Multi-agent Approach for Visualisation of Fuzzy Systems

Binh Pham and Ross Brown

Faculty of IT, Queensland University of Technology,
GPO Box 2434 Brisbane AUSTRALIA
{b.pham,r.brown}@qut.edu.au

Abstract. Complex fuzzy systems exist in many applications and effective visualisation is required to gain insights in the nature and working of these systems, especially in the implication of impreciseness, its propagation and impacts on the quality and reliability of the outcomes. This paper presents a design of a visualisation system based on multi-agent approach with the aim to facilitate the organisation and flow of complex tasks, their inter-relationships and their interactions with users. This design extends our previous work on the analysis of the fundamental ontologies which underpin the structure and requirements of fuzzy systems.

1 Introduction

Many real world problems can be represented as complex fuzzy systems which may involve a large amount of fuzzy data, fuzzy variables and fuzzy relationships. Fuzzy logic has been used extensively to model these systems in many application areas, ranging from engineering, science, medicine to environmental planning and social sciences [13]. While mathematical models are based on algebraic operations (equations, integrals), logic models rely on logic-type connectives (*and*, *or*, *if-then*), often with linguistic parameters, which give rise to rule-based and knowledge-based systems. Fuzzy logic models can combine both of these types of modelling via the fuzzification of algebraic and logical operations [1]. There are three common classes of fuzzy logic models: *information processing models* which describes probabilistic relationship between sets of inputs and outputs; *control models* which control the operations of systems governed by many fuzzy parameters; and *decision models* which model human behaviour by incorporating subjective knowledge and needs, using decision variables [6].

For some applications, fuzzy systems often perform better than traditional systems because of their capability to deal with non-linearity and uncertainty. Another advantage is that linguistic rules, when used in fuzzy systems, would not only make tools more intuitive, but also provide better understanding and appreciation of the outcomes. However, the complexity arisen from information uncertainty makes it more difficult for humans to understand the way these systems work, especially how

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2659, pp. 995–1004, 2003.

to interpret the implication of the impreciseness of each variable on its interaction with other variables, and how the propagation of such impreciseness affects the level of confidence in the outcomes at every stage.

Visualisation has been used extensively during the last decade, but the bulk of research work has been focused on those systems which involve crisp data and crisp relationships. A few current approaches have some limitations due to either their ad hoc nature, or their ability to deal with only a specific aspect of the problem of visualisation of fuzzy systems [2, 4, 5, 8, 10, 11]. In addition, visualisation methods are often focused on data sets and only loosely coupled with the analytical process. It is left to users to decide how they deploy those visualisation tools provided.

The usefulness of a visualisation system would therefore be enhanced if it is driven primarily by those tasks that need to be performed, and not by data sets because such a system would link more tightly with the analytical process which underpins human understanding and decision making. Another aspect that needs to be considered is how to cater for different types of users.

In a previous paper [3], we presented a thorough analysis on visualisation requirements for fuzzy systems by investigating the fundamental ontologies which underpin the structure and requirements of these systems. This paper focuses on the design of a multi-agent based visualisation framework with the aim to facilitate the organisation and flow of complex tasks, their inter-relationships and their interactions with users.

Section 2 discusses briefly the characteristics of fuzzy systems and their visualisation requirements. Section 3 provides the motivations behind the multi-agent approach and an overview of the system. Subsequent sections present the structure and activities of each of these agent classes, and plan for their implementation.

2 Fuzzy Systems and Their Visualisation Requirements

To design an effective generic framework for visualization of fuzzy systems, we need to understand their essence: what are they composed of, how things are related to each other, what activities are being performed, and who are the main users of these systems. To facilitate the understanding of the rest of this paper, we briefly describe these requirements here. More detailed analysis on these requirements was presented in [3].

A typical fuzzy system consists of 6 main components: entities, data objects, relationships, events, tasks and outcomes. The *entities* include both physical (e.g. machines, workers) and abstract (e.g. returns of investment). *Data objects* may be represented in different forms: numerical, symbolic (e.g. rules), visual (e.g. diagrams, images) or audio. *Relationships* which underpin the working of a fuzzy system can be classified into five categories: data-data, data-task, data-user, task-task, and user-user. Each of these categories needs to be examined carefully in order to find appropriate visualization methods to facilitate the understanding of these relationships. *Events* change the system state and exert influence on the system performance, hence it is crucial to note and record them. To distinguish the level of complexity of *tasks*, they

can be grouped into low-level and high-level tasks. The former includes the computation of numerical data, degree of fuzziness and the operation of fuzzy rules. The latter covers the detection of unusual patterns, data mining, learning process, optimization and prediction. The *outcomes* of a fuzzy system include not only the values of state variables, but also the level of acceptance of quality, degree of confidence, and degree of impreciseness of the outcomes.

