Implementing a Fuzzy Prolog

Moritz Knapp

Seminar für Computerlinguistik

< 🗗 🕨

A B M A B M

ELE DQA

Outline

Motivation

- Prolog a Popular Al Language
- The Problem of Fuzzy Data
- 2 A Solution Basing Prolog on Fuzzy Logic
 - By Defining New Predicates
 - By Building Fuzziness into the Interpreter

3 The Ciao Prolog System

- A Prolog Programming Environment
- Ciao's fuzzy package
- Example Program

Prolog - a Popular Al Language The Problem of Fuzzy Data

Outline

Motivation

- Prolog a Popular AI Language
- The Problem of Fuzzy Data
- A Solution Basing Prolog on Fuzzy Logic
 - By Defining New Predicates
 - By Building Fuzziness into the Interpreter
- 3 The Ciao Prolog System
 - A Prolog Programming Environment
 - Ciao's fuzzy package
 - Example Program

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Standard Prolog (based on two-valued logic) is widely used in Artificial Intelligence research.

- natural language processing (i.e. parsing)
- database systems
- expert ('knowledge-based') systems
 - medicine
 - finances
 - law
 - . . .

◆□▶ ◆帰▶ ◆∃▶ ◆∃▶ ∃|= のQ@

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Standard Prolog (based on two-valued logic) is widely used in Artificial Intelligence research.

- natural language processing (i.e. parsing)
- database systems
- expert ('knowledge-based') systems
 - medicine
 - finances
 - law
 - . . .

◆□▶ ◆帰▶ ◆∃▶ ◆∃▶ ∃|= のQ@

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Standard Prolog (based on two-valued logic) is widely used in Artificial Intelligence research.

- natural language processing (i.e. parsing)
- database systems
- expert ('knowledge-based') systems
 - medicine
 - finances
 - law
 - . . .

◆□▶ ◆帰▶ ◆∃▶ ◆∃▶ ∃|= のQ@

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Standard Prolog (based on two-valued logic) is widely used in Artificial Intelligence research.

- natural language processing (i.e. parsing)
- database systems
- expert ('knowledge-based') systems
 - medicine
 - finances
 - law
 - . . .

Motivation A Solution - Basing Prolog on Fuzzy Logic

The Ciao Prolog System

Prolog - a Popular AI Language The Problem of Fuzzy Data

Outline

1 Motivation

- Prolog a Popular Al Language
- The Problem of Fuzzy Data
- 2 A Solution Basing Prolog on Fuzzy Logic
 - By Defining New Predicates
 - By Building Fuzziness into the Interpreter
- 3 The Ciao Prolog System
 - A Prolog Programming Environment
 - Ciao's fuzzy package
 - Example Program

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Problem: Sometimes, the data given is not 'crisp':

• John is 35. Is John young?

- We want something other than yes or no (like fairly young)
- 'John is young' has a degree of truth!

How can we handle this?

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Problem: Sometimes, the data given is not 'crisp':

- John is 35. Is John young?
- We want something other than yes or no (like fairly young)
- 'John is young' has a degree of truth!

How can we handle this?

< □ > < 向 >

Prolog - a Popular Al Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Problem: Sometimes, the data given is not 'crisp':

- John is 35. Is John young?
- We want something other than yes or no (like fairly young)
- 'John is young' has a degree of truth!

How can we handle this?

A E > A E > E

Prolog - a Popular AI Language The Problem of Fuzzy Data

Prolog in Artificial Intelligence

Problem: Sometimes, the data given is not 'crisp':

- John is 35. Is John young?
- We want something other than yes or no (like fairly young)
- 'John is young' has a degree of truth!

How can we handle this?

