Advances in Control Education 20

Universidad Nacional de Educación a Distancia & Universidad Politécnica de Madr

Index

- Home
- Scope
- Topics
- Instructions for Authors
- Plenary Speakers
- Plenary Lectures Slides
- Committees
- Sponsors
- Program
- Conference location
- Registration
- Submission site
- Travel information
- Hotel & tourist information
- Contact

Home

ACE´06

7th IFAC Symposium on Advances in Control Education

21– 23 June 2006, Madrid, SPAIN

Organized by Universidad Nacional de Educación a Distancia & Universidad Politécnica de Madrid

In co-operation with Comité Español de Automática (CEA)

Sponsored by IFAC

Co-sponsored by IFAC TC on Control Education, IFAC TC on Developing Countries and IFAC TC on Social Impac Automation

Important Dates

Deadline for submission of papers: 20 January 2006

(EXTENDED TO 27 January 2006)

Notification of Acceptance/Rejection: 31 March 2006

Deadline for submission of final papers: 12 May 2006

Laboratory Experiments I Regular Sess

Chair: Alberto Leva Politecnico di M
Álamo University of Seville

10:30-10:50, Paper WeD01.1

A maritime control example of fuzzy logic for educational purposes

Vermeulen, Arthur Netherlands Defence Acade

10:50-11:10, Paper WeD01.2

Turning a toy into a PLC with ladder logic programming

Leva, Alberto Politecnico di M

11:10-11:30, Paper WeD01.3

An educational plant based on the quadruple tank process

Alvarado, Ignacio University of S
Limón, Daniel University of Los Ar
García-Gabín, Wiston University of Se
Álamo, Teodoro University of S
Camacho, Eduardo F. University of Se

11:30-11:50, Paper WeD01.4

Usage of BORIS environment for control of real-time laboratory model

Chalupa, Petr Tomas Bata Unive

11:50-12:10, Paper WeD01.5

New laboratory course for control education

Novák, Jakub Tomas Bata Univ
Chalupa, Petr Tomas Bata Unive
Bobál, Vladimír Tomas Bata Unive

WeB01 Roo

Software in Control Education I Regular Sessi

Chair: Antonio Sala Politech. Univ. of Vale
Santos Universidad Complutense Madrid

10:30-10:50, Paper WeB01.1

An educational example for learning fuzzy systems

Montenegro, Daniel Universidad Complutense Madr
Santos, Matilde Universidad Complutense Mad
Garmendía, Luis Universidad Complutense Madr

10:50-11:10, Paper WeB01.2

An intelligent tutoring system based on a domain ontology for the design of lead-lag controllers

García, Isaías University of
Benavides, Carmen University of I
Villar, José-Ramón University of I
Rodríguez, Francisco University of I
Aláiz, Héctor University o
Alonso, Ángel University of

AN EDUCATIONAL EXAMPLE FOR LEARNING FUZZY SYSTEMS

D. Montenegro, M. Santos, L. Garmendia*

Arquitectura de Computadores y Automática
* Sistemas Informáticos y Programación
Facultad de Informática. Universidad Complutense de Madrid
28040-Madrid

Abstract: A fuzzy inference system that implements a traffic light has been developed. It is a simple and illustrative example that allows to analyse the influence of the different fuzzy parameters on the response of the system. It can be used as an educational example for teaching, showing the main characteristics of the fuzzy inference. It also can be simulated and visualized on a Web Server. This intelligent control device is easy to modify and flexible. The fuzzy logic, the fuzzy sets and the fuzzy rules can be changed to see the effects on the results. Copyright *2006© IFAC*

Keywords: Fuzzy Control Teaching, Simulation example, Fuzzy Inference, Fuzzy Operators.

1. INTRODUCTION

Finding a simple example to show the performance of a fuzzy inference system is not an easy task. Applications that try to reflect a moderately realistic behaviour require a large number of rules; simple problems have not always a useful application in the daily world.

On the other hand, for effective teaching, it is interesting for both teachers and researchers to have some educational examples where the behaviour of a fuzzy system can be shown, with simple and basic rules, finding applications in real life and, at the same time, providing a friendly interface to help the understanding of the procedure. Other approaches enable us to identify a large number of parameters of a fuzzy model but in a theoretical environment (Piegat, 2001).

