A KNOWLEDGE BASED FRAMEWORK FOR THE DESIGN OF SOFT-COMPUTING SYSTEMS

Satheesh Ramachandran — *Knowledge Based Systems, Inc.*
Madhav Erraguntla — *KBSI.*
Perakath Benjamin — *KBSI.*

Abstract: This paper presents a systematic framework for the design of intelligent decision support systems based upon soft computing paradigms like *neural networks*, *genetic algorithms*, *simulated annealing* and *fuzzy logic*. The approach applies knowledge based systems techniques to support development and application of models in these computing paradigms. The long-term goal of this research is to automate the design of soft-computing systems from a domain expert's description of the problem situation and a set of input data.

Key words: word processing

1. INTRODUCTION

This paper presents a systematic framework for the design of intelligent decision support systems based upon soft computing paradigms like *neural networks*, *genetic algorithms*, *simulated annealing* and *fuzzy logic*. The approach applies knowledge based systems techniques to support development and application of models in these computing paradigms. The long-term goal of this research is to automate the design of soft-computing systems from a domain expert's description of the problem situation and a set of input data. The research results that are presented in this paper address a near term goal in providing intelligent assistance and model design principles to decision makers regarding the use evolving computation paradigms in decision-making. The main product of this research effort is a demonstration version of a ***Generalized Modeling and Analysis Toolkit (GMAT)***. GMAT is being developed under a DoD funded research contract (KBSI, 98). Currently GMAT supports the knowledge-based application of neural network and fuzzy logic models (Surkan, 89; Zimmerman, 91; Zadeh, 93;

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35602-0_35

Wu, 94). This focus of this paper is the knowledge-based assistance provided in GMAT in the development of neural network and fuzzy logic models.

Soft computing paradigms such as artificial neural nets (ANN) and fuzzy set theoretic modeling techniques have been used with increasing success in recent years for a variety of different *commercial* and *military* applications (Martinez, 89). The realm of current application domains extend from pattern recognition with reconnaissance images for military intelligence to inventory forecasting for retail warehousing to fraud detection in the financial sector. Two key factors responsible for the recent increases in neural network and fuzzy logic applications are (i) advances in information technology (increased efficiency in the collection, storage and control of information) and (ii) significant progress in analytical and computational techniques (because of increased availability of tractable algorithms and software). These new generation information processing systems exhibit capabilities like adaptability, robustness, generalization, and the ability to work amidst the imprecision and uncertainty of the real world, making them attractive solution technologies for complex problems (Guiasa, 93). Nevertheless, the full potential offered by these powerful techniques have remained largely unharnessed because of the inherent difficulties in designing, calibrating and applying these models.

2. RESEARCH MOTIVATIONS AND CHALLENGES

Specifically, the effective use of soft computing techniques is greatly hindered because of the following reasons:

1) Technology inaccessibility: Soft computing techniques such as neural networks and fuzzy logic technology are still relatively inaccessible to the industry. Despite progress in more versatile algorithms (such as Radial Basis Functions, Wavenets, Fuzzy Associate Memories) and their applications to forecasting, consumer behavior models, pattern recognition, signal processing and control, etc., neural networks and fuzzy logic remains a highly academic subject with applications in a few progressive domains. In spite of the power and flexibility of these techniques and their potential applicability to numerous, common problems in the industry, the proliferation of these technologies into the industry remains limited. This is mainly because of the complexity in the theory of these techniques and the high degree of expertise required to model and apply them properly. Existing tools attempt to provide assistance to users that are experts in neural network and fuzzy logic by automating the low-level calculations involved in using these techniques. However, these tools provide little assistance to a non-expert user of these technologies. A user of the currently available tools is expected to possess a fair amount of expertise before he can start using these powerful techniques to solve his problems. There is a need for knowledge based tools that provide assistance to novice users, in each and every step involved in using these technologies – right from problem identification, to development of appropriate models, to model calibration, usage, and eventually in documenting and re-use of models and expertise. Unless such assistance is provided, neural networks, fuzzy logic, genetic algorithms and simulated annealing will remain mere interesting academic concepts, and used exclusively by a "privileged few" high technology users.

2) Absence of soft computing modeling methodologies: Currently, no structured methodologies are available for developing and using soft computing models. Even though the theory of neural networks and fuzzy logic are well grounded in sound scientific principles, their application remains an art. The process of developing neural network and fuzzy logic models is a highly skilled process. There is no single, unique model corresponding to a problem situation – very often developing appropriate neural network and fuzzy logic models involves intuition and guesses based on experience in order to arrive at the correct calibration (for example, formulating the best neural network model architecture often involves iterative determination of appropriate model structure, transfer functions, leaning algorithms, parameters, etc). The initial model is then refined based on experiments. Skill is required not only in developing sound models, but also in ensuring adequate levels of model calibration (for example, over-fitting in a neural network might result in blind replication of patterns), and specifying appropriate values for the calibration parameters (such as neural network learning and momentum rates). This expertise in neural network and fuzzy logic can only be obtained through years of hands-on experience in applying these techniques to solve practical problems. The long learning curves associated with them render these powerful technologies unattractive for solving the real world, immediate problems of the industry. Availability of a structured methodology to support the process of designing, training, interpreting, and reusing neural network and fuzzy logic models would decrease the reliance of neural network and fuzzy logic experts, resulting in an increase in the use and popularity of these technologies. The timing appears right for the design and insertion of such methods; a method that is an encapsulation of "best practice".

3) Lack of design rationale capture and re-use mechanisms: As stated previously, developing and using soft computing models is a knowledge intensive, iterative process. Developing the suitable calibrated system for a given problem situation involves a significant amount of experimentation. This process involves conducting a set of experiments by making experience-based guesses about the architecture features that will be appropriate to the focus problem situation. Most often, the performance of these initial experiments will not be satisfactory. However, this experimentation process serves a very important function – the insights gained, observations made regarding the design concepts that seem to work and (often more importantly) those design concepts that do not seem to work – help in designing better neural networks in the next stage. This process of experimentation, deduction and incremental refinement is repeated until a model with acceptable performance characteristics is obtained. The experimentation process, the design *rationale*, and the knowledge gained during the entire design process are important knowledge assets that must be captured, organized, and maintained. For example, the expertise gained by neural network design and experimentation expertise is an important source of information for ideal architectures (type and number of neurons, number of hidden layers, etc.), network parameters (learning rate, momentum rates and their step sizes, etc.), training conditions (number of epochs, extent of cross-validation, etc.), input factors and variables, metrics (mean square error, absolute error, maximum error, etc.), and learning algorithms. It is important to capture such design rationale and experimentation information in a form that is re-usable. Contemporary soft computing modeling tools provide adequate support for representing and browsing the final design of the models. However, no support is provided by existing tools for capturing and organizing the vast knowledge involved in the development, incremental refinement, and introspection phases of modeling. Lack of proper methodologies to capture and re-use the knowledge involved in the design and development of soft computing models makes it difficult to exploit the results of developing one model in other subsequent design endeavors. Every model development becomes a new, unfamiliar activity to be explored and performed from scratch, without the assistance of the previous model design rationales and insights.

The GMAT research initiative targets these technological and pragmatic barriers directly. The research establishes the technical viability of a knowledge-based approach for the effective application of neural nets and fuzzy logic (these two techniques are the focus of our research) to a variety of decision problems. The following sections outline the technical approach adopted in GMAT.

3. FRAMEWORK FOR INTELLIGENT SYSTEM DESIGN

The overall solution is an end-to-end support environment that captures and delivers knowledge and experience to allow domain experts and synthetic algorithm engineers to produce workable first time applications. GMAT is designed to leverage and compliment existing COTS tools for fuzzy logic and neural networks. GMAT distinguishes itself from COTS packages by automating the entire life cycle of design and development for rapidly applying fuzzy logic and neural networks to real world problem situations. This section presents an example application to illustrate GMAT concept of operation. This commercial endeavor involves application of neural network and genetic algorithms technology to cash management solutions in the commercial banking sector. The results are currently being deployed to forecast the movement of money in over 2000 branches and 6000 ATMs nationwide.

