazykového popisu. Aktivuje se tlačítkem **Test**. Testovací okno je rozděleno na čtyři části.

3.1 Levá horní část

Zde se nastavuje vstup pro každou definovanou nezávislou proměnnou, a to následujícími způsoby:

- Vstupní hodnota se nastavuje graficky na vodorovné čáře definované pro každou antecedentovou proměnnou. Požadované hodnoty dosáhneme posunem malého červeného trojúhelníku podél čáry na odpovídající pozici.
- Konkrétní hodnotu každé proměnné zadáme explicitně ve vstupním políčku. Této možnosti většinou využíváme v případě, kdy nelze graficky nastavit dostatečně přesnou hodnotu.

Ke každé dané hodnotě se zobrazuje typický evaluační jazykový výraz, tj. jazykový výraz, který nejlépe charakterizuje danou hodnotu v daném jazykovém kontextu.

3.2 Levá dolní část

Tato část je určena k zobrazení komentáře příslušejícího zpracovávanému jazykovému popisu. Poznamenejme, že uložený jazykový popis lze prohlížet nebo editovat v libovolném textovém editoru (nedoporučujeme!).

3.3 Pravá horní část

Zde se zobrazuje výsledek inference v závislosti na zvolené inferenční a defuzzifikační metodě. Přesněji řečeno, je zde zobrazen tvar výsledné fuzzy množiny s označenou defuzzifikovanou hodnotou, typickým evaluačním výrazem a všemi pravidly použitými v inferenci.

Je možné experimentovat s různými druhy inference a defuzzifikace a sledovat jejich vliv na změnu výsledné křivky.

3.4 Pravá dolní část

Zde se zobrazují výsledky inference pro všechny možné hodnoty zvolené nezávislé proměnné.

- Pokud používáme více než jednu nezávislou proměnnou, dostaneme dvourozměrnou projekci funkce g vytvořené jazykovým popisem (viz. odstavec Typ inference) pro všechny hodnoty zvolené proměnné a zafixované hodnoty ostaních proměnných.

  Proměnná, vůči které se projekce počítá, se zadává volbou v rolovacím menu *Projection variable*. Protože samozřejmě nemůžeme vypočítat výsledky pro všechny možné hodnoty, musíme určit počet ekvidistantních uzlů z množiny všech hodnot zadané proměnné. Učiníme tak zadáním počtu uzlů v políčku *Enabled steps*.

- Lze také zobrazit třídimenzionální projekci funkce g tlačítkem **Surface**. Obě proměnné, vůči kterým se projekce bude počítat a počet uzlů se nastavují analogicky jako v předchozím případě. Zobrazenou plochou můžeme otáčet pomocí myši, pohybem barevných bodů na konci každé souřadnicové osy.

4 Využití v jiných aplikacích

4.1 LFLC 2000 COM Server

Programátorům je k dispozici speciální COM objekt RuleBaseCOM, který poskytuje inferenční mechanismus fungující na bázi jazykového popisu navrhnutého v LFLC 2000. Před použitím v jakékoli aplikaci je nutno COM objekt registrovat v operačním systému Windows. Registrace se automaticky provede při instalaci softwarového balíku LFLC 2000.

RuleBaseCOM objekt se nainstaluje do `RBaseCOM` podadresáře základního adresáře LFLC 2000. Tento podadresář obsahuje následující soubory:

`register.bat`	dávkový soubor pro registraci RuleBaseCOM objektu do systému Windows. Používá Borland Inprise utilitu „`tregsvr.exe`“. Registrace musí být provedena před použitím objektu v jakékoli aplikaci. Pro registraci spusťte tento soubor.
`RBaseCOM.dll`	soubor obsahující RuleBaseCOM objekt.
`RBaseCOM.tlb`	Type library pro importování objektu do programovacího prostředí IDE.
`tregsvr.exe`	Borland Turbo Register Server – COM Server Registration utilita.

Jestliže COM objekt nebyl během instalace registrován, spusťte soubor “`register.bat`”.

Po úspěšné registraci je RuleBaseCOM objekt připraven k použití ve vašich aplikacích. Komunikační rozhraní objektu je následující:

LoadFromFile(BSTR FileName)	Načte jazykový popis se zadaným jménem *FileName*
int NumInputVars()	Vrátí počet vstupních proměnných načteného jazykového popisu
double HiBoundOfVar (**int** VarIndex)	Vrátí horní mez kontextu proměnné s *VarIndex*
double LoBoundOfVar (**int** VarIndex)	Vrátí dolní mez kontextu proměnné s *VarIndex*
BSTR VarName (**int** VarIndex)	Vrací jméno proměnné s *VarIndex*
double Inference (TVariantInParam Inputs)	Provede inferenci pro hodnoty *Inputs*