For this purpose, in this work a learning environment that simulates a fuzzy traffic light has been implemented. It is a fuzzy rule based control system with the corresponding linguistic inputs and outputs, the rules, the inference operators, and the implication methods (Zadeh, 1965; Lewis, 1997). The students can learn how to design a fuzzy system in an easy way and can see how the fuzzy parameters influence on the response.

The computational tool Xfuzzy has been selected as the software for the implementation because of its diverse functionalities. The basic features of this software package are described in section 2. In section 3 the fuzzy traffic light is implemented, showing how it can be simulated and visualized on a Web server. The parameters that can be modified are defined. Finally, Section 4 gives some ideas about the use of this example for didactic purposes.

2. THE SOFTWARE PACKAGE XFUZZY

Xfuzzy (IMSE, 2003) has been developed in Seville University, as a platform for designing inference systems based in fuzzy logic. Xfuzzy has been programmed in Java and has GNU license.

It consists of several tools that cover the different stages in the design process of a fuzzy inference system.

One of its most suitable characteristics is the capacity for developing complex systems and the flexibility to extend the set of available fuzzy functions.

The different tools of Xfuzzy can be executed independently. The simulation environment integrates them under a graphical interface that facilitates the design procedure to the users. In the main screen of Xfuzzy these tools are divided into modules (Figure 1).

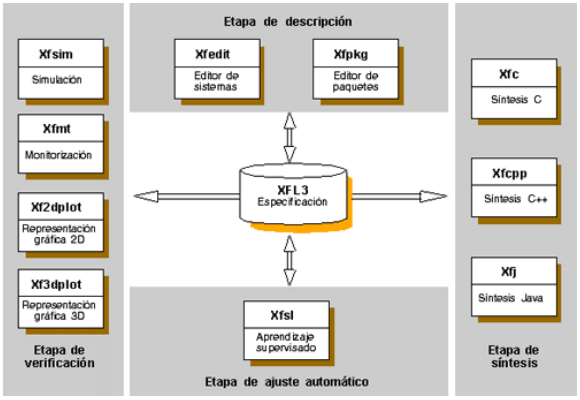

Figure 1: Main screen of Xfuzzy 3.0

- The *description* module includes graphical tools for the definition of the fuzzy system and the functions libraries.
- The *verification* unit is composed by tools for simulation, debugging and graphical visualization of the system behaviour.
- The *tuning* module facilitates the application of learning algorithms in order to improve the system.
- The *synthesis* unit includes tools to generate code in high-level languages (C, C++ and Java), for software and hardware implementations of the designed fuzzy inference system.

Xfuzzy uses the specific language XFL (XFuzzy Language). It is a flexible and powerful language that allows not only to express complex relations between fuzzy variables but to use hierarchical rule bases and connectives, linguistic modifiers, membership functions and different methods of defuzzyfication. It also serves as a link between all the tools.

3. AN EDUCATIONAL EXAMPLE OF FUZZY SYSTEM

The traffic light is a simple and illustrative example of a control system that can be implemented by using fuzzy inference. The model is based on the relationship between cars and pedestrians at a traffic light, and its behaviour will depend on the number of each one of them. Its performance is given by a set of fuzzy rules.

The S*ystems Editor* and the *Java Synthesis* tools have been used to develop the fuzzy traffic light. The tools of the *verification* unit: *Graphical Interface*, *Simulation* and *Monitorization*, can help to see the traffic light as a didactic system. The *Editor of Packages* can also be helpful to the students who can learn how to define their own functions (t-norms, co-norms, etc.) for implementing the fuzzy relations, and to see the influence they have on the response. Nevertheless, the most common functions on fuzzy control applications are already implemented in Xfuzzy. This flexibility allows to try new operators and to observe the behaviour of the system when they are changed.