3.1 Bank Cash Management Application:

We employed the GMAT system to develop a cash management application for the banking industry. The Department of Treasury prescribes norms for the cash level that must be maintained by each bank to support reserve requirements. As a result of recent regulation changes, banks currently have more leeway as to the levels of cash reserves they must maintain. Banks are attempting to minimize the cash they carry to the new reserve levels and yet meet the customer requirements. The less cash they carry for day-to-day operations, the more they will have for long term investments, resulting in more profits without increased fees. A critical piece of information required for solving this optimization problem is an accurate forecast of customers' daily withdrawals and deposits, so that banks can keep a minimal amount of cash required to support the forecasted level of activity. We used GMAT to develop neural network based predictive (causal and time-series based) models to forecast withdrawals and deposits for branches and ATMs. Although the application seemingly falls under the purview of a traditional MRP-type forecasting (followed by optimization) problem, many pitfalls were encountered. The data in many cases were insufficient, often spurious, and most critically, the amount of time that was required for building forecasting models for each of the over 2000 branches/ATM's were prohibitive in terms of time and effort. Firstly, the GMAT data management functions for data verification, repair and transformation were useful in reducing the upfront data preparatory time, and the consistency and quality of the prepared data had an impact on the ultimate performance of the forecasting models. The support provided by GMAT in integrating the various soft-computing paradigms led to significant reduction in the 2000+ model calibrations. One such innovation involved the application of cluster analysis to identify groupings in ATM's and branches. The idea here is that ATM's and branches that exhibited statistically similar cash withdrawal and deposit patterns could be served by the same forecasting models. This enabled us to reduce the number of forecasting models that were built. Once the cash requirements are forecasted, the GMAT tool also provided the capability to develop optimal cash delivering schedules (ordering policies) based upon a genetic algorithm based optimizing capability. The forecasts that are supplied by the neural network models are ultimately used by a genetic algorithm based optimizer to determine the optimal cash ordering and transportation policies.

3.2 Structured Methodology

The GMAT methodology provides guidelines for the development process of intelligent information processing models. The supporting GMAT toolkit is intended to support the domain user in every step of the problem-solving life cycle. The systematic model development process that is supported by GMAT is described in Sections 4.1 through 4.7:

4. GMAT FUNCTIONAL DESCRIPTION

In this section, we describe the functional description of the GMAT toolkit.

4.1 Problem Domain and Goal Specification

In GMAT, the highest-level definition is a *project*, which corresponds to a subset of an enterprise that has one or more problem situations and a set of enterprise variables that are used in generating models to solve the problem situations. The GMAT tool helps with the definition of decision-making goals that are associated with each of the problem situations. The goals are used to focus and direct the knowledge discovery process. In its current implementation, the GMAT goal modeler allows for the (a) selection goals from the Goal Template Library, (b) addition of new goals, (c) goal editing, and (d) organization of goals in a hierarchy. The rationale for goal capture is the observation that decision-making endeavors are goal-directed. The goal definition capability thus provides the preliminary context needed for decision-making using knowledge discovery.

4.2 Data Acquisition, Assessment, Verification and Validation

An early step in the GMAT methodology involves capturing domain knowledge in terms of the variables in the domain and their data sources. Data source refers to the location in a database or text file where the values for that variable are present. GMAT supports data retrieval from relational databases and comma-delimited text files. There are three types of basic variable categorization that is supported by GMAT. A variable can be either crisp or fuzzy. Crisp variables have numeric or text values. Fuzzy variables can be specified in terms of their fuzzy classes, and membership functions can be specified for these classes. Variables are also classified as basic or derived, based on whether the data is directly available or some transformation needs to be performed. Finally variables can be defined as either independent or dependent. Independent variables are analogous to key fields in databases. Once identified, raw data must next be verified, validated, repaired, and transformed before it can be applied to solve domain problems. By supporting these data management functions independent of the problem goals, GMAT allows for reuse of validated and cleansed data across different modeling goals.

Among the vexing problems in data intensive applications like neural networks and fuzzy logic based systems is the error in input data. Errors in the input data could result from mis-specification, omission, corruption, approximation, etc. In the calibration of soft-computing systems, even a few incorrect data points could drastically change the intended pattern that these systems are expected to learn, resulting in poor model performance. In other situations, the data point might be correct, but the value it represents might be an extremely rare phenomenon that skews the underlying pattern. In such situations, it might be necessary to remove that data point or modify it appropriately so that the model learns the desired pattern and not some isolated deviation in the input data. GMAT provides automated as well as user-assisted support for detection of possible errors or anomalies in the data. GMAT supports the following methods for detecting errors in the input data:

- Visual Inspection—in this method the data points associated with variable are displayed in different ways as a graph or plot (2D or 3D visualizations). GMAT has a built-in data browser that retrieves data from the data source and displays it for the user. As a result of this feature, the user requires minimal knowledge in database management systems
- Detection of standard anomalies—Users can specify standard anomalies such as negative values, zero values, or null values where

such conditions are not expected. Also, the user can get a list of data points that are outside the 2σ and 3σ limits.

- User-defined Anomalies—In addition to standard anomalies, users can specify mathematical relationships/constraints that need to be maintained between different variables. For example, in the banking application, the user can check for inconsistencies to the following constraint: *End Cash = Begin cash – Withdrawals + Deposits*. GMAT provides an SQL builder to aid the user in defining and validating such relationships.

