Extending Jess with Type-2 Fuzzy Logic

Carelia Gaxiola-Pacheco, Dora-Luz Flores, Manuel Castañón-Puga, Antonio Rodríguez-Díaz, Juan-Ramón Castro, and Iván Espinoza-Hernández

Universidad Autónoma de Baja California,
Tijuana, Baja California, Mexico
{cgaxiola,dflores,puga,ardiaz,jrcastror,espinoza.ivan}@uabc.edu.mx

Abstract. Jess (Java expert system shell) has been used as an inference system to model decision-making in multi-agent systems (MAS). In addition, fuzzy inference systems (FIS) have been mostly used to regulate, control and implement decision-making systems. To extend its functionality, we present an extension to Jess applying interval type-2 FIS. We conclude with a case of study as an example.

Keywords: Java, Expert System, Type-2 Fuzzy Logic.

1 Introduction

Traditional artificial intelligence approaches are limited in developing autonomous robots and social simulations, however fuzzy logic, neural networks, genetic algorithms, and symbolic processing may be useful to improve their performance. While machine learning is considered probably the most difficult task to improve, collaboration between robots or agents is also a crucial feature to consider. These types of systems will be most useful when many of these features are networked together, but if getting one system to intelligently operate in the real world is extremely challenging, then getting several of them to work together is even more difficult [1]. From a computational point of view, fuzziness and distribution can be tackled with hybrid intelligent software agents. The use of fuzzy logic in agents has previously shown satisfactory results [2]. Computational models such as AvatarSim, FLAME, and PETEEI use type-1 fuzzy logic to simulate emotions and personality in agents [3].

1.1 Type-2 Fuzzy Logic

A type-2 fuzzy logic (T2FL) tool was developed as a toolbox for Matlab by [4] and has been used for control applications [5], edge detection in digital images [6], and more. Castro et al. show the creation of a fuzzy inference system using this tool in [7]. A novel tool for simulations based on agents named Wiinik has been developed [8], where an agent's behavior is determined by an interval type-2 fuzzy inference system (IT2FIS), which is designed to define the agent's psychological elements, semantic networks are used to represent the information

Advances in Soft Computing Algorithms
Research in Computing Science 49, 2010, pp. 121-129

acquired by the agent from its environment, allowing the agent to communicate in a more meaningful way. Semantic networks, although not standardized, have great potential to model any kind of knowledge, regardless of an agent's capability.

1.2 Fuzzy Jess

The Integrated Reasoning Group of the Institute for Information Technology of the National Research Council of Canada has developed two fuzzy logic software packages: FuzzyCLIPS and FuzzyJ [9]. FuzzyCLIPS is an extension of the CLIPS (C Language Integrated Production System), which is an expert system shell developed by NASA. It integrates a fuzzy reasoning engine to CLIPS facts [10]. The FuzzyJ Toolkit is a Java API (Application Program Interface) for handling fuzzy logic systems. It can be used as a standalone system or integrated with Jess (FuzzyJess). FuzzyJess provides similar capabilities but it is more flexible than FuzzyCLIPS. Inputs and outputs can be handled using Java APIs. FuzzyJess is the rule engine used for implementing fuzzy sets and the reasoning mechanism used to simulate an agent's performance. Epstein et al. and Martínez et al. present tools which allow simulations in a MAS where agents decide according to fuzzy-logic rules in [11, 12] .

1.3 Multi-Agents Systems

In recent years, the development of computational capabilities has redefined the way in which philosophers, social scientists and other researchers see the world. Scientific enquiries have generally shifted from paradigms based on a linear conception of the world to a nascent, more holistic view based on non-linearity. The distinction between linear and non-linear science has not been fully resolved in the literature [13], but as a working definition for this paper, we can say that the linear paradigm of the past is based on the assumption of independence of whatever atoms or agents are used to describe reality through a given model. In stark contrast, the growing non-linear view of the world is based on a perception of ingrained interdependence. The most common way of describing the difference between the linear and non-linear paradigms is that in the former the aggregate is equal to the sum of its parts, while in the latter the aggregate is more than the sum of its parts [14].