3.1 Constructing the Fuzzy Traffic Light

The first step is to define the inputs and output variables. The traffic light has two inputs and one output. The first input is the pedestrian's density (P), a discreet variable whose domain is a set of integers between 0 and 20. Five fuzzy sets are defined for this input: very low (VL), low (L), average (A), high (H) and very high (VH), with Gauss bell membership functions uniformly distributed in the universe of discourse (Figure 2).

$$P = \{VL, L, A, H, VH\} \quad (1)$$

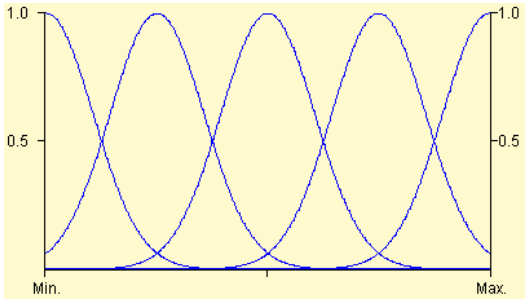

Figure 2: Pedestrians Density (P)

The second input is also a discreet one. It is the density of traffic (T), which is defined as a set of integers between 0 and 30. It is described by three fuzzy sets: null (N), moderate (M) and intense (I), with Gauss bell memberships. (Figure 3)

$$T = \{N, M, I\} \quad (2)$$

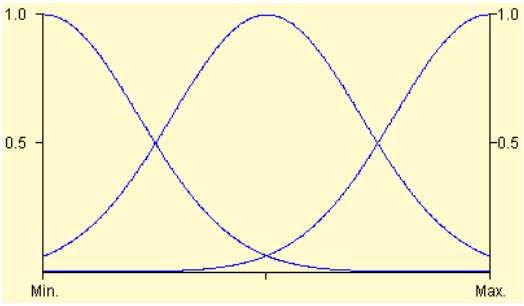

Figure 3: Traffic Density variable (T)

The output variable is the traffic light (TL). It is domain is the real numbers interval between 0 and 1. Two fuzzy sets are defined: red (R) and green (G), with triangular membership functions (Figure 4).

$$TL = \{R, G\} \quad (3)$$

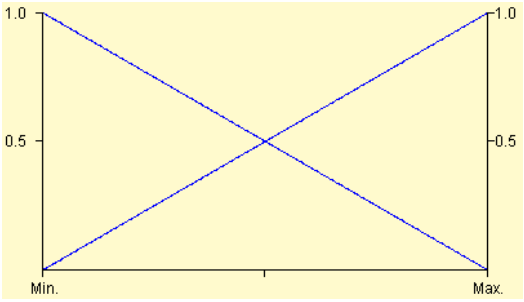

Figure 4: Traffic Light variable (TL)

The inference rules for the fuzzy system that are described in Table 1, have the following form,

If **Pedestrians** = A and **Traffic** = B
then **Traffic Light** = C (4)

where A, B, and C are the fuzzy sets previously defined for each variable.

The behaviour of the fuzzy system depends on the mathematical t-norms operators that implement: the connectives of the rules (Schweizer and Sklar, 1960), the operators to implement the implication fuzzy relations, and the defuzzyfication method. Initially the product xfl.prod() has been assigned to the operator "*and*". The default implication in Xfuzzy is Larsen (Trillas and Valverde, 1985), and the defuzzyfication method is given by the function *xfl.FuzzyMean*().

Table 1: Inference Rules

Pedestrians	Traffic	Traffic Light
Very Low	Null	Green
Very Low	Moderate	Green
Very Low	Intense	Green
Low	Null	Red
Low	Moderate	Green
Low	Intense	Green
Average	Null	Red
Average	Moderate	Green
Average	Intense	Green
High	Null	Red
High	Moderate	Red
High	Intense	Green
Very High	Null	Red
Very High	Moderate	Red
Very High	Intense	Red

Once the system is defined it is necessary to export it to a programming language for its simulation. Using the *Synthesis* tool, the system is exported generating four classes in Java (BEA System):

- FuzzyInferenceEngine.java: is the file that describes the inference method of the fuzzy system. Four methods of inference are available to implement this process.
- MembershipFunction.java: it is the file that contains the description of the interface to describe a fuzzy number. It only admits a method called 'compute' that calculates the membership degree of each value of the input variables.
- FuzzySingleton.java: this file is a class that implements the MembershipFunction interface, and it represents a crisp value as a fuzzy number (fuzzyfication).
- Trafficlight.java: is the file that contains the class that describes the complete fuzzy system. This class is a particular implementation of the *FuzzyInferenceEngine* interface. Therefore, the public methods that implement the inference are *crispInference* and *fuzzyInference*.