By Defining New Predicates By Building Fuzziness into the Interprete

Outline

Motivation

• Prolog - a Popular Al Language

• The Problem of Fuzzy Data

2 A Solution - Basing Prolog on Fuzzy Logic

- By Defining New Predicates
- By Building Fuzziness into the Interpreter

3 The Ciao Prolog System

- A Prolog Programming Environment
- Ciao's fuzzy package
- Example Program

By Defining New Predicates By Building Fuzziness into the Interpreter

Degrees of Truth

A statement is **true to some degree**. The degree of truth marks the degree of membership in the set of true statements.

- fuzzy facts are associated with truth values in the real interval [0, 1]
- fuzzy rules have truth values determined by the combination of its goals' truth values
- common combinations of truth values:
 - minimum (conjunction)
 - maximum (disjunction)
 - complement (negation)

글 🖌 🖌 글 🕨

By Defining New Predicates By Building Fuzziness into the Interpreter

Degrees of Truth

A statement is **true to some degree**. The degree of truth marks the degree of membership in the set of true statements.

- fuzzy facts are associated with truth values in the real interval [0,1]
- fuzzy rules have truth values determined by the combination of its goals' truth values
- common combinations of truth values:
 - minimum (conjunction)
 - maximum (disjunction)
 - complement (negation)

Degrees of Truth

A statement is **true to some degree**. The degree of truth marks the degree of membership in the set of true statements.

- fuzzy facts are associated with truth values in the real interval [0, 1]
- fuzzy rules have truth values determined by the combination of its goals' truth values
- common combinations of truth values:
 - minimum (conjunction)
 - maximum (disjunction)
 - complement (negation)

Degrees of Truth

A statement is **true to some degree**. The degree of truth marks the degree of membership in the set of true statements.

- fuzzy facts are associated with truth values in the real interval [0, 1]
- fuzzy rules have truth values determined by the combination of its goals' truth values
- common combinations of truth values:
 - minimum (conjunction)
 - maximum (disjunction)
 - complement (negation)

ヨト イヨト ヨヨ のへの

By Defining New Predicates By Building Fuzziness into the Interprete

Outline

Motivation

- Prolog a Popular Al Language
- The Problem of Fuzzy Data
- A Solution Basing Prolog on Fuzzy Logic
 By Defining New Predicates
 - By Building Fuzziness into the Interpreter

3 The Ciao Prolog System

- A Prolog Programming Environment
- Ciao's fuzzy package
- Example Program

By Defining New Predicates By Building Fuzziness into the Interpreter

Example

Our data:

- average temperatures of locations
- average number of hours of sunshine at location

Query:

• How "hot and sunny" is location X?

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

By Defining New Predicates By Building Fuzziness into the Interpreter

Example

Our data:

- average temperatures of locations
- average number of hours of sunshine at location

Query:

```
• How "hot and sunny" is location X?
```

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

By Defining New Predicates By Building Fuzziness into the Interpreter

Example

Our data:

- average temperatures of locations
- average number of hours of sunshine at location Query:
 - How "hot and sunny" is location X?

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

Example

Our data:

- average temperatures of locations
- average number of hours of sunshine at location Query:
 - How "hot and sunny" is location X?

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

B b d B b

Additional Arguments for Truth Values

Our data:

- average temperatures of locations
- average number of hours of sunshine at location

Query:

• How "hot and sunny" is location X?

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

Additional Arguments for Truth Values

Our data:

- average temperatures of locations
- average number of hours of sunshine at location

Query:

• How "hot and sunny" is location X?

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

Additional Arguments for Truth Values

Our data:

- average temperatures of locations
- average number of hours of sunshine at location

Query:

• How "hot and sunny" is location X?

```
hot_and_sunny_area(X,CHI):-
    av_temp(X,T), hot(T,CHI1),
    av_sun(X,S), sunny(S,CHI2),
    combine(CHI1,CHI2,CHI).
```

By Defining New Predicates By Building Fuzziness into the Interpreter

Outline

Motivation

- Prolog a Popular AI Language
- The Problem of Fuzzy Data

A Solution - Basing Prolog on Fuzzy Logic By Defining New Predicates

• By Building Fuzziness into the Interpreter

3 The Ciao Prolog System

- A Prolog Programming Environment
- Ciao's fuzzy package
- Example Program

By Defining New Predicates By Building Fuzziness into the Interpreter

Handling Fuzziness Automatically

- The Prolog interpreter is reimplemented to handle fuzziness
- Reduced notation:

not_and_sunny_area(X): av_temp(X, T), hot(T),
 av_sun(X, S), sunny(S).