Simulation has been adopting multi-agent systems as one of the most promising ways to achieve social computational models. Agent-based models facilitate the analysis and design of models, once that agents' behavior is part of a theoretical system, this behavior can be computationally implemented. Then the language of agents from the point of view of analysis and the language of agents from the point of view of design are very similar, but not necessarily equal.

From the scientific point of view, MAS depend on the available methodology: the easiness offered by the analysis of actor and context to construct theoretically agents. MAS from the computational point of view, depend on the available

technology, this means, the easiness offered by the software and hardware design to generate software agents [15].

2 Extending Jess with Type-2 Fuzzy Logic

The addition of fuzzy logic to Jess is not new, existing extensions such as FuzzyJess provide a traditional type-1 fuzzy extension, allowing the rules to be written as fuzzy rules and providing a variety of membership functions and modifiers to describe linguistic variables. While linguistic variables provide a simple and appropriate way to represent a range of values that have different degrees of belonging, these ranges must be given a single, precisely defined value. T2FL extends fuzzy sets by allowing a range of values to define the range of belonging of a given element in a fuzzy set. By modeling the uncertainty on a fuzzy set we are able to widen our definitions or more precisely model our views on what does and does not belong in a concept.

When modeling real world abstract concepts such as hot, cold, fast or slow, using a type-1 membership function can be made to fit the concept but it will be tied to the views of a very small set of opinions, what is considered hot for some might be considered not so hot or even cold for others. With type-2 fuzzy sets we are able to represent this disparity in opinion for each concept. Depending on the application, this advantage can be used to give greater margin of acceptance when processing signals with noise, recognising imagery with a neuro-fuzzy network, representing the opinions of a group on certain issues or defining abstract concepts in natural language processing.

In this paper we discuss the possibility of using T2FL in conjunction with a rule-based system to control the behavior of computer agents. Agent models adhering to the production rule system will need a rule interpreter. Among the many options Jess presents itself as simple and adequate, its features are sufficient for most tasks and when it would seem that we reach a limit to what we can do, it is also capable of handling extensions.

Using existing methods, T2FL comes with a considerable performance cost. As it is a relatively new approach, we hope that this issue will be resolved by the introduction of efficient algorithms and the increase in computational capabilities. Taking into account this limitation, our work focuses on the potential modeling benefits rather than on immediate practical usage.

2.1 Adding Interval Type-2 Fuzzy Inference System to Jess

Using type-2 fuzzy extension with Jess is as simple as importing a library and using Java objects. The extension is self contained in a Java JAR file, few user functions are defined to facilitate creating fuzzy sets and it is mainly handled as a typical Java object. What this means is that, unlike FuzzyJess, using this extension does not require re-engineering any rules to conform to a fuzzy rule and it also does not require the addition of any code to the standard Jess interpreter.

Inputs and outputs are added to this object which will be used by a set of rules that must also be defined. The general layout in creating a working FIS is in essence exactly the same as any common type-1 inference system, the only notable difference will be defining membership functions and the defuzzified value obtained for outputs. Instead of returning a single numerical value, we can obtain three values: a leftmost point, a rightmost point, and a centre point value.

The following code demonstrates creating a FIS object, adding one input and one output, and two linguistic variables for each:

```
(bind ?fis ((new Fis))
(call ?fis addInput "Time")
(call ?fis addToInput "Time" "Fast" (IGaussMT2 ?ls1 3.0 0.6 1.4))
(call ?fis addToInput "Time" "Slow" (IGaussMT2 ?ls1 3.0 9.6 10.4))
(call ?fis addOutput "Score")
(call ?fis addToOutput "Score" "0" (ITriT2 ?ls4 -1 0 1 -1.2 0.2 1.2))
(call ?fis addToOutput "Score" "1" (ITriT2 ?ls4 0 1 2 0.2 1.2 2.2))
```

2.2 Adding Fuzzy Rules into Extension

Once a FIS object has been created and at least one input and one output have been added, it will now be possible to add the rules that will determine its behavior. Adding a rule is demonstrated with the following code:

```
(?fis addRule "[Time:Slow] [Cost:Cheap] [Capacity:Low]" "[Score:1]")
```

To add a rule, the method requires two formatted strings as inputs. The first string will identify linguistic variables for inputs, each set of input and linguistic variable surrounded by brackets and each input name and linguistic variable separated by a colon. The seconds string follows the same pattern to identify outputs and their linguistic variables.

