Extending the Expressiveness of Fuzzy Logic Languages

Mejoras de la Expresividad de Lenguajes Lógicos con el Manejo de Información Negativa y Difusa

PhD Thesis

Víctor Pablos Ceruelo

The Babel Research Group, Facultad de Informática
Universidad Politécnica de Madrid, Spain
http://babel.ls.fi.upm.es/

June 2015

Motivation & Overview

- Frameworks for modelling fuzzy logic are more focused in efficiency that in functionality (negation, fuzzification functions, default values, similarity, ...) and they are (usually) ad hoc solutions.
- They do not implement any semantics allowing to ensure that the subset of real-world cases are represented by a rule (as the multi-adjoint semantics).
- They do not allow to reuse the huge amount of information that companies have in non-fuzzy databases.

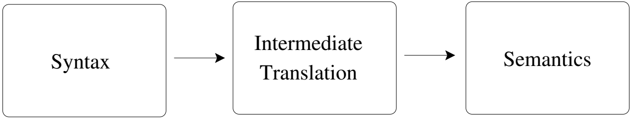
Motivation & Overview

- Frameworks for modelling fuzzy logic are more focused in efficiency that in functionality (negation, fuzzification functions, default values, similarity, ...) and they are (usually) ad hoc solutions.
- They do not implement any semantics allowing to ensure that the subset of real-world cases are represented by a rule (as the multi-adjoint semantics).
- They do not allow to reuse the huge amount of information that companies have in non-fuzzy databases.

RFuzzy: A framework for modelling fuzziness in logic programming programs.

Motivation & Overview

Motivation & Overview



RFuzzy Syntax

RFuzzy Syntax - DB connection

Database connection

$$\begin{aligned}
&\%\ \textit{Defines the database location, username and passwd} \\
&sql_persistent_location(database_id, \\
&\qquad db('SQL', user, pass, 'host' : port)). \qquad (1)
\end{aligned}$$

$$\begin{aligned}
&\%\ \textit{Example} \\
&sql_persistent_location(myDatabase, \\
&\quad db('SQL', 'me', 'myPass', 'localhost' : 1521)). \qquad (2)
\end{aligned}$$

$$\begin{aligned}
&\%\ \textit{Links a Prolog predicate to a database table} \\
&:-sql_persistent(\\
&\qquad predicate_name(Prolog\ type\ for\ each\ column), \\
&\qquad database_table_name(columns'\ names), \\
&\qquad database_id). \qquad (3)
\end{aligned}$$

$$
\begin{aligned}
&\%\ Example\ of\ linkage\ with\ restaurant's\ food\ type\ db\ table\\
&:-sql\_persistent(\\
&\qquad rest\_food\_type(integer, string),\\
&\qquad restaurant\_food\_type(id,\ food\_type),\\
&\qquad myDatabase). \qquad (4)\\
&\%\ Example\ of\ linkage\ with\ restaurant's\ distance\ to\ the\ center\\
&\%\ database\ table\\
&:-sql\_persistent(\\
&\qquad rest\_dist\_to\_tcc(integer, integer),\\
&\qquad restaurant\_dist\_to\_tcc(id,\ dist\_to\_tcc),\\
&\qquad myDatabase). \qquad (5)\\
&\qquad (6)
\end{aligned}
$$

RFuzzy Syntax - DB connection

Database connection

$$
\begin{aligned}
&\%\ \mathit{Example\ how\ to\ link\ two\ tables\ together}\\
&\mathit{restaurant}(\mathit{Id},\ \mathit{Food\_type},\ \mathit{Dist\_to\_tcc}) :-\\
&\qquad \mathit{rest\_food\_type}(\mathit{Id},\ \mathit{Food\_type}),\\
&\qquad \mathit{rest\_dist\_to\_tcc}(\mathit{Id},\ \mathit{Dist\_to\_tcc}).
\end{aligned} \tag{7}
$$

$$
\begin{aligned}
&\%\ \mathit{Example\ one\ table\ with\ all\ the\ information}\\
&:-\mathit{sql\_persistent}(\\
&\qquad \mathit{restaurant}(\mathit{integer},\ \mathit{string},\ \mathit{integer}),\\
&\qquad \mathit{restaurant}(\mathit{id},\ \mathit{food\_type},\ \mathit{dist\_to\_tcc}),\\
&\qquad \mathit{myDatabase}).
\end{aligned} \tag{8}
$$

$$ \tag{9} $$

$$
\begin{aligned}
&\%\ \textit{Another example of linkage with a database table}\\
&:-sql\_persistent(\\
&\quad restaurantAux(integer, string, string, string,\\
&\qquad integer, integer\ integer, integer),\\
&\quad restaurant(restaurant\_id, name,\\
&\qquad restaurant\_type, food\_type,\\
&\qquad years\_since\_opening,\\
&\qquad distance\_to\_the\_city\_center,\\
&\qquad price\_average, menu\_price),\\
&\quad myDatabase). \qquad (10)\\
& \qquad (11)
\end{aligned}
$$

$$
\begin{aligned}
&\%\ Removal\ of\ column\ (Restaurant\_id)\\
&restaurant(Name,\ Restaurant\_type,\\
&\qquad Food\_type, Years\_since\_opening,\\
&\qquad Distance\_to\_the\_city\_center,\\
&\qquad Price\_average,\ Menu\_price) :-\\
&\quad restaurant(Restaurant\_id,\\
&\qquad\quad Name,\ Restaurant\_type,\\
&\qquad\quad Food\_type,\ Years\_since\_opening,\\
&\qquad\quad Distance\_to\_the\_city\_center,\\
&\qquad\quad Price\_average,\ Menu\_price).
\end{aligned}
\tag{12}
$$