We wish to examine the visualization requirements for fuzzy systems from user- and task-oriented points of view, so that a user is allowed to interact and select what to visualize and how to do it on the fly. Thus, it is also necessary to distinguish three main types of users and their different needs. The *users of fuzzy systems* wish to be able to interpret data and its salient characteristics, to understand the implication of each decision by setting up 'what-if' scenarios, and to adapt the system to their individual needs and preferences. The *designers of fuzzy systems*, on the other hand, require information on the internal structures of these systems for planning, verification and analysis. They also seek for conditions under which optimal solutions are obtained at each stage. The *designers of visualization systems* wish to understand how users make use of visualization techniques and the effectiveness of these techniques with the intention to identify drawbacks and to find ways to continuously improve the systems. We categorise four main types of visualization tasks:

- *Interactive exploration* to provide insights into: the degree of uncertainty of each variable and its effects on each task; the inter-dependency of two or more fuzzy variables or fuzzy rules; and the effects of different operations performed on fuzzy rules.
- *Automatic computer-supported exploration* to automatically highlight salient characteristics and unusual results; to display and compare alternatives (e.g. using statistical analysis); to optimize tasks under specified constraints; and to support batch processing of tasks via scripting or visual languages.
- *Capturing feedback from users* such as instructions on tasks; input parameters, variables, constraints; users' preferences, judgements and desired degree of fulfillment of outcomes in qualitative forms.
- *Capturing users' profiles and adaptation* in order to re-organise data and re-prioritise tasks to suit; and to automatically provide tasks and data according to detected patterns.

3 A Multi-agent Visualisation Framework

Our aim is to design a systematic framework based on a high level of abstraction, where visualisation is driven by users' needs which in turn are driven by application tasks and personal view points. Search and navigation methods and tools should be context-sensitive and should operate only on relevant information space. Thus, data should be organised according to task requirements to ensure efficiency.

To this end, we propose a visualisation framework based on 5 classes of agents: control agent, computation agent, symbolic agent, visualisation agent and profile agent. Fig. 1 shows a schematic diagram of the system architecture.

The *control agent* receives users' input which includes specifications, queries and parameters. Based on such input, this agent distributes tasks to appropriate agents. It also receives results and demands from other agents when a task is completed or when further information is needed. Another duty for this agent is to generate new tasks if required based on the results sent by other agents. The control agent may be viewed as a representative of the user in an automatic mode. In our model, the user can be included in the loop and allowed to intercept the control agent in order to give different instructions if desired. The user can also intercept other agents to select different methods for performing an operation instead of the default ones built into the system. The *computation agent* performs all numerical computation required by the system (e.g. statistics, probabilistic calculus, rough set operations, fuzzy set operations). It receives instructions from both the control agent and the visualisation agent. The *symbolic agent* makes use of the knowledge base to performs rule inferencing. It receives instructions from both the control agent and the visualisation agent. The *visualisation agent* receives instructions from the control agent and request information from the computation agent and rule agent in order to select appropriate visualisation techniques to provide displays. The results of the display then trigger the control agent or the user to issue another task. Another cycle then continues.

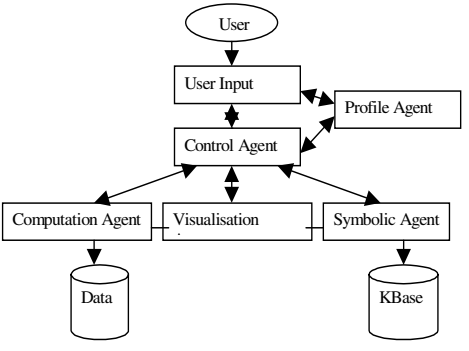

Fig. 1. System overview diagram.

The *profile agent* records the pattern of the user's behaviour in terms of the selection of tasks, visualisation techniques, numerical methods or inference rules. Based on this information, the profile agent then modifies the instructions issued by the control agent (e.g. re-prioritise tasks, change preferences, modes of display, etc.).

The following subsections show, using an object-based paradigm, the general methods within the agents as a specification of their functionality.