3 Case of Study

We illustrate our proposal with a simple case taken from [16, 17] where a planning mechanism is required to schedule tasks within a community of autonomous agents. Each agent represents a production unit in a factory and some agents must negotiate task allocation according to available resources. Figure 1 shows the structure of a given organisation, which involves at least three agent types: Production Planning Agent (PPA), Production Managing Agent (PMA) and Production Agent (PA).

Several agents can be created with PPA, PMA and PA roles. Each agent can use a FIS to resolve input values depending on the assigned role. Agent interactions create a dialogue between roles where each agent constantly make decisions depending of the plan and the tasks to be processed.

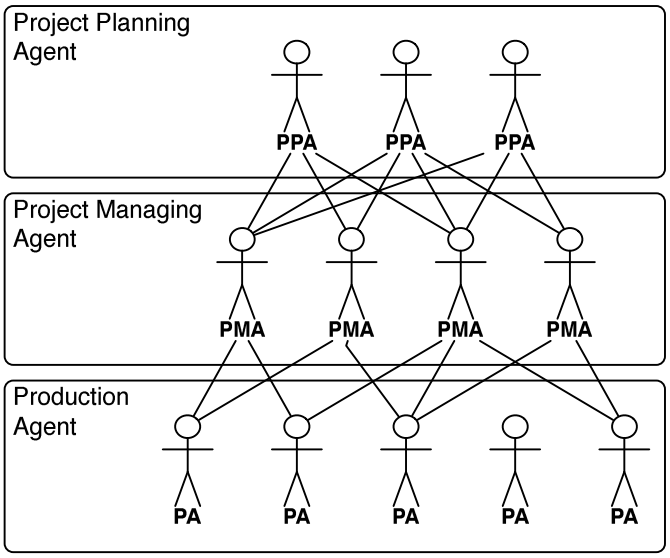

Fig. 1. Each PA represents a production unit in a factory. A PMA must negotiate the delegation of work and decomposition of tasks according to available resources in order to meet production schedules set by the PPA.

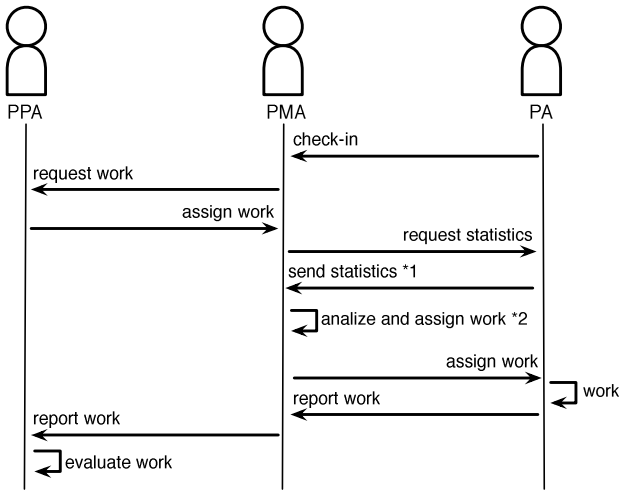

Fig. 2. Script flow Company.

3.1 Fuzzy Decision Making System into Organizational Structure

In this scheduling example, the PPA is in charge of project planning. The PMA performs project management in terms of contracting the best possible PA (in terms of operational costs, the delivery time and current capacity availability). The PA represents the lowest level production unit that simulates or encapsulates shop floor production process.

We will compare the case where the PMA must decide which PA seek to assign new tasks according to their production capacity.

Nonfuzzy approach with Jess In the nonfuzzy logic approach, the PMA can opt for an election where the PA has an absolute value at the time of production in comparative with others.