RFuzzy Syntax - VDBT

Definition of the virtual database table

$$\%\ Definition\ of\ virtual\ database\ table$$
$$define\_database(pT/pA, [(pN, pT')]). \quad (13)$$

$$\%\ Example\ of\ definition\ of\ virtual\ database\ table$$
$$\begin{aligned}
& define_database(restaurant/7, [\\
& \quad (name,\ string_type),\\
& \quad (restaurant_type,\ enum_type),\\
& \quad (food_type,\ enum_type),\\
& \quad (years_since_opening,\ integer_type),\\
& \quad (distance_to_the_city_center,\ integer_type),\\
& \quad (price_average,\ integer_type),\\
& \quad (menu_price,\ integer_type)]).
\end{aligned} \quad (14)$$

RFuzzy Syntax - VDBT

Definition of the virtual database table - example cont.

$$
\begin{aligned}
&\%\ \textit{Predicates available from the previous example}\\
&name(R,\ Value) :-\ \ldots\\
&restaurant\_type(R,\ Value) :-\ \ldots\\
&food\_type(R,\ Value) :-\ \ldots\\
&years\_since\_opening(R,\ Value) :-\ \ldots\\
&distance\_to\_the\_city\_center(R,\ Value) :-\ \ldots\\
&price\_average(R,\ Value) :-\ \ldots\\
&menu\_price(R,\ Value) :-\ \ldots
\end{aligned} \tag{15}
$$

RFuzzy Syntax - VDBT

Definition of the virtual database table - Comparison operators available by the things characteristics' type

type	operator	meaning of operator
boolean_type	=	"is equal to"
	=/=	"is different from"
enum_type	=	"is equal to"
	=/=	"is different from"
	=~=	"is similar to"
interger_type	=	"is equal to"
float_type	=/=	"is different from"
	>	"is bigger than"
	<	"is lower than"
	>=	"is bigger than or equal to"
	=<	"is lower than or equal to"
string_type	=	"is equal to"
	=/=	"is different from"

RFuzzy Syntax - Declarations

Declarations: Definition of the truth value of an individual or a set of individuals

$$fPredName(pT) :\sim value(TV) \quad (16)$$
$$if(\ pN(pT) \ \ comp \ \ value) \quad (17)$$
$$with\_credibility(credOp, \ credVal) \quad (18)$$
$$only\_for\_user \ 'UserName' \quad (19)$$

$$cheap(restaurant) :\sim value(0.5) \quad (20)$$

- *comp* is a comparison operator and can take the values "*is_equal_to*", "*is_different_from*", "*is_bigger_than*", "*is_lower_than*", "*is_bigger_than_or_equal_to*" and "*is_lower_than_or_equal_to*"
- *value* can be of type *integer_type*, *enum_type* or *string_type*.

RFuzzy Syntax - Declarations

Declarations: Definition of the truth value of an individual or a set of individuals (examples)

$$cheap(restaurant) :\sim value(0.1) \quad if(name(restaurant)\ is\_equal\_to\ zalacain). \tag{21}$$

$$cheap(restaurant) :\sim value(0.3) \quad if(name(restaurant)\ is\_equal\_to\ don\_jamon) \quad with\_credibility(min,\ 0.8). \tag{22}$$

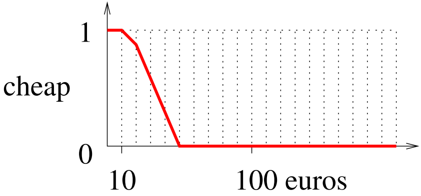
$$close\_to\_the\_city\_center(restaurant) :\sim value(0) \quad if(name(restaurant)\ is\_equal\_to\ zalacain) \quad only\_for\_user\ \ 'Lara' \tag{23}$$

RFuzzy Syntax - Dyn. Decl.

Dynamic Declarations: Definition of fuzzification functions

$$fPredName(pT) :\sim function(\ pN(\ pT\ ),\ [\ (valIn,\ valOut)\ ]). \qquad (24)$$

$$cheap(restaurant) :\sim function(\ price\_average(restaurant),\ [(0,\ 1),(10,\ 1),(20,\ 0.9),(50,\ 0),(200,0)]). \qquad (25)$$



RFuzzy Syntax - Defaults

Definition of default truth values

$$fPredName(pT) :\sim defaults\_to(TV) \tag{26}$$

$$close\_to\_the\_city\_center(restaurant) :\sim defaults\_to(0). \tag{27}$$

$$cheap(restaurant) :\sim defaults\_to(0.5). \tag{28}$$

$$fPredName(pT) :\sim rule(fPredName2(pT)) \tag{29}$$

$$\begin{aligned} fPredName(pT) :\sim rule(aggr, (&fPredName2(pT),\\ &fPredName3(pT),\\ &\ldots)) \end{aligned} \tag{30}$$

$$\begin{aligned} &\%\ It\ is\ a\ tempting\ restaurant\ if\ the\ worst\ value\ between\\ &\%\ being\ close\ to\ the\ center\ and\ being\ cheap\ is\ not\ so\ bad.\\ &tempting\_restaurant(restaurant) :\sim rule(\ min,\\ &\qquad (\ \ close\_to\_the\_city\_center(restaurant),\\ &\qquad\ \ cheap(restaurant)\ )) \end{aligned} \tag{31}$$

RFuzzy Syntax - Synonyms

Definition of a fuzzy characteristic as synonym or antonym of another one

$$\begin{aligned} & fPredName(pT) :\sim \\ & \quad synonym\_of(\ fPredName2(pT)) \end{aligned} \tag{32}$$

$$\begin{aligned} & unexpensive(restaurant) :\sim \\ & \quad synonym\_of(cheap(restaurant)) \\ & \quad with\_credibility(prod,\ 0.9). \end{aligned} \tag{33}$$

$$\begin{aligned} & fPredName(pT) :\sim \\ & \quad antonym\_of(fPredName2(pT)). \end{aligned} \tag{34}$$

$$\begin{aligned} & expensive(restaurant) : \\ & \quad antonym\_of(cheap(restaurant)). \end{aligned} \tag{35}$$