3.2 Simulation and Visualitation on a Web Application

It is possible to simulate the system using the exported Java classes. The BEA Weblogic server and its IDE BEA Workshop will be used to implement a *pageFlow* (Servlet for the BEA technology), that realizes the simulation. However, as the fuzzy inference system is exported as a Java class, it can be

simulated with any other visualization method: Servlets, Swing, AWT, Midlets, etc.

In this simulation the inputs are initialised with random values, and depending on the state of the traffic light, the system leaves the way clear to pedestrians or cars. The pedestrians and the cars are distributed in lanes: three for cars (all in a same direction) and two for the pedestrians, on the left and right hand sides of the cars (Figure 5).

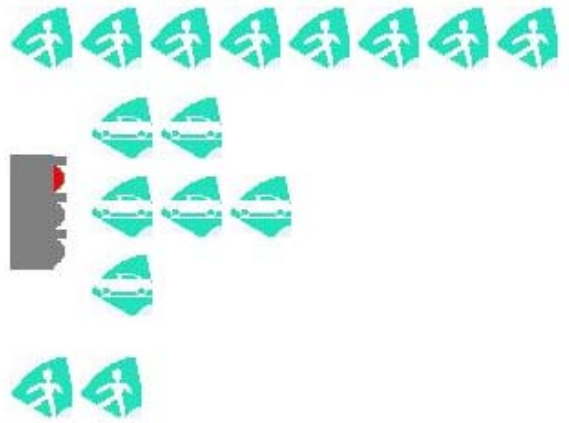

Figure 5: Lanes of pedestrians and cars.

In each simulation cycle the inference system is called to find out the state of the traffic light. The inference process results in a value that makes way to pedestrians or cars. Then, the number of the element of the corresponding lane is decreased by one. Simultaneously, new pedestrians or cars can randomly arrive to each lane, keeping the system into an infinite cycle to show the behaviour.

The yellow light means the intermediate state before any change of the light. It does not depend on the fuzzy inference system, which has only two outputs, red or green. The yellow light implements the fact that the traffic light requires a bit of time to change. The delay time is defined as 5 cycles of the system. The visualization can show the information in a realistic way, assigning half a second to each cycle.

When executing the *pageFlow* we obtain a main window with the lanes of cars and pedestrians. The traffic light will be updated each simulation cycle, making way to one or another. Figure 6 shows the screen of the system:

3.3 Instalation

The server BEA Weblogic 8.1 and IDE BEA Workshop are needed for the system operation. For development purposes, this tool is free and a copy can be asked in the BEA Web site.

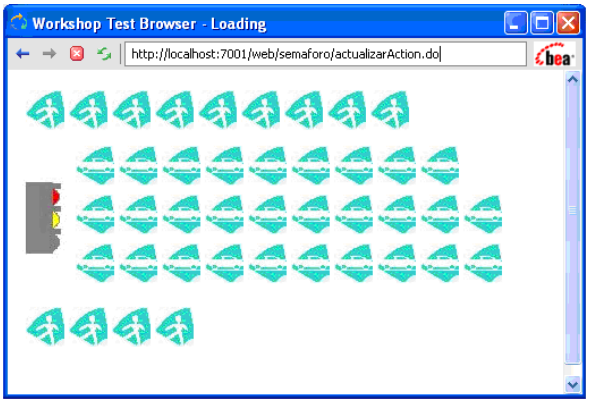

Figure 6: Traffic Light Simulation.

When having installed the server BEA Weblogic 8.1, it is necessary to create a domain to lodge the project that is going to be created. Using the default functions of the Configuration Wizard of BEA, a simple domain with all the necessary characteristics can be generated to execute the example.

The project must be created on this instance of the server as a Web Application, where the pageFlow and the visualization page are in a Web Project, and a Control Project has to be added with the classes generated by Xfuzzy. Finally the Control Project must be compiled and then it runs on the server.

4. COMPARISON OF DIFFERENT LOGICS WITH THE TRAFFIC LIGHT

This simple and illustrative example of fuzzy system allows analysing the behaviour of the functions that describe the fuzzy logic operators (Scheweizer and Sklar, 1960; Nguyen and Walker, 1966; Klir and Yuan 1995; Trillas and Valverde, 1985). It is easy to modify any of them, and then it possible to see the changes that are produced on the response of the inference system. In fact, the results obtained from the use of Minimum (Min), Product (Prod) and Lukasiewicz (W) shown in Table 1, help to choose one of these functions for the development of conjunctions depending on the properties and characteristics of the environment.

The analysis is done with certain initial characteristics: numbers of fulfilled cycles (4 different cycles: 25, 50, 75 and 100) and entrance values (5 different types from entrance values: random, minimum of pedestrians and vehicles, minimum of pedestrians and maximum of vehicles, maximum of pedestrians and minimum of vehicles, and maximum of pedestrians and vehicles); also 6 different aspects was considered for help the analysis of the traffic light: the maximum number of pedestrians and vehicles presented until the moment

Cycles		25 Cycles						50 Cycles						75 Cycles						100 Cycles					
		Maximum of Vehicles	Maximum of Pedestrian	Average of Vehicles	Average of Pedestrians	Red Cycles	Green Cycles	Maximum of Vehicles	Maximum of Pedestrian	Average of Vehicles	Average of Pedestrians	Red Cycles	Green Cycles	Maximum of Vehicles	Maximum of Pedestrian	Average of Vehicles	Average of Pedestrians	Red Cycles	Green Cycles	Maximum of Vehicles	Maximum of Pedestrian	Average of Vehicles	Average of Pedestrians	Red Cycles	Green Cycles
Random	Min	17	17	12.44	13.8	13	12	21	19	13.76	14.82	27	23	23	20	15.38	16	39	36	23	20	15.48	16.24	52	48
	Prod	20	13	11.96	8.76	17	8	20	17	13.24	10.28	32	18	22	17	14.01	11.61	45	30	22	19	14.8	12.75	58	42
	W	20	20	6.8	17.96	25	0	20	20	4.22	18.98	50	0	20	20	3.34	19.32	75	0	20	20	2.87	19.49	100	0
MaxPed MaxVeh	Min	30	20	24.88	18.72	13	12	30	20	23.38	18.3	26	24	30	20	23.29	17.7	41	34	30	20	22.62	17.52	54	46
	Prod	30	20	24.2	17.16	15	10	30	20	22.92	17.24	28	22	30	20	20.88	17.76	40	35	30	20	20.52	17.54	53	47
	W	30	20	14.08	20	25	0	30	20	7.64	20	50	0	30	20	5.66	20	75	0	30	20	4.67	20	100	0
MaxPed MinVeh	Min	15	20	7.56	16.32	10	15	21	20	11.16	17.3	23	27	27	20	15.01	17.06	36	39	27	20	17	17.49	49	51
	Prod	12	20	6.16	16.84	12	13	19	20	8.94	16.42	25	25	26	20	12.74	16.41	39	36	29	20	15.52	16.93	51	49
	W	3	20	1.2	20	25	0	3	20	1.3	20	50	0	3	20	1.4	20	75	0	3	20	1.38	20	100	0
MinPed MaxVeh	Min	30	14	17.2	8.16	19	6	30	16	15.36	11.32	32	18	30	20	16.74	13.69	45	30	30	20	18.59	14.53	58	42
	Prod	30	14	16.88	9.28	18	7	30	17	16.74	11.02	32	18	30	20	16.69	13.26	45	30	30	20	16.66	14.38	58	42
	W	30	20	11.88	13.6	25	0	30	20	6.76	16.8	50	0	30	20	5	17.86	75	0	30	20	4.15	18.4	100	0
MinPed MinVeh	Min	12	8	6.56	2.6	12	13	17	10	8.4	4.74	27	23	19	16	10.01	7.33	41	34	21	20	11.36	9.78	53	47
	Prod	12	6	6.52	1.76	12	13	14	16	7.5	4.16	26	24	17	17	8.97	7.65	38	37	22	18	10.58	9.33	50	50
	W	23	10	12.2	2.92	10	15	23	20	7.4	10.72	35	15	23	20	5.33	13.97	60	15	23	20	4.48	15.36	85	15