3.1 Control Agent

The control agent maintains a list of the other agent instantiations within the system and thus is able to control the flow of data around the agent-based visualisation system. The main tasks of the control agent are to process user events, distribute tasks to other agents and to process agent results and demands. Therefore it communicates with the Profile, Computation, Visualisation and Symbolic Agents. It is a form of automatic user within the system. However, the user can override its tasks through the *ProcessUserEvent* method.

Description	**Parameters**	**Outputs**
ProcessUserEvent-processes events generated by the user and other agents in the system.	*Specifications*–Choice of actions by the user. *Parameters*–for the specified action by the user.	
DistributeTask-takes any tasks as defined by the user and other processes and distributes them to other agents within the system.	*AgentID*–ID number of the agent to be notified. *TaskParameters*–parameters for the task to be sent to the agent.	
ProcessAgentResults-takes the results from agents and decides on next step in visualisation sequence. Involves distributing further tasks to other agents.	*AgentID*–ID number of the agent returning results. *TaskResults*–results returned by the agent.	*AgentID*–ID number of agent to pass on new task. *TaskParameters*–parameters for the new task.
ProcessAgentDemands-receives demands from other agents in the system to perform a task. Involves distributing further tasks to other agents.	*AgentID*–ID number of the agent making a demand. *DemandParameters*— parameters for the demand by the agent.	*AgentID*–ID number of the agent to accept the new task. *TaskParameters*–parameters for the new task.

3.2 Computation Agent

The main items stored by this agent includes the data to be visualised within the system, and a complete list of operations that can be performed by the computation agent. It is not an autonomous agent, as it is entirely controlled by the visualisation and control agents.

The computation agent processes the data upon requests from the control and or visualisation agent–for example, statistics, probabilistic calculus, rough set operations, fuzzy set operations, etc.

Description	Inputs	Outputs
ProcessData-processes the data according to user specifications or agent specifications in the parameters.	*DataOpID*–ID number of computation to enforce on data. *ProcessParameters*–parameters for computation, including data to load.	*ProcessResults*–results of computations.

3.3 Symbolic Agent

This agent is the interface to the knowledge database for the visualization system. It is used by the control agent and the visualization agent. It is not autonomous, as it simply provides a front end query interface to the knowledge database.

The main function of the symbolic agent is to process queries directed at the knowledge database. This database contains knowledge of appropriate visualisation techniques for the fuzzy data. Thus, the agent returns the appropriate information about techniques, parameters to use etc., as responses to queries from the control and visualisation agents.

Description	Inputs	Outputs
ProcessQuery-a query is made upon the knowledge base within the symbolic agent and it then returns inferences for the other agent to enact within the visualisation task.	*AgentID*–ID number of agent seeking inference from knowledge base. *QueryParameters*–parameters for the knowledge base query.	*QueryResults*–results returned from knowledge base query.

3.4 Visualization Agent

The visualization agent handles the graphical rendering of data to the output device. This agent draws direction from the symbolic agent, and is able to thus recommend a visualisation technique automatically, based upon the qualities inherent in the data. The data is received from querying the computational agent, which is directed by the visualisation agent to provide the data in a valid format for the visualisation technique.

There is only one major function listed, as the visualisation agent is fairly autonomous in its ability to organise a visualisation of data. Any information required for the visualisation is queried from the computational and symbolic agents,

by using their querying methods. The control agent can, at the behest of the user, override the visualisation agent by making a call to the VisualiseData method.

Description	Inputs	Outputs
VisualiseData-produces a visualization of the data passed in to this function.	*VisOpID*–ID number of visualization technique to use. *VisParameters*–parameters for chosen visualization technique including the data to visualise.	*VisResults*–results of visualisation in form of image or movie.

As an example, we trace the execution of the visualization of the Iris data shown in a previous paper, which is a commonly used test data set for classification algorithms. The classification is done based on a training set of 75 plants, 11 fuzzy rules, 4 features (sepal length, sepal width, petal length and petal width) and 3 classes. In this case the user commences the system and inputs the Iris data file as the beginning of the visualization. The control agent then commences the dialog by noting the multidimensional nature of the data to be visualized. Information about the nature of the visualization is elicited from the user, who specifies a visualization for rule culling purposes. The control agent then passes this information onto the visualization agent.

The visualization agent then queries the symbolic agent for suggested visualization techniques. The symbolic agent replies with a suggestion of using the parallel coordinate visualization technique as shown in Fig. 3 (Left) [2]. In this method, n Cartesian coordinates are mapped into n parallel coordinates, and an n-dimensional point becomes a series of (n-1) lines connecting the values on n parallel axes. The visualization agent then requests the fuzzy data in an appropriate format for the parallel coordinate technique. Therefore, the visualization agent commences the rendering of the 2D parallel coordinate visualisation. However, the user requires a 3D version of the visualization, and chooses this using the appropriate menu options.

The control agent at this stage interrupts the visualization agent thread, and then enforces a 3D parallel coordinate visualization, as shown in Fig. 3 (Right). This is then rendered by the visualisation agent, whereupon control is given back to the user to interact with the visualization. This process is then repeated until termination by the user.

4 Visual Features

In previous work we have analysed in detail the mapping of various visual features to visualisation of fuzzy logic information [3]. In this section, we summarise some of the most importance features and show examples of their mappings to visualisation tasks.

Hue is heavily used to highlight data that is different, or to represent gradients in the data [9][12][8]. For fuzzy data, it can also be used to categorise the membership of a particular data point. In the example in Fig. 2 (Left), we see that the membership

of different fuzzy terms can be illustrated by different hues. The region where the colours overlap indicates intuitively the location where these membership functions share areas of the domain.

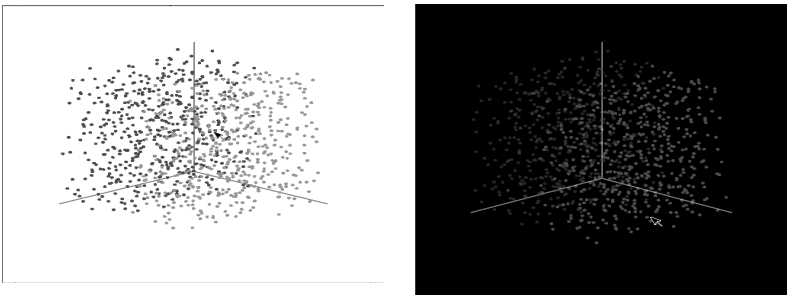

Fig. 2. (Left) Different colours used to indicate membership of different membership function terms. In this case, the red indicates slow motion, while the green indicates fast motion. (Right) Intensity difference as an indication of the membership degree for a fuzzy membership function Hot.

Luminance may be used to signify categories and highlight differences within scalar data. Luminance can also be used to directly indicate the Degree of Fulfilment (DOF) value for a single membership function. In the example in Fig. 2 (Right), the intensity represents the DOF value for a fuzzy function of the term Hot. The regions with the highest intensity indicate where the term Hot has its highest DOF value.

A number of other visual features can be utilised to indicate the precision of fuzzy data used in a visualisation, when applied to the visualisation objects:

- *size*-can be used to indicate the imprecision of data [12];
- *transparency*-can show the possibility of the fuzzy variable;
- *texture*-can indicate the level of precision, ambiguity or fuzziness;
- g*lyphs* and *icons*-for example data points with error bars indicate imprecision;
- *particles*-represent the fuzziness of a region by varying the space between them;
- *blurring*-can be used to show the indistinct nature of data points [7].

5 Higher Spatial Representations

The visual features listed above are usually spatially arranged to form a coherent display in graphic forms, which enable the perception of various patterns in the data. We can combine the use of such visual features for denoting the imprecision in data with a number of common representation methods employed to display spatial data in higher dimensions: 2D, 3D, parametric, dynamic, metaphors and multimedia sensors. These methods have already been discussed in [3] and are not repeated here. Instead, we now discuss ways by which we can improve the visualization method provide by [2] by using parallel coordinates. The authors used the thickness or grey intensity of

lines to indicate the fuzziness of points. One drawback is that it is difficult to visually distinguish fine grades of grey level on single lines. Another drawback is that it is not possible to perceive the core and support of a fuzzy set simultaneously.

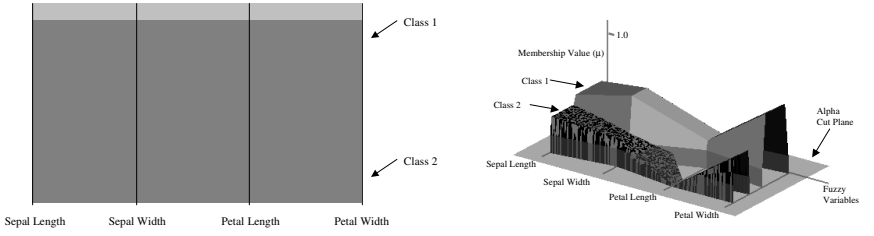

Fig. 3. (Left) Illustration of 2D method developed by Holve and Berthold, for representing multidimensional fuzzy rules using fuzzy parallel coordinates [2]. Two rules are illustrated from their Iris data example. (Right) Visualisation of the Iris data with lit and textured surfaces showing the same Iris data. Note how the alpha cutting of the membership function for Rule 2 on the Petal Length dimension is easily perceived.

