A METHODOLOGY FOR HUMAN BEHAVIOR MODELING IN COMPUTER GENERATED FORCES

Michael J. Johnson
Department of Mathematical Sciences
United States Military Academy
West Point, NY 10996, U.S.A.

Mansooreh Mollaghasemi
Department of Industrial Engineering and Management Sciences
University of Central Florida
Orlando, FL 32816, U.S.A.

Michael Georgiopoulos
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816, U.S.A.

Michael McGinnis
Department of Systems Engineering
United States Military Academy
West Point, NY 10996, U.S.A.

Abstract

One way to improve how simulations can more realistically model human behavior is through use of intelligent computer generated objects (ICGOs). This paper proposes a framework for incorporating intelligent objects using fuzzy ARTMAP neural networks, into existing simulations in a way that significantly reduces the need for updating and maintaining the behavioral aspects of the code. The framework enables the ICGO to learn from existing computer generated object rules, as well as learn from human operators participating in the simulation. The final product will be an ICGO that exhibits certain human behavioral traits including the ability to adapt to changes and basing decisions on previous experiences. The methodology is demonstrated in real-time using OneSAF Testbed developed by the Simulation, Training and Instrumentation Command (STRICOM).

Keywords

Artificial neural networks, artificial intelligence, simulation, computer generated objects, computer generated forces.

1. Introduction

Computer generated objects play a crucial role in a many of today's simulations. Regardless of the application, or the role of the CGO, they are expected to behave like the real-world entities they represent. This research primarily focuses on the role of the CGO in military simulations. However, research results and findings presented throughout this paper, as well as many of the applications, can be generalized for use in non-military simulations through a straightforward extension of the methodology presented here. For instance, in the entertainment software industry a myriad of simulation games can apply this methodology so the enemy characters learn from the on-going simulation. Similarly, emergency service agencies use simulations to train for critical scenarios. Invoking this methodology can provide a more realistic virtual population. A key technology that promises to provide a near-term solution to the problems of behavior representation is artificial intelligence (AI). One AI technique, artificial neural networks (ANN), have been shown to be very well suited for problems characterized by

a rich set of data but where information regarding the intricacies of how the system works is not well known. ANNs are especially useful when a clear mathematical formula of system behavior that describes the behavior of the system cannot be derived. That is to say, an explicit methodology cannot be gleaned from the domain [1]. This accurately describes most situations involving CGOs. Using data created from a well-defined logic structure governing a decision episode, it is possible to train an ANN (representing a CGO) to make a decision based on a current set of relevant parameters that represent the state of the simulated environment. Evaluating the outcome of a decision, and providing the model with alternative outcome provides a means for the CGO to learn from its previous experiences. The result is an intelligent CGO that does not require extensive updates to existing code in order to achieve behavior in the object that mimics behavior of the real-world entity. The object's behavior self-modifies as it updates itself in response to a changing synthetic environment. This approach can also be adapted to the changing behavior of its human opponent, as well as, changing behaviors of other semi-automated forces. Another very desirable and inherent characteristic of this ICGO is its potential to model human-like decision making.

2. Fuzzy ARTMAP

Fuzzy ARTMAP is an ANN for incremental, supervised learning and prediction of binary inputs as well as analog inputs defined by a fuzzy set of features. During supervised learning, fuzzy ARTMAP receives a stream of input patterns and an associated stream of output classification categories. The algorithm creates a certain number of recognition categories required to satisfy an established level of accuracy. Once trained, fuzzy ARTMAP can accept additional input vectors and efficiently classify them. It can also be incrementally trained using newly acquired data. Fuzzy ARTMAP operates with only a few parameters. The baseline vigilance parameter $r \in [0,1]$ calibrates the minimum amount to correct a predictive error. A choice parameter $b > 0$, affects the dynamics involving which recognition category is selected first. A learning rate parameter $a \in [0,1]$ can be implemented if the application requires other than *fast learning*. More intricate details of fuzzy ARTMAP are discussed in [2].

This research methodology uses a fuzzy ARTMAP (FAM) neural network to create an intelligent computer generated object. This type of network has many desirable characteristics that allow modelers to incorporate an intelligent CGO into simulations. These benefits include:

1. On-line (incremental) learning
2. Automatic creation of an appropriate sized network
3. Minimal parameter selection
4. Quick convergence
5. Accepts binary and analog data
6. Easily explainable results.

3. Methodology

Five tasks comprise the methodology. A key step is to *define the decision episode*. Without fully understanding the process, a modeler will have difficulty implementing any methodology. The next step is to *create the model* from logic governing the decision episode. The result is a FAM neural network that is *implemented into the simulation*. Once the intelligent CGO has been successfully implemented, the modeler must conduct certain *verification and validation* (V&V) procedures to ensure the model is performing according to its intended purpose, and the actions of the ICGO are believable and credible. The final task, *maintenance*, ensures process efficiency and accuracy within the simulation. These five tasks are discussed in more detail below.

3.1 Task 1: Define the decision episode

Obtaining accurate and complete information regarding the task to be performed is the first and most critical task in order to construct a model that exhibits human (or special) behavior [3]. For a particular decision to be realistically modeled, it must be described in detail, including all pertinent parameters that effect the decision, and the range of values for these parameters. All possible outcomes must be identified as well. In many cases a weighting scheme is used to place proper emphasis on certain parameters. It is necessary to identify a method of decision evaluation to determine if the "correct" action was chosen. It is also critical to understand how quickly the learning process should occur. For instance, should a CGO alter its behavior based upon one instance? Or, should

learning take place more slowly? In most cases, the simulation may require several *confirming* occurrences before it is willing to alter its behavior or select an alternate action.

Some decisions are certainly more complex than others. When defining the decision it is important to determine the total number of possible outcomes. In a simple case, there may be only two outcomes. Binary decisions pose a much simpler problem for the network during the simulation. However, if there are several possible outcomes, it can become rather complicated to identify the "next best alternative".

At some point in time following the decision episode, each decision that was selected will be evaluated and determined to be correct or incorrect based on a favorable or unfavorable outcome for the CGO. Therefore it is necessary to identify the decision evaluation criteria. It may be very obvious and easy to identify, or it may be a combination of factors and less clear. Since the learning process hinges upon evaluation of the decision it is very important to clearly define and quantify this measure. Key elements of information or subtasks are summarized below. This list assumes that the decision process exists in the simulation and does not need to be developed from the ground up.

- Ensure the decision episode requires an ICGO
- Identify algorithm parameters (inputs)
- Determine parameter values and ranges
- