GALWAY-MAYO INSTITUTE OF TECHNOLOGY

Department of Computer Science & Applied Physics

Fuzzy Logic Project Staffing Example

1. Overview

JFuzzyLogic is an open-source Java API that implements the Fuzzy Control Language (FCL) specification, a standard for Fuzzy Control Programming published by the *International Electrotechnical Commission* (IEC). Although the full specification is not publicly accessible, the original draft standard is available on Moodle. The JFuzzyLogic API has the capability to parse a FCL file and apply basic fuzzy inference to the linguistic variables and terms defined in FCL. In this lab we will construct a set of linguistic variables, membership functions and fuzzy rules, and use the fuzzy inference engine provided to compute a result.

For the following linguistic variables and values/terms: ***funding*** (inadequate, marginal, adequate), ***staffing*** (small, large), ***risk*** (low, normal, high), we define fuzzy rules:

1. **IF** funding is adequate OR staffing is small THEN risk is low
2. **IF** funding is marginal AND staffing is large THEN risk is normal
3. **IF** funding is inadequate THEN risk is high

Create a new Java project in Eclipse and add the JFuzzyLogic API to your build path. Create a new folder called ***fcl*** in the project that contains a file called ***funding.fcl***.

Create the FCL Constraints

The terms and rules for fuzzy inference are defined in FCL in terms of a function block. Add the following definition to the file ***funding.fcl***.

```
FUNCTION_BLOCK Project
END_FUNCTION_BLOCK
```

It is important to understand that the inputs and outputs to fuzzy inference engines are not fuzzy! The inputs and output represent real data defined in terms of standard data types. For the rules above, we have two inputs and one output defined inside the function block:

```
VAR_INPUT
    funding : REAL;
    staffing : REAL;
END_VAR

VAR_OUTPUT
  risk : REAL;
END_VAR
```

The first step in fuzzy inference is the fuzzification of the input variable. The two inputs are linguistic variables that can be fuzzified in terms of fuzzy values. The fuzzification process involves the specification of values/terms and their set membership functions. The JFuzzyLogic API has built-in membership functions such as triangular, trapezoidal, gaussian, bell, sigmoidal and custom (Piece-wise linear).

Add the following declarations to fuzzify the input variables into fuzzy sets:

```
FUZZIFY funding
  TERM inadequate := (0, 1) (30, 1) (45, 0);
  TERM marginal := trian 25 50 75;
  TERM adequate := (50, 0)  (70, 1) (100, 1);
END_FUZZIFY

FUZZIFY staffing
  TERM small := (1, 1)  (15, 0);
  TERM large := (6, 0)  (20, 1);
END_FUZZIFY
```

Before the final result (risk) can be computed, the output of the fuzzy inference must be defuzzified.

```
DEFUZZIFY risk
  TERM low := (0, 1)  (45, 0);
  TERM normal := trian 20.0 50.0 80.0;
  TERM high := (55, 0)  (100, 1);
  METHOD : COG;
  DEFAULT := 0;
END_DEFUZZIFY
```

Finally, the set of fuzzy rules should be declared:

```
RULEBLOCK No1
  AND : MIN;
  ACT : MIN;
  ACCU : MAX;
  RULE 1 : IF funding IS inadequate THEN risk IS high;
  RULE 2 : IF funding IS adequate OR staffing IS small THEN risk IS low;
  RULE 3 : IF funding IS marginal AND staffing IS large THEN risk IS normal;
END_RULEBLOCK
```

Apply Fuzzy Inference Using the JFuzzyLogic API

Create a new Java class with the following statements:

```
FIS fis = FIS.load("fcl/project.fcl", true); //Load and parse the FCL
fis.chart(); //Display the linguistic variables and terms
fis.setVariable("funding", 60); //Apply a value to a variable
fis.setVariable("staffing", 14);
fis.evaluate(); //Execute the fuzzy inference engine
System.out.println(fis.getVariable("risk").getValue()); //Output end result
```