RFuzzy Syntax - Similarity

Definition similarity between attributes of the individuals in the database

$$similarity\_between(pT,\ pN(value1),\ pN(value2),\ TV). \quad (36)$$

$$similarity\_between(\ restaurant,\ food\_type(mediterranean),\ food\_type(spanish),\quad 0.8). \quad (37)$$

$$\%\ Tapas\ restaurants\ are\ valid\ answers\ if\ we\ look$$
$$\%\ for\ a\ spanish\ restaurant,\ but\ not\ in\ the\ other\ way.$$
$$similarity\_between(\ restaurant,\ food\_type(spanish),\ food\_type(tapas),\quad 0.7)\ with\_credibility(prod,\ 1). \quad (38)$$

RFuzzy Syntax - Others

New connective (conjunctor, disjunctor, aggregator) definition

$$define\_connective(Name/3,\ Var\_In\_1,\ Var\_In\_2,\ Var\_Out) :- prolog\_code. \quad (39)$$

$$define\_connective(max\_but\_at\_most\_a\_half/3,\ TV\_In\_1,\ TV\_In\_2,\ TV\_Out) :- max(TV\_In\_1,\ TV\_In\_2,\ TV\_Aux),\ min(TV\_Aux,\ 0.5,\ TV\_Out). \quad (40)$$

RFuzzy Syntax - Others

New modifier definition

$$define\_modifier(Name/2,\ Var\_In,\ Var\_Out) :- \\ prolog\_code. \tag{41}$$

$$define\_modifier(a\_little/2,\ TV\_In,\ TV\_Out) :- \\ TV\_Out\ *\ TV\_Out\ .=.\ TV\_In. \tag{42}$$

RFuzzy Syntax - Others

New negation operator definition

$$define\_negation\_op(Name/2,\ Var\_In,\ Var\_Out) :- \\ prolog\_code. \tag{43}$$

$$define\_negation\_op(godel\_neg/2,\ TV\_In,\ TV\_Out) :- \\ (\ (TV\_In\ .=.\ 0,\ \ TV\_Out\ .=.\ 1)\ \ ; \\ (\backslash + (TV\_In\ .=.\ 0),\ \ TV\_Out\ .=.\ 0)). \tag{44}$$

RFuzzy High Level Semantics

RFuzzy High Level Semantics

Intermediate translation

The structures with low level semantics are:

$$A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} @_j(D_1, \ldots, D_i, \ldots, D_n) \quad \text{if} \quad COND(A) \tag{45}$$

$$A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} D \quad \text{if} \quad COND(A) \tag{46}$$

$$A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} (\mathrm{p}',\ \mathrm{v}') \quad \text{if} \quad COND(A) \tag{47}$$

RFuzzy High Level Semantics - Declarations

Translation of the definition of truth value for an individual or a set of individuals

$$fPredName(pT) :\sim value(TV) \tag{48}$$

$$if(\ pN(pT)\ \ comp\ \ value) \tag{49}$$

$$with\_credibility(credOp,\ credVal) \tag{50}$$

$$only\_for\_user\ 'UserName' \tag{51}$$

is translated into

$$fPredName(Individual) \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} (1,\ TV) \quad \text{if}\ \ COND \tag{52}$$

where the "by default" values for p, v, $\&_i$ and $COND$ are 0.8, 1, *prod* (product) and *true*.

RFuzzy High Level Semantics - Declarations

Summary of the values given "by default" to the variables p, v, $\&_i$, $@_j\ (B_1, \ldots, B_n)$ and $COND$.

construction	p	v	$\&_i$	$@_j\ (B_1, \ldots, B_n)$	$COND$
fuzzy value	0.8	1	prod	TV	true
synonym	0.8	1	prod	TV	true
antonym	0.8	1	prod	TV	true
fuzzification function	0.6	1	prod	$pN(Individual) * \frac{(valOut\_2 - valOut\_1)}{(valIn\_2 - valIn\_1)}$	$valIn\_1 < pN(Individual) \leq valIn\_2$
fuzzy rule	0.4	1	prod	$@_j\ (B_1, \ldots, B_n)$	true
default fuzzy value	0	1	prod	TV	true

RFuzzy High Level Semantics - Declarations

Changes in the values given to the variables p, v, $\&_i$ and *COND* when the tails' constructions in eqs. 17, 18, 19 are used.

tail constr.	p	v	$\&_i$	$COND$
eq. 17	$p + 0.05$	v	$\&_i$	$COND \wedge (\ pN(Individual) \ \ comp \ \ value\ )$
eq. 18	p	$credVal$	$credOp$	$COND$
eq. 19	$p + 0.1$	v	$\&_i$	$COND \wedge currentUser(Me) \wedge Me \ = \ 'UserName'$

$$fPredName(pT) :\sim value(TV)$$

$$if(\ pN(pT) \ \ comp \ \ value) \tag{53}$$

$$with\_credibility(credOp, \ credVal) \tag{54}$$

$$only\_for\_user \ 'UserName' \tag{55}$$

$$fPredName(pT) :\sim function(\ pN(\ pT\ ),\ [\ (valIn,\ \ valOut)\ ]). \tag{56}$$

is translated into a set of clauses, one for each piece the programmer has defined for the piecewise function. Each one of the resulting clauses has the form

$$fPredName(Individual) \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} (1,\quad pN(Individual) \ * \ \frac{(valOut\_2 - valOut\_1)}{(valIn\_2 - valIn\_1)}) \quad \text{if} \quad COND \tag{57}$$

in which p, v, $\&_i$ and $COND$ take *by default* the values 0.6, 1, *prod* (product) and

$$(valIn\_1 < pN(Individual) \leq valIn\_2) \tag{58}$$

RFuzzy High Level Semantics - Defaults

Translation of the definition of default truth values

$$fPredName(pT) :\sim defaults\_to(TV) \quad (59)$$

is translated into

$$fPredName(Individual) \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} (1,\ TV) \ \text{ if } \ COND \quad (60)$$

where the *by default* values for the variables p, v, $\&_i$ and *COND* are 0, 1, *prod* (product) and *true*.

RFuzzy High Level Semantics - Rules

Translation of rules

$$fPredName(pT) :\sim rule(fPredName2(pT)) \tag{61}$$

$$fPredName(pT) :\sim rule(aggr, (fPredName2(pT), fPredName3(pT), \ldots\ )) \tag{62}$$

are (respectively) translated into

$$fPredName(Individual) \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} fPredName2(Individual) \text{ if } COND \tag{63}$$

$$fPredName(Individual) \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} @(\ fPredName2(Individual), fPredName3(Individual),\ \ldots) \text{ if } COND \tag{64}$$

where the *by default* values for the variables p, v, $\&_i$ and *COND* are 0.4, 1, *prod* (product) and *true*.

RFuzzy High Level Semantics - Synonyms

Translation of the definition of a fuzzy characteristic as the synonym of another one

$$fPredName(pT) :\sim \quad synonym\_of(\ fPredName2(pT)\ ) \tag{65}$$

is translated into

$$fPredName(Individual) \xleftarrow{(p,\ v),\ \&_i} fPredName2(Individual) \ \text{if}\ \ COND \tag{66}$$

where the *by default* values for the variables p, v, $\&_i$ and *COND* are 0.8, 1, *prod* and *true*.

RFuzzy High Level Semantics - Synonyms

Translation of the definition of a fuzzy characteristic as the antonym of another one

$$\begin{aligned} & fPredName(pT) :\sim \\ & \qquad antonym\_of(\, fPredName2(pT)\,) \end{aligned} \tag{67}$$

is translated into

$$\begin{aligned} fPredName(Individual) \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} & \\ & not(fPredName2(Individual)) \\ & \text{if } \ COND \end{aligned} \tag{68}$$

where the *by default* values for the variables p, v, $\&_i$ and *COND* are 0.8, 1, *prod* and *true*.

$$similarity\_between(pT,\ pN(value1),\ pN(value2),\ TV). \quad (69)$$

is translated into

$$fuzzy\_comparator('=\sim=',\ pN,\ value1,\ Individual) \xleftarrow{\text{(p, v), } \&_i} TV \text{ if } pN(Individual,\ value2) \quad (70)$$

so that we can ask for the similarity between the characteristic entered in argument *value*1 and the value that the individual *Individual* has for the same characteristic.
The *by default* values for the variables p, v, $\&_i$ and *COND* are 0.8, 1, *prod* and *true*.

RFuzzy Low Level Semantics

RFuzzy Low Level Semantics - Priority inclusion

Definition ($\mathbb{KT}$)

$$\mathbb{KT} = \{\bot\} \cup \{ (p, v) \mid v \in [0, 1] \wedge p \in [0, 1] \} \quad (71)$$

Definition ($\preccurlyeq_{\mathbb{KT}}$)

The ordering between the values in $\mathbb{KT}$ is fixed by the definition of $\preccurlyeq_{\mathbb{KT}}$:

$$\bot \preccurlyeq_{\mathbb{KT}} \bot$$

$$\bot \preccurlyeq_{\mathbb{KT}} (p, v)$$

$$(p_1, v_1) \preccurlyeq_{\mathbb{KT}} (p_2, v_2) \leftrightarrow (\ p_1 < p_2\ )\ \textit{or}\ (\ p_1 = p_2\ \textit{and}\ v_1 \leq v_2\ )$$

where $<$ is defined as usually (remember that v_i and p_j are just real numbers between 0 and 1). It is obvious that the pair $(\mathbb{KT}, \preccurlyeq_{\mathbb{KT}})$ forms a complete lattice.

RFuzzy Low Level Semantics - Priority inclusion

For each kind of connective we define how to obtain the priority of the result from the priority of the inputs.

$$x \circ_{\&} y = \frac{x + y}{2} \tag{72}$$

$$z \circ_{\leftarrow} y = max(0, min(1, 2 * z - y)) \tag{73}$$

$$x \circ_{\vee} y = \frac{x + y}{2} \tag{74}$$

$$x \circ_{@} y = \frac{x + y}{2} \tag{75}$$

$$\circ_{\neg}(x) = x \tag{76}$$

$$\circ_{\Diamond}(x) = x \tag{77}$$

RFuzzy Low Level Semantics - Priority inclusion

- The functions are extended as usual when applied to more than two input values.
- The aim of taking into account the priority of every single root (each D_i, $i \in 1 \ldots n$) involved in the inference process removes the possibility to use mathematical operators in which the result remains unchanged when some input does not (i.e. min, max, etc).
- the operator must be defined for each possible connective.

RFuzzy Low Level Semantics - Modifiers inclusion

Definition (Basic formula)

A basic formula can be an atom (Eq. 78), the application of a modifier or a negation operator to an atom (Eqs. 79 and 80 resp.) or the application of a negation operator to the application of a modifier to an atom (Eq. 81).

A	(78)	*hot*
$\Diamond(A)$	(79)	*very*(*hot*)
$\neg(A)$	(80)	*not*(*hot*)
$\neg(\Diamond(A))$	(81)	*not*(*very*(*hot*))

RFuzzy Low Level Semantics - Low level syntax

Definition (Multi-Adjoint Logic Program)

A multi-adjoint logic program is a set of clauses of the form

$$\{\ A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} @_j(D_1, \ldots, D_i, \ldots, D_n) \quad \text{if} \ \ COND(A)\ \} \qquad (82)$$

where $(\mathrm{p},\ \mathrm{v}) \in \mathbb{KT}$, $\&_i$ is a conjunctor, $@_i$ an aggregator, A an atom, each D_i, $i \in [1..n]$, a basic formula (see Def. 3) and $COND(A)$ a first-order formula (a boolean condition that needs to be satisfied for A to get the value computed by the rule) formed by the predicates in $\mathrm{TB}_{\Pi,\Sigma,V}$, the predicates $=$, $\neq$, $\geq$, $\leq$, $>$ and $<$ restricted to terms from $\mathrm{TU}_{\Sigma,V}$, the symbol t and the conjunction $\wedge$ and disjunction $\vee$ in their usual meaning.

Definition (Multi-Adjoint Logic Program (cont.))

If $n = 1$ then $@_j$ is omitted (there is no need for an aggregator to combine the tuples of two or more basic formulas D_i because there is only one) and we represent it with the form

$$A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} D \tag{83}$$

If $n = 0$ the clause is intended to be used for assigning a truth value to an atom, with more or less credibility. In this case there is no aggregator nor basic formulas in the clause's body and we represent it as follows

$$A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} (\mathrm{p}',\ \mathrm{v}') \tag{84}$$

where $(\mathrm{p}',\ \mathrm{v}')$ is the truth value and priority assigned to the fact ($(\mathrm{p},\ \mathrm{v})$ is still the credibility assigned to the rule).

Suppose a program $\mathbb{P}$ formed by the set of rules

$$A \leftarrow \neg(B)$$
$$B \leftarrow (1,\ 0.1)$$

and assume that the definition of the negation operator is

$$Output \ = \ 1 \ - \ Input$$

The following interpretations are models of our program $\mathbb{P}$:

$$I_1 \ = \ \{\ (B,\ (1,\ \hat{}0.1)),\ (A,\ (1,\ \hat{}0.9))\ \}$$
$$I_2 \ = \ \{\ (B,\ (1,\ \hat{}0.4)),\ (A,\ (1,\ \hat{}0.6))\ \}$$

But their least model (the infimum between the interpretations) cannot be its declarative semantics since it is no more a model of $\mathbb{P}$:

$$I_3 \ = \ \{\ (B,\ (1,\ \hat{}0.1)),\ (A,\ (1,\ \hat{}0.6))\ \}$$

Extending the Expressiveness of Fuzzy Logic Languages

Motivation & Overview
RFuzzy Syntax
RFuzzy High Level Semantics
RFuzzy Low Level Semantics
Priority inclusion
Modifiers inclusion
Low level syntax
Practical applications
Conclusions
The end
Appendix
References

Definition (Stratification, adapted and extended from the definition proposed by Przymusinski in [Prz89])

A general program $\mathbb{P}$ is stratified if and only if it is possible to decompose the set S of all predicates of $\mathbb{P}$ into disjoint sets $S_1, \ldots, S_r$, called strata, so that for every clause $Cl_i \in \mathbb{P}$ of the form

$$\{ A \xleftarrow{(\mathrm{p},\ \mathrm{v}),\ \&_i} @_j(D_1, \ldots, D_j, \ldots, D_n) \text{ if } COND(A) \}, \quad (85)$$

$$\forall\ i\ stratum(A) \geq stratum(D_i) \text{ if } D_i \text{ has the form } B \text{ or } \Diamond B \quad (86)$$

$$\forall\ i\ stratum(A) > stratum(D_i) \text{ if } D_i \text{ has the form } \neg B \text{ or } \neg\Diamond B \quad (87)$$

Practical applications

Practical applications - Emotion recognition

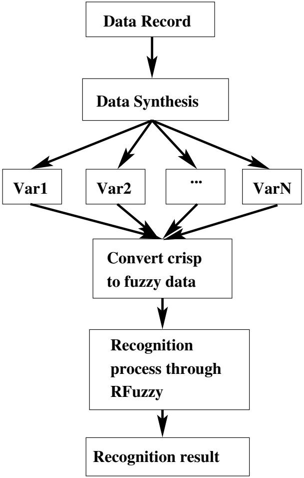


Figure: Methodology of data recognition, from [FM09].

Practical applications - RoboCup Soccer

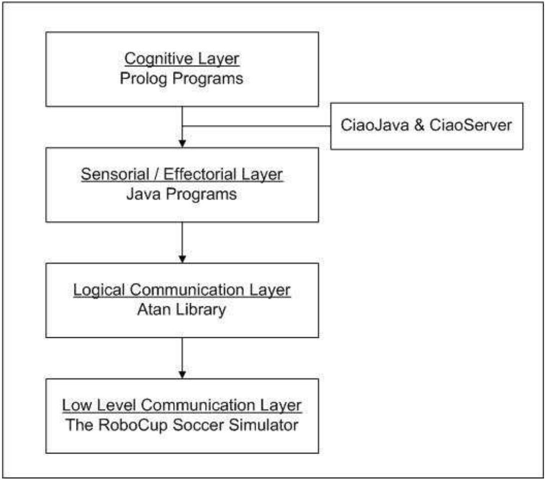


Figure: System Architecture for RoboCup Soccer Server.

Practical applications - Fuzzy Granularity Control in Parallel/Distributed Computing

In [TLGM10; TLGMH10] T. Trigo de la Vega, P. López-García and S. Muñoz-Hernández present an study in which they relax the (conservative) conditions taken into account for deciding not to parallelize a task (communication costs and others) by using fuzzy logic.

In the experimental assessment included in the study they show that the new fuzzy conditions implemented in RFuzzy select the optimal type of execution in most cases.