- more natural, cleaner
- more readable
- behaves like standard Prolog, if there is no fuzziness

(日)

By Defining New Predicates By Building Fuzziness into the Interpreter

Handling Fuzziness Automatically

- The Prolog interpreter is reimplemented to handle fuzziness
- Reduced notation:

```
hot_and_sunny_area(X):-
    av_temp(X, T), hot(T),
    av_sun(X, S), sunny(S).
```

- more natural, cleaner
- more readable
- behaves like standard Prolog, if there is no fuzziness

Image: Image:

By Defining New Predicates By Building Fuzziness into the Interpreter

Handling Fuzziness Automatically

- The Prolog interpreter is reimplemented to handle fuzziness
- Reduced notation:

```
hot_and_sunny_area(X):-
    av_temp(X, T), hot(T),
    av_sun(X, S), sunny(S).
```

- more natural, cleaner
- more readable
- behaves like standard Prolog, if there is no fuzziness

★ ■ ▶ ★ ■ ▶ ■

A Prolog Programming Environment Ciao's fuzzy package Example Program

Outline

Motivation

- Prolog a Popular Al Language
- The Problem of Fuzzy Data
- A Solution Basing Prolog on Fuzzy Logic
 By Defining New Predicates
 - By Building Fuzziness into the Interpreter
- 3 The Ciao Prolog System
 - A Prolog Programming Environment
 - Ciao's fuzzy package
 - Example Program

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

The **Coo** Prolog Development System

- extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance . . .
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - . . .
- includes an emacs interface
- License: GNU Library General Public License (LGPL).
- Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

The **Coo** Prolog Development System

• extends Prolog with

- object-oriented programming
 - modules, classes, inheritance ...
- constraint programming
 - X .=. Y+Z, X .=<. 2*Y
- alternative computation rules
 - breadth-first, iterative deepening
- external interfaces (C, Java, ...)
- . . .
- includes an emacs interface
- License: GNU Library General Public License (LGPL).
- Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance . . .
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - . . .
 - includes an emacs interface
 - License: GNU Library General Public License (LGPL).
 - Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance . . .
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - . . .
 - includes an emacs interface
 - License: GNU Library General Public License (LGPL).
 - Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance ...
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening ...

```
• external interfaces (C, Java, ...)
```

- . . .
- includes an emacs interface
- License: GNU Library General Public License (LGPL)
- Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance ...
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - ...
 - includes an emacs interface
 - License: GNU Library General Public License (LGPL).
 - Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance . . .
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - ...
 - includes an emacs interface
 - License: GNU Library General Public License (LGPL).
 - Homepage:

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

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance . . .
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - ...
 - includes an emacs interface
 - License: GNU Library General Public License (LGPL).
 - Homepage:

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

▲冊 ▶ ▲ 臣 ▶ ▲ 臣 ▶ 三 臣 ■ り へ ()

A Prolog Programming Environment Ciao's fuzzy package Example Program

A Prolog Programming Environment

- The **Coo** Prolog Development System
 - extends Prolog with
 - object-oriented programming
 - modules, classes, inheritance . . .
 - constraint programming
 - X .=. Y+Z, X .=<. 2*Y
 - alternative computation rules
 - breadth-first, iterative deepening
 - external interfaces (C, Java, ...)
 - ...
 - includes an emacs interface
 - License: GNU Library General Public License (LGPL).
 - Homepage:

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

▲冊 ▶ ▲ 臣 ▶ ▲ 臣 ▶ 三 臣 ■ り へ ()

A Prolog Programming Environment Ciao's fuzzy package Example Program

The emacs Interface

M (and () al	
File Edit Contenes Buffers Table ClasSins ClasSins ClasBB Bries ClasSink Classibile Hole	
File Eak Options Bullers Tools Claubby Claubby Claubry Erudo Claubpis Clauberp Help	
🙆 🖗 × 🚯 🕉 🛩 🕪 🕉 🖗 🗳 🐜 😔 🖉 🖉 🦉	S C O O C C C C
🖸 🕼 🖶 🕈 🌾	
<pre></pre>	
% A function	
fact(0) := 1.	
fact(N) := N * ~fact(N) := N > 0.	
X A predicate	
append([],X,X).	
append(EXIY],Z,EXIW]) :-	
append(Y,Z,W).	
% Using constraints (CLP(Q))	
lib(X,Y) :- X .=. 0, Y .=. 0.	
fib(X,Y) := X = 1, Y = 1.	
T1D(N,r) := N - 1	
N2 .=. N - 2.	
fib(N1, F1).	
fib(N2, F2), F = F1+F2	
-	
/ (Ciac (Pealer) 12- Terrererererererererererererererererere	
<u>1 1201_11.p1 (01307F010g7-113-10p</u>	
E Cae	
{Including /home/herme/local/lib/ciaopp/ciaopp_1 0/path_init pl	
}	
)	
Clao 1.11 #308: Mon Mar 14 15:23:07 CET 2005	
:** *('ign/Prolog* (fice/Prolog/Prolog/Indec listener: run)==	9-011
Quit	
	- 《 다 》 《 다 》 《 는 》 《 는 》 '문 드 ' ~)

Moritz Knapp Implementing a Fuzzy Prolog

A Prolog Programming Environment Ciao's fuzzy package Example Program

Outline

Motivation

- Prolog a Popular Al Language
- The Problem of Fuzzy Data
- A Solution Basing Prolog on Fuzzy Logic
 By Defining New Predicates
 - By Building Fuzziness into the Interpreter

3 The Ciao Prolog System

- A Prolog Programming Environment
- Ciao's fuzzy package
- Example Program

A Prolog Programming Environment Ciao's fuzzy package Example Program

Using the fuzzy Package

Packages can be imported:

• :- use_package(fuzzy).

3

Motivation A Prolog Programming Er A Solution - Basing Prolog on Fuzzy Logic The Ciao Prolog System Example Program

Interface Extensions: New Declaration

aggr/1

Aggregators implement the combination of truth values. Predefined aggregators:

- min
- prod a × b
- luka max{0, a + b 1}
- max
- dprod a + b a \times b
- dluka min $\{a + b, 1\}$

< □ > < 同 >

A B M A B M

A Prolog Programming Environment Ciao's fuzzy package Example Program

Interface Extensions: New Predicates

:#/2 and fuzzy_predicate/1

• for defining fuzzy predicates by listing a piecewise linear continuous membership function

young :\# fuzzy_predicate([(0,1),(35,1),(45,0),(120,0)]) :~/2

• for fuzzy clauses:

▲ Ξ ▶ ▲ Ξ ▶ Ξ Ξ = 𝒴 𝔄 𝔄 𝔅

A Prolog Programming Environment Ciao's fuzzy package Example Program

Interface Extensions: New Operators

- fnot/1 [1150,fx] fuzzy negation of a fuzzy predicate
- :=/2 [1200,xfx]
- :=/1 [1200,×f]
- :#/2 [1200,xfx]
- =>/1 [1175,fx]
- aggr/1 [1150,fx]
- . . .

▲ ■ ▶ ▲ ■ ▶ ■ ■ ■ ● ● ● ● ●

- ∢ 🗇 🕨

A Solution - Basing Prolog on Fuzzy Logic The Ciao Prolog System The Ciao Prolog System

Outline

Motivation

- Prolog a Popular Al Language
- The Problem of Fuzzy Data
- A Solution Basing Prolog on Fuzzy Logic
 By Defining New Predicates
 - By Building Fuzziness into the Interpreter

3 The Ciao Prolog System

- A Prolog Programming Environment
- Ciao's fuzzy package
- Example Program

```
Motivation A Prolog Programming Environment
A Solution - Basing Prolog on Fuzzy Logic
The Ciao Prolog System Example Program
```

```
small :#
    fuzzy_predicate([(1,1),(2,1),(3,0.7),(4,0.3),(5,0),(6,0)]).
large :#
    fuzzy_predicate([(1,0),(2,0),(3,0.3),(4,0.7),(5,1),(6,1)]).
die1(X.M) :~
    small(X,M).
die2(X,M) :~
    large(X,M).
two_dice(X,Y,M):~ prod
    die1(X.M1).
    die2(Y,M2).
sum(2,M) :~
    two dice(1.1.M1).
sum(5,M) :\sim dprod
    two_dice(4,1,M1),
    two dice(1.4.M2).
    two_dice(3,2,M3),
    two_dice(2,3,M4).
```

◆□▶ ◆□▶ ◆□▶ ◆□▶ 三日 のへ⊙

```
        Motivation
        A Prolog Programming Environment

        A Solution - Basing Prolog on Fuzzy Logic
        Ciao's fuzzy package

        The Ciao Prolog System
        Example Program
```

```
small :#
    fuzzy_predicate([(1,1),(2,1),(3,0.7),(4,0.3),(5,0),(6,0)]).
large :#
    fuzzy_predicate([(1,0),(2,0),(3,0.3),(4,0.7),(5,1),(6,1)]).
die1(X.M) :~
    small(X,M).
die2(X,M) :~
    large(X,M).
two_dice(X,Y,M):~ prod
    die1(X.M1).
    die2(Y,M2).
sum(2,M) :~
    two_dice(1,1,M1).
sum(5,M) :\sim dprod
    two_dice(4,1,M1),
    two dice(1.4.M2).
    two_dice(3,2,M3),
    two_dice(2,3,M4).
```

◆□▶ ◆□▶ ◆□▶ ◆□▶ 三日 のへ⊙

Motivation A Prolog Programming Environment A Solution - Basing Prolog on Fuzzy Logic The Ciao Prolog System Example Program

Demo

Demo

Moritz Knapp Implementing a Fuzzy Prolog



- fuzzy data is a challenge that can be handled in different ways
 coo
 is one possible solution
- nice IDE for experimenting with fuzziness
- also worth a look: reimplementations of the Prolog interpreter (see literature)

医下子 医



- fuzzy data is a challenge that can be handled in different ways
 Coo
 is one possible solution
- nice IDE for experimenting with fuzziness
- also worth a look: reimplementations of the Prolog interpreter (see literature)

医下子 医



- fuzzy data is a challenge that can be handled in different ways
 Coo
 is one possible solution
- nice IDE for experimenting with fuzziness
- also worth a look: reimplementations of the Prolog interpreter (see literature)



- fuzzy data is a challenge that can be handled in different ways
 Coo
 is one possible solution
- nice IDE for experimenting with fuzziness
- also worth a look: reimplementations of the Prolog interpreter (see literature)

Thanks for Listening

∃ → < ∃</p>

三日 のへで

Literature I



S. Gottwald. Fuzzy Sets and Fuzzy Logic. Vieweg, 1993.

I.A. Zadeh.

Fuzzy Sets.

Information and Control 8, 338–353, 1965.

T.P. Martin, J.F. Baldwin, B.W. Pilsworth. The Implementation of FProlog - A Fuzzy Prolog Interpreter. Fuzzy Sets and Systems 23, 119–129, 1985.