Extending Prolog with Incomplete Fuzzy Information

Susana Muñoz-Hernández

Claudio Vaucheret

Facultad de Informática

Universidad Politécnica de Madrid

28660 Madrid, Spain

susana@fi.upm.es

vaucheret@ibap.com.ar

Overview

- Introduction: Fuzzy Prolog
- Problem: Incomplete Information
- Solution: Default Values
- Conclusion

Overview

- Introduction: Fuzzy Prolog
- Problem: Incomplete Information
- Solution: Default Values
- Conclusion

Introduction

- Existing Fuzzy Prolog systems:
  - Prolog-Elf
  - Fril Prolog
  - f-Prolog
- Our Fuzzy Prolog approach:

  (S. Guadarrama, C. Vaucheret, S. Muñoz-Hernández)
  - CLP($\mathcal{R}$) based implementation
  - Truth Value (union of sub-intervals) $\mathcal{B}([0,1])$
  - Aggregation operators (min, max, luka, ...)

Aggregation Operators

- A function $f : [0,1]^n \rightarrow [0,1]$ that verifies $f(0,\ldots,0) = 0$, $f(1,\ldots,1) = 1$, and in addition it is monotonic and continuous, then it is called `aggregation operator`

Aggregation Operators

- A function $f : [0,1]^n \rightarrow [0,1]$ that verifies $f(0,\ldots,0) = 0$, $f(1,\ldots,1) = 1$, and in addition it is monotonic and continuous, then it is called `aggregation operator`
- Given an aggregation $f : [0,1]^n \rightarrow [0,1]$ an `interval-aggregation` $F : \mathcal{E}([0,1])^n \rightarrow \mathcal{E}([0,1])$ is defined as follows:

$$F([x_1^l, x_1^u], ..., [x_n^l, x_n^u]) = [f(x_1^l, ..., x_n^l), f(x_1^u, ..., x_n^u)]$$

Union Aggregation

- Given an interval-aggregation $F : \mathcal{E}([0,1])^n \longrightarrow \mathcal{E}([0,1])$ defined over intervals, a `union-aggregation` $\mathcal{F} : \mathcal{B}([0,1])^n \rightarrow \mathcal{B}([0,1])$ is defined over union of intervals as follows:

$$\mathcal{F}(B_1, \ldots, B_n) = \cup\{F(\mathcal{E}_1, ..., \mathcal{E}_n) \mid \mathcal{E}_i \in B_i\}$$

Syntax

Let

$A, B_1, \ldots, B_n$ be atoms,

v a truth value in $\mathcal{B}([0,1])$ (constraints over $[0,1]$),

F is an *aggregation operator*.

- *fuzzy fact*: $A \leftarrow v$
- *fuzzy clause*: $A\ v \leftarrow_F B_1\ v_1, \ldots, B_n\ v_n$
- *fuzzy query*: $v \leftarrow A\ ?$

Implementation: Syntax

```
tall(john):~ [0.8, 0.9].

good_player(X):~ min
                      tall(X),
                      swift(X).
```

Implementation: CLP($\mathcal{R}$)

```
tall(john):~
    [0.8, 0.9].
```

```
good_player(X):~min
    tall(X),
    swift(X).
```

```
tall(john,V) :-
    V .>=. 0.8,
    V .=<. 0.9.
```

```
good_player(X,V) :-
    tall(X,Vt),
    swift(X,Vs),
    minim([Vt,Vs],V),
    V.>=.0, V.=<.1.
```

Initial Evaluation

- Implementation over CLP($\mathcal{R}$): SIMPLICITY
- Aggregation operator: GENERALITY
- Definition of new operators: FLEXIBILITY
- Using Prolog resolution: EFFICIENCY

Overview

- Introduction: Fuzzy Prolog
- Problem: Incomplete Information
- Solution: Default Values
- Conclusion

Incomplete Information

But.....

What happend when not all the information is available?....

Incomplete Information

But.....

What happend when not all the information is available?....

Prolog $\Rightarrow$ Failure of the computation !!!

Combining Crisp and Fuzzy Logic

```
student(john).
student(peter).

------------------------------------

age_about_15(john):~ 1.
age_about_15(susan):~ 0.7.
age_about_15(nick):~ 0.

------------------------------------

teenager_student(X):~
        student(X),     % CRISP
        age_about_15(X).% FUZZY
```

```
?- student(john).
yes
?- student(nick).
no            FALSE
?- age_about_15(john,V).
V = 1
?- age_about_15(nick,V).
V = 0
?- age_about_15(peter,V).
no            UNKNOWN
?- teenager_student(john,V).
V .=. 1
?- teenager_student(susan,V).
V .=. 0
?- teenager_student(peter,V).
no            UNKNOWN
```

Overview

- Introduction: Fuzzy Prolog
- Problem: Incomplete Information
- Solution: Default Values
- Conclusion

Solution: Default Knowledge

```
student(john).
student(peter).
:-default(f_student/2,0).
f_student(X,1):-
         student(X).
------------------------------------
:-default(age_about_15/2,[0,1]).
age_about_15(john):~ 1.
age_about_15(susan):~ 0.7.
age_about_15(nick):~ 0.
------------------------------------
:-default(teenager_student/2,[0,1]).
teenager_student(X):~
        f_student(X),
        age_about_15(X).
```

```
?- f_student(john,V).
V = 1
?- f_student(nick,V).
V = 0           FALSE
?- age_about_15(john,V).
V = 1
?- age_about_15(nick,V).
V = 0
?- age_about_15(peter,V).
V .>=. 0, V .<=. 1    UNKNOWN
?- teenager_student(john,V).
V .=. 1
?- teenager_student(susan,V).
V .=. 0
?- teenager_student(peter,V).
V .>=. 0, V .<=. 1    UNKNOWN
```

Default Value

We assume there is a function *default* which implement the Default Knowledge Assumptions. It assigns an element of $\mathcal{B}([0,1])$ to each element of the Herbrand Base.