FleSe Demo

FleSe: Flexible Searches in Databases

Conclusions

Conclusions

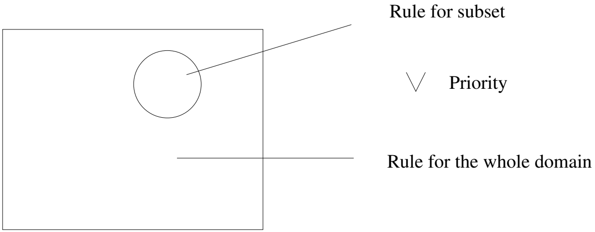
Improvements with respect to other fuzzy logic frameworks

- Overloading of fuzzy predicates ($cheap(car)$ and $cheap(house)$ do not behave in the same way).
- Priorities, allowing to have different rules for the same goal and obtaining the result from the rule with more priority instead of from the rule obtaining the highest truth value. Useful for:
  - Default value rules (We can give more importance to those rules using less default information)
  - Rules for subsets of individuals
  - Personalized rules
- Modifiers (very cheap) and negation (not cheap). And they can be used in programs and queries.
- Synonyms and Antonyms.
- Similarity between non-fuzzy attributes, (ask for spanish restaurants and obtain tapas restaurants).

Conclusions

Improvements with respect to other fuzzy logic frameworks

- We can group individuals to get rules where credibility is not as small as before, getting rules that represent better the world behaviour and results much more accurate than before.



Conclusions

Dissemination results

The work is the result of a very fertile research.
We have published our results in

- a workshop, ([PCMHS08]),
- eight conferences ([MHPCS09; PCMH11; PCMH14a; PCMH14b; PCMH14c; PCMH14d; PCSMH09; SMHPC09].
- a journal paper ([MHPCS11])
- and we have sent a second journal paper that we expect to be published soon ([PCMH15]).

Conclusions

Impact of RFuzzy and apps developed with it

- An American university ([Duk15]) was trying to determine if the emotion of a CEO CFO, manager, analyst, ... could affect the stock market
- An Spanish company ([Inn12]) was interested in using RFuzzy as we do it in FleSe to represent fuzzy concepts and translate fuzzy queries into queries that can be run against a non-fuzzy database.
- Some companies ([Emo12; Mar15]) showed interest for the emotion recognition application presented by Farooque and Muñoz-Hernández.

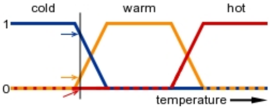
The end

Thank you for coming

Any question ?

Appendix

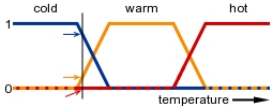
Appendix
Linguistic Variables



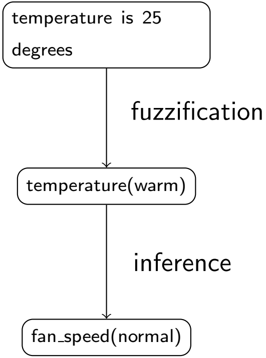
temperature is 25 degrees

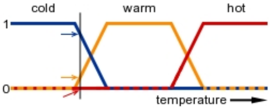
↓ fuzzification

temperature(warm)



Rules:
if temperature(warm) then fan_speed(normal)

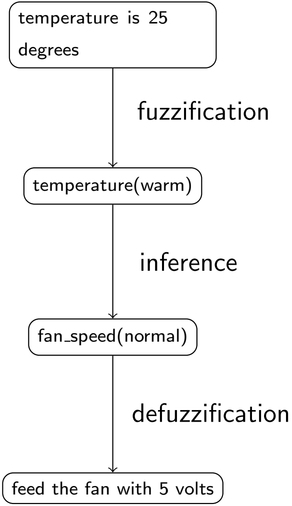




Rules:

if temperature(warm) then fan_speed(normal)

Fan_speed	Voltage
fast	10 volts
normal	5 volts
slow	2.5 volts
stop	0 volts



Appendix
Credibility necessity

Name	Swift	Tall	Good Player
Tim	0.9	0.9	0.2
John	0.8	0.7	0.7
Sarah	0.7	0.8	0.7

<u>Without credibility</u>

if swift and tall then good_player

aggregator: minimum

$$Tim: \quad min(0.9, 0.9) = 0.9 \neq 0.2 \quad \textit{WRONG !!}$$
$$John: \quad min(0.8, 0.7) = 0.7 = 0.7$$
$$Sarah: \quad min(0.7, 0.8) = 0.7 = 0.7$$

With Credibility

Name	Swift	Tall	Good Player
Tim	0.9	0.9	0.2
John	0.8	0.7	0.7
Sarah	0.7	0.8	0.7

if swift and tall then
good_player with credibility 0.2

aggregator: minimum
conjunctor: minimum (Gödel Logic)

$$Tim: \quad min(0.2,\ min(0.9, 0.9)) = 0.2 \leq 0.2$$
$$John: \quad min(0.2,\ min(0.8, 0.7)) = 0.2 \leq 0.7$$
$$Sarah: \quad min(0.2,\ min(0.7, 0.8)) = 0.2 \leq 0.7$$

Appendix
Multi-Adjoint Semantics

In our humble opinion, the multi-adjoint semantics [MOAV01a; MOAV01b; MOAV01c; MOAV02; MOAV04; MO02] are the best way to link together the real-world data and the credibility of our rules.

Departing from a Poset (a partially ordered set) $< P, \leq >$ and introducing a pair of operations ($\&, \leftarrow$), we say that the operations form an adjoint pair if

(1) $\&$ is increasing in both arguments,

(2) $\leftarrow$ is increasing in its first argument and decreasing in the second one and

(3) (*the adjoint property*) for any $x, y, z \in P$ we have that $z \leq (x \leftarrow y)$ holds if and only if $z \,\&\, y \leq x$.