4.2.1 Data Repair

GMAT allows for automatically fixing suspected anomalous data points. The repair strategies supported in GMAT include the replacement of data points with different approximation strategies such as average within a range, average based on a condition, median, interpolation, etc.

4.2.2 Data Transformation

In most industrial and military applications, there is a large mismatch between the available data and data that is required as input to a neural network. For example, temporal data may be clustered better if the clustering algorithm was done on the frequency series rather than the time series. As another example, consider an image processing application, where the input is a pixel grid representing a reconnaissance satellite picture. Before this input can be analyzed using neural networks for patterns or classification, data reduction transformations are needed to reduce the huge input matrix (on the order of 600x800) to a more manageable size. Existing COTS packages for soft computing methods require the input data to be externally pre-processed and converted to the appropriate input format. Such pre-processing and conversion is supported by GMAT in the form of a repository of data transformation utilities. GMAT supports the following transformations: 1) Fuzzy Transformations, and Qualitative Encoding, 2) Frequency Transformations (Fourier Transformations), and 3) Mathematical Transformations.

More recent hybridized neuro-fuzzy models have the ability to include qualitative factors in the analysis in addition to quantitative data. For pure mathematical and statistical analysis of input data, there are numerous analytical methods that perform the job as efficiently as neural networks. However, these neuro-fuzzy hybrid modeling paradigms excel in comparison to traditional techniques in their ability to include highly subjective, qualitative factors in the analysis. These qualitative factors represent experience-based, empirical associations developed by human experts gained during years of working in the domain. These associations are often too fuzzy and qualitative to be accounted for in an analytical model. At the same time, they are an important part of the domain knowledge and have to be captured and exploited in order to obtain sound analysis results. For example, in the banking application, it was known that cash demand is "low" on Mondays, "high" on Fridays, and "medium" on other days of the week. GMAT provides support for incorporation of such qualitative, human inputs into the analysis. During training, GMAT provides assistance to refine these qualitative associations based on the patterns observed in the input data.

4.3 Model Selection

Once domain knowledge for a project has been captured in terms of project variables that are valid, verified, transformed, and scrubbed, they can be used across different problem situations. In GMAT, the problem situation definition includes specification of the problem focus name and the problem focus type. Currently, users can select the following problem focus types: clustering, discrimination, forecasting, qualitative analysis (fuzzy operators, fuzzy neural networks, fuzzy associative memories) and causal models (neural networks based curve-fitting). Once a problem situation type has been specified, GMAT provides knowledge-based support for selection of appropriate solution technology type. For example, if the user specifies the problem situation as being of the *clustering* type, she can choose either the *k-mean* algorithm or the unsupervised Kohonen Neural Network as the technology solution. Additionally, GMAT has built-in templates for standard technology solution types. The knowledge of pairing a problem situation type to a solution technology type is implemented as a knowledge base. Users can specify alternative solution designs for a given problem situation. Each solution design uses a specific technology type.

4.4 Model Design and Configuration

In this section, we describe the assistance that GMAT provides for neural network model design (the support for other paradigms follow similar principles). For neural network based technology, generating the best neural network is by no means a simple task. This is because there are various choices of neural network architecture design, input and output variable combinations, and exemplar constraints. As a result, it is possible to create a large number of alternative neural network designs (NND) and training sets (TS). The final neural network model must be selected from among the various NND-TS pairs. However, for *each* NND-TS pair, a substantial amount of time and effort must be spent on experimentation and simulation (Section 4.5). Unless automated support is provided, neural network model generation is an extremely tedious and time-consuming task. GMAT supports this activity by automatically generating neural network models from the design specifications, and by preparing the input and desired output training sets. The following four design criteria give rise to different NND-TS pairs.

1. **Neural Network Architecture:** A typical neural network architecture design involves parameters such as (i) Number of processing elements at the input/output level, (ii) Input level transformation functions, (iii) Output level transformation functions, (iv) Number of Hidden layers, (iv) Number of axons for each hidden layer, (v) Hidden layer transformation functions, (vi) Number of axons for each hidden layer, (vii) Learning Algorithm and type.
2. **Input combination.** This is the list and order of project-level variables that form inputs to the neural network. The number of processing elements in the input layer is determined by the input combination.
3. **Output combination.** This is the list and order of project-level variables that form output to the neural network. The number of processing elements in the output layer is determined by the output combination.
4. **Exemplar constraint.** This is a set of restrictions for deriving a training set. For example, the user may want to avoid all training sets that have negative values.

While the combinations of the former three design criteria give rise to the different NNDs, the combinations of the latter three design criteria give rise to the different TSs.

4.5 Neural Network Calibration (or Training) Management

Developing a good neural network model for a given NND-TS pair is itself an iterative, experiment-based, incremental refinement process. After training and analyzing the performance of a neural network model, improvements are made. This process of training and incremental refinement is continued until a neural network with acceptable performance is obtained. Existing neural network tools provide little support for this iterative training and refinement process. Factors that determine the result of training of a NND with a TS include the (i) learning rate, (ii) momentum rate, (iii) the step size, (iv) maximum number of epochs, (v) type and extent of cross validation, and the (vi) termination criterion for training (number of epochs versus desired error level).

GMAT supports various experimental designs based on combinations of the above variables. For each experiment, GMAT invokes the neural network simulator to train each NND-TS pair, calculate the mean percent error, and save it to the GMAT database. As an alternative, the user can specify an experiment to be under the control of the *Neural Network Controller* that can monitor the performance of a neural network in real-time while the neural network is being trained. In order to optimize the performance of the neural network, the controller modifies the learning rate and the momentum rate in real-time. This controller itself is implemented using fuzzy logic and neural network technologies.

4.6 Training Result Browser and Execution Control

GMAT displays the results of the neural network training for each experiment. The user can browse through the various executions and determine the performance of the training of each NND with any training set. The user can also execute the best neural network or any given neural network with new data that is different from the training set. Each new execution is stored in the GMAT database to facilitate reuse of results.

4.7 Design Rationale Capture

GMAT provides extensive support for capture of design rationale for each experiment and a visual trace of all experiments relating to a problem situation. All aspects of a problem situation, problem solution, experiment design, neural network training, and execution are stored in the GMAT database for future reference. Using the design rationale capture mechanism and the trace, the user can quickly understand the experiments that were conducted. Description fields provide for the capture of notes for a given problem situation, experiment or training, where as name of experimenter/modeler and date created/executed provide for configuration management.

5. GMAT ARCHITECTURE

Due to page limitation GMAT architecture is not included in the final version. Please contact the authors for the complete version of the paper.

Acknowledgements

GMAT application was designed and developed under funding from Army SBIR Phase I and Phase II funding (KBSI, 98). We acknowledge the support of Army SBIR program and Mr. Kachesh Pathak in particular in our endeavors.

References

1. Davis, L. (Editor). (1991). *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, NY.
2. Dubois, Didier and Prade, H. (1980). *Fuzzy Sets and Systems: Theory and Applications*, Academic Press, New York.
3. Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.
4. Graham, A. (1997). Seed Money Available to Develop and Prototype National Space Reconnaissance Data Capabilities – How MERIT Can Help Your Program. *PM*, May-June.
5. Guiasa, S. (1993). A Unitary Treatment of Several Known Measures of Uncertainty Induced by Probability, Possibility, Fuzziness, Plausibility, and Belief, *Uncertainty in Intelligent Systems*. Amsterdam: North Holland Press, 355-366.
6. Knowledge Based Systems, Inc. (1998). *"Generalized Event Representation Modeling and Analysis Tool (GERMAT)*, Army Phase II SBIR Contract No. DASG60-98-C-0051.
7. Kosko, B. (1992). *Neural Networks and Fuzzy Systems: A Dynamical Systems Approach for Machine Intelligence*. New York: Prentice Hall.
8. Lin, C.T. & George, C.S. (1991). Neural Network Based Fuzzy Logic Control System. *IEEE Transactions on Computers*, 1320-1325.
9. Martinez, Tony. (1989). Neural Networks Applicability: Classifying the Input Space. *Proceeding of the Fifth IASTED International Symposium.*
10. Rumelhart D. E., et al. (1987). *Parallel Distributed Processing*, MIT Press, Cambridge, MA, 1987.
11. Surkan, A. & Cao, H. (1989). Models of Artificial Neural Networks: An Emerging Learning Technology. *Proceeding of the Fifth IASTED International Symposium.*
12. Wu, J.K. (1994). *Neural Networks and Simulation Methods.*
13. Zadeh, L. (1993). Soft Computing and Fuzzy Logic. *IEEE Software*, 48-56.
14. Zimmerman, H. J. (1991). *Fuzzy Set Theory*, Kluwer, Boston, MA.