One way of addressing these drawbacks is to use a 3D representation where the parallel coordinates are displayed on the x-y plane, and the fuzzy set membership functions are displayed in the z-coordinate (Fig. 3 Right). Different alpha-cuts of fuzzy rules can be identified by applying horizontal cutting planes. The separation of classes based on the confidence of a decision can be highlighted by using filled polygons with texture (Fig.3 Right) or using colour.

6 Profile Agent

This agent records a user's profile in various ways: patterns of tasks performed; patterns of usage of data and operations applied on fuzzy rules; specific types of constraints; desired degree of fulfilment of outcomes; and choice of visualization techniques. By communicating with both the user and the Control Agent, the Profile Agent uses these detected patterns to issue instructions to re-organise data and re-prioritise tasks. It also automatically offers choice of operations on fuzzy rules and degree of fulfilment of outcomes. Such adaptation would allow a user to gradually customize the visualization system to own application and subjective preferences.

7 Conclusion and Future Work

We have discussed the motivations behind the use of a multi-agent approach to develop a framework for visualization of fuzzy systems which is user- and task-oriented. This framework is based on our design of fundamental ontologies underlying the structure and requirements of these systems. We have also presented

the structure and activities of these agent classes and how they would be implemented. A case study has been investigated to provide visualization support for a fuzzy model which had been developed to predict electricity spot prices [3]. We are continuing to implement agent classes to extend fully their capabilities and to evaluate this framework using further case studies of fuzzy systems for different applications.

References

1. Berkan, R.C. and Trubatch, S.L.: Fuzzy System Design Principles, IEEE Press, NY (1997)
2. Berthold, M., and Holve, R.: Visualizing high dimensional fuzzy rules, in Proceedings of the 19th International Conference of the North American Fuzzy Information Processing Society: NAFIPS, Dept. of Electr. Eng. & Comput. Sci., California Univ., Berkeley, CA, USA (2000) 64–68
3. Brown, R. and Pham, B., Visualisation of Fuzzy Decision Support Information: A Case Study, IEEE International Conference on Fuzzy Systems, St Louis, USA, May (2003) in print
4. Cox, Z., Dickerson, J.A., and Cook, D.: Visualizing Membership in Multiple Clusters after Fuzzy C-means Clustering, in Proceedings of Visual Data Exploration and Analysis VIII (2001) 60–68.
5. Dickerson, J.A., Cox, Z., Wurtele, E.S., and Fulmer, A. W.: Creating metabolic and regulatory network models using fuzzy cognitive maps, in Proceedings of IFSA World Congress and 20th NAFIPS International Conference, Joint 9th, Dept. of Electr. Eng, Iowa State Univ., Ames, IA, USA, 4 (2001) 2171–2176.
6. Farwowski, W., and Mita, A. (Eds.) Applications of fuzzy set theory in human factors (1986)
7. Gershon, N. D.: Visualization of fuzzy data using generalized animation, Visualization '92, Proceedings, Mitre Corp., McLean, VA, USA, (1992) 268–273
8. Jiang, B.: Visualisation of Fuzzy Boundaries of Geographic Objects, Cartography: Journal of Mapping Sciences Institute, Australia, 27, (1998) 31–36
9. Keller, P., and Keller, M.: Visual Cues. Piscataway, USA: IEEE Press (1993)
10. Nurnberger, A., Klose, A., and Kruse, R.: Discussing cluster shapes of fuzzy classifiers, in Fuzzy Information Processing Society, 1999. NAFIPS. 18th International Conference of the North American, Fac. of Comput. Sci., Magdeburg Univ., Germany, (1999) 546–550.
11. Nurnberger, A., Klose, A., and Kruse, R.: Analyzing borders between partially contradicting fuzzy classification rules, in Fuzzy Information Processing Society, NAFIPS. 19th International Conference of the North American, Fac. of Comput. Sci., Magdeburg Univ., Germany, (2000) 59–63
12. Tufte, E.: The Visual Display of Quantitative Information. Cheshire, USA: Graphics Press (1983)
13. Zadeh, L. A.: Toward a Theory of Fuzzy Information Granulation and its Centrality in Human Reasoning and Fuzzy Logic, Fuzzy Sets and Systems, 90, 2, (1997) 111–127.