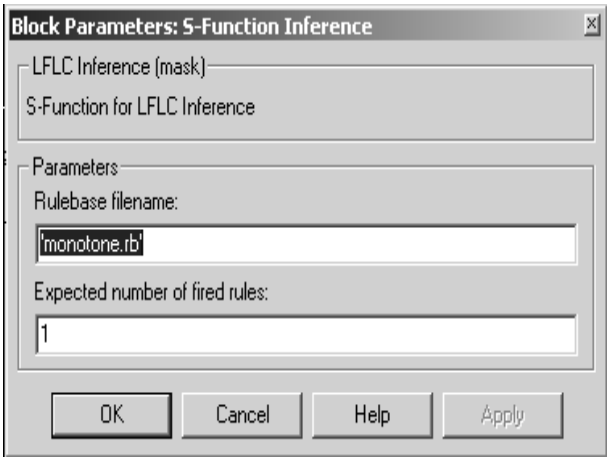
Pro pochopení použití COM objektu jsme připravili dva příklady, které jsou součástí LFLC 2000 a jsou umístěny v adresáři `LFLC 2000/RBaseCOM/examples`. Demonstrují použití RuleBaseCOM objektu ve vývojovém prostředí Borland C++ Builder 5.0.

ConsoleApp	Ukazuje jak se používá RuleBaseCOM v řádkové aplikaci
WinGUIApp	Znázorňuje použití RuleBaseCOM ve Windows GUI aplikaci

4.2 LFLC 2000 MATLAB/Simulink klient

Tento klient je speciální MATLAB/Simulinkovou funkcí, který se používá ve schématech Simulinku. Funkce tvořící nový blok “LFLC Inference” je obsažena

v souboru LFLCLibrary.mdl nacházející se v adresáři `LFLC 2000/matlab`.



V prostředí Simulink se zadávají parametry prostřednictvím dialogového okna Block Parameters S-Funkce Inference jak je zobrazeno na předchozím obrázku. První atribut je jmeno souboru obsahující jazykový popis (soubor „`xxx.rb`“) a druhý je počet zobrazených pravidel na jejichž základě byl vyhodnocen výstup.

Několik demonstračních souborů je k dispozici v `LFLC 2000/examples`.

Upozornění: LFLC 2000 MATLAB/Simulink klient používá LFLC 2000 COM Server RuleBaseCOM, který tedy musí být registrován v systému Windows před použitím klienta ve vývojovém prostředí Matlabu.

4.3 LFLC 2000 MATLAB klient

Součástí instalace je také MATLAB mex-funkce, kterou je možno použít v matlabovských programech nebo simulacích. Jméno funkce je LFLCInfer. Účelem funkce je spočítat výsledek inference na základě vstupních dat. LFLCInfer mex-Funkce používá následující konvenci. Před výpočtem inference je nutné funkci inicializovat načtením příslušného jazykového popisu („`xxx.rb`“ souboru). Funkce LFLCInfer má dva vstupní parametry.

- Prvním parametrem je číslo popisu. Díky tomuto číslu je možno pracovat s několika jazykovými popisy najednou. Číslo může nabývat hodnot od 0 po 9, tedy lze pracovat s deseti popisy současně. Pokud se načítá pouze jeden popis, můžeme tento parametr vynechat a implicitně se nastaví hodnota 0.
- Druhý parametr má dva různé účely, které se rozpoznají v závislosti na typu parametru. Parametr typu řetězec určuje jméno souboru, jenž bude

načten s celou cestou, a inicializuje příslušný jazykový popis. Pokud je druhý parametr jednorozměrné pole hodnot typu double, je rozpoznáno jako pole vstupních hodnot pro inferenci, jejíž výstupní hodnota je výstupem této LFLCInfer mex-Funkce.

Volání funkce LFLCInfer ve vývojovém prostředí Matlabu ilustruje následující příklad:

```
% načtení jazykového popisu
% ID číslo popisu je vynecháno

LFLCInfer('twovar.rb')

% spočtení hodnoty inference
result = LFLCInfer([-0.5, -0.5])

% stejného výsledku je dosaženo i v případě,
% kdy specifikujeme ID

result = LFLCInfer(0, [-0.5, -0.5])

% pokud použijeme více popisů zároveň,
% musíme použít ID číslo jazykového popisu

LFLCInfer(1, 'monotone.rb')

% výstup jedné inference lze přímo použít
% jako vstup inference jiné

result = LFLCInfer(1, LFLCInfer([-0.5, -0.5]))

% abychom získali informaci o použitém pravidle,
% je nutno změnit změnit typ požadovaného výsledku result na pole
[result, rule] = LFLCInfer([-0.5, -0.5])

% občas se používá více než jedno pravidlo v průběhu inferenční
% metody, přičemž informace o nich obdržíme obdobným způsobem
[result, rule1, rule2, rule3] = LFLCInfer([-0.5, -0.5])

% pokud obdržíme v předchozím příkladu pro rule2 a rule3 nulové
% hodnoty, znamená to, že během inference bylo použito pouze
% jedno pravidlo.
```

Upozornění: Stejně jako LFLC 2000 MATLAB/Simulink klient i LFLC 2000 MATLAB klient používá LFLC 2000 COM Server RuleBaseCOM, který tedy

musí být registrován v systému Windows před použitím klienta ve vývojovém prostředí Matlabu.