- If the Closed World Assumption is used, then $default(A) = [0,0]$ for all A in Herbrand Base.
- If Open World Assumption is used instead, $default(A) = [0,1]$ for all A in Herbrand Base.

Interpretation

An *interpretation* I consists of the following:

1. a subset B_I of the *Herbrand Base*,
2. a mapping V_I, to assign
   (a) a truth value, in $\mathcal{B}([0,1])$, to each element of B_I, or
   (b) $default(A)$, if A does not belong to B_I.

Former Operational Semantics

A *transition* in the *transition system* is defined as:

1. $\langle A \cup a, \sigma, S \rangle \rightarrow \langle A\theta, \sigma \cdot \theta, S \wedge \mu_a = v \rangle$
   if $h \leftarrow v$ is a fact of the program P, θ is the mgu of a and h, and μ_a is the truth variable for a, and $solvable(S \wedge \mu_a = v)$.

2. $\langle A \cup a, \sigma, S \rangle \rightarrow \langle (A \cup B)\theta, \sigma \cdot \theta, S \wedge c \rangle$
   if $h \leftarrow_F B$ is a rule of the program P, θ is the mgu of a and h, c is the constraint that represents the truth value obtained applying the union-aggregator F on the truth variables of B, and $solvable(S \wedge c)$.

3. $\langle A \cup a, \sigma, S \rangle \rightarrow fail$ if none of the above are applicable.

New Operational Semantics

A *transition* in the *transition system* is defined as:

1. $\langle A \cup a, \sigma, S \rangle \rightarrow \langle A\theta, \sigma \cdot \theta, S \wedge \mu_a = v \rangle$
   if $h \leftarrow v$ is a fact of the program P, ...
2. $\langle A \cup a, \sigma, S \rangle \rightarrow \langle (A \cup B)\theta, \sigma \cdot \theta, S \wedge c \rangle$
   if $h \leftarrow_F B$ is a rule of the program P, ...
3. $\langle A \cup a, \sigma, S \rangle \rightarrow \langle A, \sigma, S \wedge \mu_a = v \rangle$
   if none of the above are applicable and
   $solvable(S \wedge \mu_a = v)$ where $\mu_a = default(a)$.

Semantics Equivalence

Given a program P, the three semantics:

1. Least model $lm(P)$, under the $\sqsubseteq$ ordering.
2. Declarative meaning $lfp(T_P)$, least fi xpoint for a consequence operator $T_P(I)$.
3. Success set $SS(P)$ of a transitional system.

are equivalent: $SS(P) = lfp(T_P) = lm(P)$.

Example

```
        q(a) :~ [0.2,0.3].
        w(a) :~ [0.1,0.5].
        v(a) :~ 0.9.

p(X):~ min
     q(X),
     r(X),
     w(X),
     v(X).
```

Example - Incomplete

```
q(a) :~ [0.2,0.3].
w(a) :~ [0.1,0.5].
v(a) :~ 0.9.

p(X):~ min
     q(X),
     r(X),
     w(X),
     v(X).
```

```
:- p(a).
```

Failure in Prolog

Example - Default

```
q(a) :~ [0.2,0.3].
w(a) :~ [0.1,0.5].
v(a) :~ 0.9.
```

```
p(X):~ min
    q(X),
    r(X),
    w(X),
    v(X).
```

```
r(a) :~ [0,1].
```

Unknown OWA
(default value)

Overview

- Introduction: Fuzzy Prolog
- Problem: Incomplete Information
- Solution: Default Values
- Conclusion

Conclusion

- Representation of real problems
- Crisp + Fuzzy logic: EXPRESIVITY
- $[0,1]$ to represent total uncertainty ($0 \leq v \wedge v \leq 1$). Lack of information do not stop the evaluation: ACCURACY
- Provides answers: CONSTRUCTIVE

Conclusion

- Representation of real problems
- Crisp + Fuzzy logic: EXPRESIVITY
- $[0, 1]$ to represent total uncertainty ($0 \leq v \wedge v \leq 1$). Lack of information do not stop the evaluation: ACCURACY
- Provides answers: CONSTRUCTIVE

Available implementation:

http://clip.dia.fi.upm.es/Software/Ciao/

Extending Prolog with Incomplete Fuzzy Information

Susana Muñoz-Hernández

Claudio Vaucheret

Facultad de Informática

Universidad Politécnica de Madrid

28660 Madrid, Spain

susana@fi.upm.es

vaucheret@ibap.com.ar