The interesting point about the adjoint property is that for a fuzzy rule

if B then A with credibility z

where truth values for B and A are y and x respectively we can evaluate x from y and z or z from x and y:

$$z \leq (x \leftarrow y) \quad \text{iff} \quad z \,\&\, y \leq x$$

Example

Suppose we have four fuzzy facts, A, B, C and D. D is always satisfied with at least the maximum truth value of A, B, and C. From this knowledge we can extract the following rule:

$$D \xleftarrow{1,\ Gödel} max(A, B, C) \tag{88}$$

The interesting point is that the rule's credibility value, 1, has been computed from the real-world data. From Gödel's implication operator definition

$$b \xleftarrow{X,\ Gödel} a = \left\{ \begin{array}{lll} 1 & \text{if} & a \leq b \\ b & \text{if} & b < a \end{array} \right\} \tag{89}$$

and knowing that the satisfaction of D is always higher than the satisfaction of A, B and C we can obtain the rule's credibility value.

Example

Suppose we have four fuzzy facts, A, B, C and D and the rule in Eq. 88. Knowing that A, B and C are satisfied with, at least, the values 0.3, 0.4 and 0.5 we can compute how much satisfied is D:

$$\hat{D} = min(1,\ max(0.3, 0.4, 0.5)\ )\ =\ 0.5 \qquad (90)$$

Name	Swift	Tall	Good Player
Tim	0.9	0.9	0.2
John	0.8	0.7	0.7
Sarah	0.7	0.8	0.7

if swift and tall then
good_player with credibility z

aggregator: minimum
conjunctor: minimum (Gödel Logic)
implicator: $\overset{\wedge}{\leftarrow}(x, y) =$ x if $x < y$ else 1 (Gödel Logic)

$$\begin{aligned} Tim: &\quad min(0.9, 0.9) = 0.9; \quad \text{since} \;\; 0.9 > 0.2 \;\; \text{then} \;\; 0.2 \\ John: &\quad min(0.8, 0.7) = 0.7; \quad \text{since} \;\; 0.7 = 0.7 \;\; \text{then} \;\; 1 \\ Sarah: &\quad min(0.7, 0.8) = 0.7; \quad \text{since} \;\; 0.7 = 0.7 \;\; \text{then} \;\; 1 \end{aligned}$$

$$z = infimum(0.2,\ 1,\ 1) = 0.2$$

References

References

[Duk15] Duke University. *Duke University site*. [Online; accessed March 23, 2015]. 2015.

[Emo12] Emotion Explorer Lab. *Emotion Explorer Lab site*. [Online; accessed October 14, 2012]. 2012.

[FM09] Mahfuza Farooque and Susana Muñoz-Hernández. "Easy Fuzzy Tool for Emotion Recognition: Prototype from Voice Speech Analisys". In: *Proceeding of ICFC 2009 - First International Conference on Fuzzy Computation*. Ed. by J. Kacprzyk J. Filipe and A. Dourado. Madeira, Portugal: INSTICC Press, 2009.

[Inn12] Innovation4Information. *Innovation4Information site*. [Online; accessed October 14, 2012]. 2012.

References

[Mar15] Marketing de Servicios. *Marketing de Servicios (MDS) site*. [Online; accessed March 23, 2015]. 2015.

[MOAV01a] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtáš. "A Completeness Theorem for Multi-Adjoint Logic Programming". In: *FUZZ-IEEE*. 2001, pp. 1031–1034.

[MOAV01b] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtáš. "A Procedural Semantics for Multi-adjoint Logic Programming". In: *EPIA*. Ed. by Pavel Brazdil and Alípio Jorge. Vol. 2258. Lecture Notes in Computer Science. Springer, 2001, pp. 290–297. ISBN: 3-540-43030-X.

References

[MOAV01c] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtáš. "Multi-adjoint Logic Programming with Continuous Semantics". In: *LPNMR*. Ed. by Thomas Eiter, Wolfgang Faber, and Miroslaw Truszczynski. Vol. 2173. Lecture Notes in Computer Science. Springer, 2001, pp. 351–364. ISBN: 3-540-42593-4.

[MOAV02] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtáš. "A Multi-Adjoint Approach to Similarity-Based Unification". In: *Electronic Notes in Theoretical Computer Science* 66.5 (2002). UNCL'2002, Unification in Non-Classical Logics (ICALP 2002 Satellite Workshop), pp. 70 –85. ISSN: 1571-0661. DOI: `DOI:10.1016/S1571-0661(04)80515-2`.

References

[MOAV04] Jesús Medina, Manuel Ojeda-Aciego, and Peter Vojtáš. "Similarity-based unification: a multi-adjoint approach". In: *Fuzzy Sets and Systems* 146.1 (2004), pp. 43–62.

[MO02] Jesús Medina Moreno and Manuel Ojeda-Aciego. "On First-Order Multi-Adjoint Logic Programming". In: *11th Spanish Congress on Fuzzy Logic and Technology*. 2002.

References

[MHPCS09] Susana Muñoz-Hernández, Víctor Pablos-Ceruelo, and Hannes Strass. "RFuzzy: An Expressive Simple Fuzzy Compiler". In: *IWANN (1)*. Ed. by Joan Cabestany, Francisco Sandoval, Alberto Prieto, and Juan M. Corchado. Vol. 5517. Lecture Notes in Computer Science. Springer, 2009, pp. 270–277. ISBN: 978-3-642-02477-1. DOI: http://dx.doi.org/10.1007/978-3-642-02478-8_34.