Table 1: The Traffic Light with Different Logics

of the analysis (to know if some of the two variables exceed the limits allowed), the average of pedestrians and vehicles (to know the efficiency the traffic light) and the number of cycles in red or green (to know the frequency of change of the traffic light). We looked for a traffic light that had the smaller average of pedestrians and vehicles (what it would indicate the efficiency) and have a fast answer of stabilization in front of boundary situations.

In this environment, the aggregation of the inferred knowledge has different behaviour for different logics.

We can observe in the results that the functions of Minimum (Zadeh´s logic) and Product with unbalanced entrances (one with maximum capacity and the other one with minimum) there is a certain number of cycles that the traffic light takes to find a balance and an order of flux of vehicles and pedestrians.

When using the Lukasiewicz (W) logic, the behaviour of the traffic light was erratic because the Lukasiewicz t-conorm (Scheweizer and Sklar, 1960) to compute the disjunctions of the outputs of each implication rule is the sum, and so, when many rules are given it always tends to 1 (all light cycles are red), and so the light never changes its colour. With the use of the Zadeh´s logic (t-norm Minimum and t-conorm Maximum) and the Product logic the results are good and very similar.

The logic that balances best the number of cycles red and green is the Zadeh´s logic. Note that the used aggregation of the consequences of the rules is the maximum t-conorm, which is the lowest t-conorm, and gives a very balanced results. The probabilistic sum sum t-conorm is a little bit higher than the maximum and gives a little bit unbalanced cycles, and the Lukasiewicz t-conorm is too much unbalanced, because the final knowledge obtained by every rule for the output variable is added. So we give an example in where a given logic is working fine and another logic is not. Any other real live examples could work best with a different logic that represents better its behaviour.

The traffic light is more efficient to give passage to vehicles and pedestrians when the number of cycles is increased.

This work can be used as an educational example for teaching the main characteristics of the fuzzy inference process and the influence of the chosen parameters and chosen fuzzy logic.

It can be simulated and visualized on a Web Server (Mishra and Sharma, 2004).

5. FURTHER WORK

Another change that could be considered is to increase the number of input variables, i.e., considering each line of pedestrians and/or cars as independent inputs. This would allow the use of other operators as the t-conorms (Scheweizer and Sklar, 1960) for the logical operator "*or*" in the definition of the premises of different rules, and not just to aggregate the information of the conclusions.

The rules can also be changed to contain negations and then enlarging the range of possible solutions.

ACKNOWLEDGES

Authors would like to acknowledge the support of the UCM Project on Innovative Education PIE 2003/7, MCyT MTM2005-08982-C04-01, and CAM Ref. 910149.

REFERENCES

BEA Systems. BEA WebLogic Help. http://www.bea.com.

Klir G. and B. Yuan (1995). *Fuzzy Sets and Fuzzy Logic Theory and its Applications*. Prentice Hall.

Lewis, H.W. (1997). *The Foundations of Fuzzy Control*. Springer.

Mishra, S. and Sharma, R. (2004). *Interactive multimedia in Education and Training*. Ed. Idea Group Inc (IGI).

Nguyen H.T. and E. A. Walker (1996). *A first course in Fuzzy Logic***.** CRC Press.

Piegat, A. (2001). *Fuzzy Modeling and Control*. Springer

Schweizer B. and A. Sklar (1960). *Probabilistic Metric Spaces***.** North-Holland. New York.

Trillas E. and L. Valverde (1985). On mode and implication in approximate reasoning. *Approximate reasoning in expert systems*. Eds. M. M. Gupta. North-Holland. pp. 157-166.

IMSE (2003). Centro Nacional de Microelectrónica. *Herramientas de CAD para Lógica Difusa*. Xfuzzy 3.0. Seville. http://www.imse.cnm.es.

Zadeh L.A. (1965). Fuzzy sets. *Information and Control* 8, pp. 338-353.