For example, the following rule represents the decision of PMA to choose the PA within available resources.

```
(bind ?score1 ( + (/ ?co1 20) (/ (- 10 ?ti1) 10) (/ ?ca1 3)))
(bind ?score2 ( + (/ ?co2 20) (/ (- 10 ?ti2) 10) (/ ?ca2 3)))
(if (< ?score1 ?score2) then (return 1))
(if (> ?score1 ?score2) then (return -1))
(return 0)
```

Each variable is given equal weight by normalising its value and adding them giving a maximum value of three for the best option and a minimum of zero for the worst option. This process allows each agent to be given a score and based on this value, they can be arranged in descending order making the best option the first in the list.

Type-1 Fuzzy Approach with FuzzyJess In a fuzzy approach, the PMA will not receive a numerical value but a fuzzy set as input. From this it must determine the score in a similar fashion as the previews case using fuzzy rules. The inputs, outputs and rules are set up as it is described in Table 1.

Table 1. PMA linguistic variables and values.

Rule	Time	Cost	Capacity	Score
r1	slow	cheap	low	1
r2	slow	cheap	high	2
r3	slow	expensive	low	0
r4	slow	expensive	high	1
r5	fast	cheap	low	2
r6	fast	cheap	high	3
r7	fast	expensive	low	1
r8	fast	expensive	high	2

The following code demonstrates how a type-1 fuzzy rule would look like using FuzzyJess. It should be noted that they are written in the similar form as Jess rules, making it necessary to mix fuzzy rules with nonfuzzy rules for our case of study where we control an agents behavior with rules.

```
(defrule r1
(theTime ?ti& :(fuzzy-match ?ti "slow"))
(theCost ?co& :(fuzzy-match ?co "cheap"))
(theCapacity ?ca& :(fuzzy-match ?ca "low"))
->
(assert (theScore (new nrc.fuzzy.FuzzyValue ?*scoreFvar* "1")))
)
```

Example of the same rule with the proposed extension would be written as:

```
((?fis addRule "[Time:Slow] [Cost:Cheap] [Capacity:Low]" "[Score:1]")
```

Type-2 fuzzy Approach with T2FIS Extension In a type-2 fuzzy approach, the PPA must not only choose an agent without an accurate production time, but also where each linguistic variable involves a degree of uncertainty, and will therefore have to interpret not only the evaluation but consider the degree of uncertainty in each assessment.

For example, the following rules represent the decision of a PMA to choose the best PA within the available options.

```
(bind ?score1 (?dfscore1 getXcoa))
;consider range of uncertianty in answer
(bind ?dif1 (- (?dfscore1 getXl) (?dfscore1 getXr)))
(bind ?score2 (?dfscore2 getXcoa))
;consider range of uncertianty in answer
(bind ?dif2 (- (?dfscore2 getXl) (?dfscore2 getXr)))
(if (> ?dif1 ?dif2) then
(bind ?score2 (+ ?score2 0.3)))
(if (< ?dif1 ?dif2) then
(bind ?score1 (+ ?score1 0.3)))
```

Uncertainty is not given as much weight as the three variables since a small amount of uncertainty is preferred to a long amount of time for job completion for this example. There are many factors to consider when determining how to weight each variable, every job will have different needs and there must be a certain tolerance level for uncertainty when dealing with real scheduling problems.

Figure 3 shows PA's outputs using different approaches: nonfuzzy, type-1 and type-2 fuzzy logic. Uncertainty can be observed in T2FL output. Compared with nonfuzzy output, which shows only a single value as a result, a whole range

of values can be obtained using a fuzzy approach, leading to a a more realistic estimate of an area where our actual value may be.

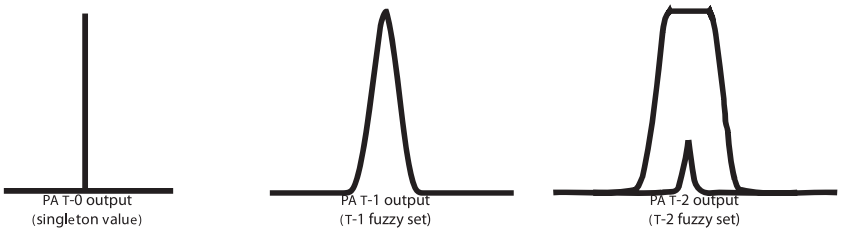

Fig. 3. PA's outputs: nonfuzzy, type-1 and type-2 fuzzy logic.

4 Conclusions and Future Work

It is possible to extend the functionality of Jess to introduce a type-2 fuzzy inference system. We used the example of Planning-Management-Resource to validate its capability.

Current work is focused on type-2 fuzzy inference systems for modeling real cases in order to explore its application.

As future work the possibility of extending Jess with a learning system to strengthen the system of type-2 fuzzy inference is being considered, so that agents change their behavior by evaluating, modifying and/or adding new rules in its inference system to build agents that adapt to the circumstances.

The vision for the future is to have a software agent model with an intelligent hybrid approach allowing better representation of MAS.

References

1. Long, L.N., Hanford, S.D., Janrathitikarn, O., Sinsley, G.L., and Miller, J.A.: A Review of Intelligent Systems Software for Autonomous Vehicles. In: Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Security and Defense Applications, (2007)
2. Sharma, S., Singh, H., and Prakash, A.: Multi-agent modeling and simulation of human behavior in aircraft evacuations. IEEE Transactions on Intelligent Transportation Systems, 44(4), (2008) 1477–1488
3. El–Nasr Seif, M., and Yen, J.: Agents, Emotional Intelligence and Fuzzy Logic. In: Proceedings of NAFIPS. Florida, US (1998)
4. Castro, J.R., Castillo, O., and Martínez, L.G.: Interval Type-2 Fuzzy Logic Toolbox, Engineering Letters, 15(1), (2007) 89–98
5. Castro, J.R., Castillo, O., and Melin, P.: An Interval Type-2 Fuzzy Logic Toolbox for Control Applications, FUZZ-IEEE, (2007), 1–6

6. Mendoza, O., Melin, P., and Castro, J.R.: The Use of Interval Type-2 Fuzzy Logic as a General Method for Edge Detection. In: IFSA/EUSFLAT Conf., (2009) 774–779
7. Castro, J.R., Castillo, O., Melin, P., and Rodríguez–Díaz, A.: Building Fuzzy Inference Systems with a New Interval Type-2 Fuzzy Logic Toolbox. Transactions on Computational Science, vol 1, (2008) 104–114
8. Flores, D.L., Rodríguez–Díaz, A., Castro, J.R., and Gaxiola, C.: TA-Fuzzy Semantic Networks for Interaction Representation in Social Simulation. Evolutionary Design of Intelligent Systems, SCI, Springer Berlin / Heidelberg (2009) 257–270
9. Firedman-Hill, E.J.: Java Expert System Shell. Available from http://herzberg.ca.sandia.gov/jess. (1998)
10. $http://www.iit.nrc.ca/IR_public/fuzzy/$
11. Epstein J.G., Michael Mohring,M., and Troitzsch, K.G.: Fuzzy-Logical Rules in a Multi-Agent System. In: SimSocVI Workshop, Groningen, (2003) 25
12. Martínez–Miranda, J., Aldea, A., and Bañares–Alcántara, R.: Agent Based Simulation in the Selection of Work Teams. Computación y Sistemas. 7(3) ISSN 1405-5546. (2004) 210–223
13. Abbott, R.: Putting complex systems to work. Complexity, 13(1) (2007) 30–49
14. Bar–Yam, Y.: Dynamics of Complex Systems, Addison–Wesley, Reading, MA, (1997)
15. Suarez, E.D., Rodríguez-Díaz, A., and Castañón-Puga, M.: Fuzzy agents. In: Castillo O, Melin P, Kacprzyk J, Pedrycz W (eds) Soft computing for hybrid intelligent systems, vol 154. Studies in computational intelligence. Springer, Berlin / Heidelberg (2008)
16. Pechoucek, M., Riha, A., Vokrinek, J., Marik, V., and Prazma, V.: Explantech: Applying multi-agent systems in production planning. International Journal of Production Research 40 (15):(2002) 3681–3692
17. Pechoucek, M., Rehak, M., Charvat, P., Vlcek, T., and Kolar, M.: Agent-based approach to mass-oriented production planning: Case study. In: IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 37 (3):(2007) 386–395