References

[MHPCS11] Susana Muñoz-Hernández, Víctor Pablos-Ceruelo, and Hannes Strass. "RFuzzy: Syntax, Semantics and Implementation Details of a Simple and Expressive Fuzzy Tool over Prolog". In: *Information Sciences* 181.10 (2011). Special Issue on Information Engineering Applications Based on Lattices, pp. 1951 –1970. ISSN: 0020-0255. DOI: 10.1016/j.ins.2010.07.033.

References

[PCMH11] Víctor Pablos-Ceruelo and Susana Muñoz-Hernández. "Introducing priorities in RFuzzy: Syntax and Semantics". In: *CMMSE 2011 : Proceedings of the 11th International Conference on Mathematical Methods in Science and Engineering*. Vol. 3. Benidorm (Alicante), Spain, 2011, pp. 918–929. ISBN: 978-84-614-6167-7.

References

[PCMH14a] Víctor Pablos-Ceruelo and Susana Muñoz-Hernández. “Enriching Traditional Databases with Fuzzy Definitions to Allow Flexible and Expressive Searches”. In: *Proceedings of the International Conference on Fuzzy Computation Theory and Applications*. Ed. by António Dourado, José M. Cadenas, and Joaquim Filipe. Rome, Italy, 2014, pp. 111–118. ISBN: 978-989-758-053-6. DOI: http://dx.doi.org/10.5220/0005074101110118.

References

[PCMH14b] Víctor Pablos-Ceruelo and Susana Muñoz-Hernández. "FleSe: A Tool for Posing Flexible and Expressive (Fuzzy) Queries to a Regular Database". In: *Distributed Computing and Artificial Intelligence, 11th International Conference*. Ed. by Sigeru Omatu, Hugues Bersini, Juan M. Corchado Rodríguez, Sara Rodríguez, Pawel Pawlewski, and Edgardo Bucciarelli. Vol. 290. Advances in Intelligent Systems and Computing. DCAI 2014. Salamanca, Spain: Springer, 2014, pp. 157–164. ISBN: 978-3-319-07592-1.

References

[PCMH14c] Víctor Pablos-Ceruelo and Susana Muñoz-Hernández. "Introducing Similarity Relations in a Framework for Modelling Real-World Fuzzy Knowledge". In: *IPMU (3)*. Ed. by Anne Laurent, Olivier Strauss, Bernadette Bouchon-Meunier, and Ronald R. Yager. Vol. 444. Communications in Computer and Information Science. online isbn: 978-3-319-08852-5, series issn: 1865-0929. Springer International Publishing, 2014, pp. 51–60. ISBN: 978-3-319-08851-8. DOI: 10.1007/978-3-319-08852-5_6.

References

[PCMH14d] Víctor Pablos-Ceruelo and Susana Muñoz-Hernández. "On modelling real-world knowledge to get answers to fuzzy and flexible searches without human intervention". In: *Fuzzy Systems (FUZZ-IEEE), 2014 IEEE International Conference on*. Beijing, China, 2014, pp. 2329–2336. ISBN: 978-1-4799-2073-0. DOI: 10.1109/FUZZ-IEEE.2014.6891723.

[PCMH15] Víctor Pablos-Ceruelo and Susana Muñoz-Hernández. "FleSe". In: *unknown* 0.0 (2015). to be published, pp. 0 –0.

References

[PCMHS08] Víctor Pablos-Ceruelo, Susana Muñoz-Hernández, and Hannes Strass. "RFuzzy framework". In: *18th Workshop on Logic-based methods in Programming Environments, WLPE 2008*. Ed. by Puri Arenas and Damiano Zanardini. Vol. abs/0903.2188. Udine, Italy, 2008, pp. 62–76.

References

[PCSMH09] Víctor Pablos-Ceruelo, Hannes Strass, and Susana Muñoz-Hernández. “RFuzzy—A framework for multi-adjoint Fuzzy Logic Programming”. In: *Fuzzy Information Processing Society, 2009. NAFIPS 2009. Annual Meeting of the North American Fuzzy Information Processing Society Annual Conference*. Cincinnati, Ohio, USA, 2009, pp. 1–6. ISBN: 978-1-4244-4575-2. DOI: 10.1109/NAFIPS.2009.5156427.

[Prz89] Teodor C. Przymusinski. “On the Declarative and Procedural Semantics of Logic Programs”. In: *Journal of Automated Reasoning* 5.2 (1989), pp. 167–205.

References

[SMHPC09] Hannes Strass, Susana Muñoz-Hernández, and Víctor Pablos-Ceruelo. "Operational Semantics for a Fuzzy Logic Programming System with Defaults and Constructive Answers". In: *IFSA/EUSFLAT Conf.* Ed. by João Paulo Carvalho, Didier Dubois, Uzay Kaymak, and João Miguel da Costa Sousa. 2009, pp. 1827–1832. ISBN: 978-989-95079-6-8.

References

[TLGM10] Teresa Trigo de la Vega, Pedro López-García, and Susana Muñoz-Hernández. "A Fuzzy Approach to Resource Aware Automatic Parallelization". In: *Computational Intelligence - Revised and Selected Papers of the International Joint Conference, IJCCI 2010, Valencia, Spain, October 2010*. Ed. by Kurosh Madani, António Dourado Correia, Agostinho C. Rosa, and Joaquim Filipe. Vol. 399. Studies in Computational Intelligence. Springer, 2010, pp. 229–245. ISBN: 978-3-642-27533-3. DOI: 10.1007/978-3-642-27534-0_15.

References

[TLGMH10] Teresa Trigo de la Vega, Pedro López-García, and Susana Muñoz-Hernández. "Towards Fuzzy Granularity Control in Parallel/Distributed Computing". In: *IJCCI (ICFC-ICNC)*. Ed. by Joaquim Filipe and Janusz Kacprzyk. Best Student Paper Award ICFC 2010. SciTePress, 2010, pp. 43–55. ISBN: 978-989-8425-32-4.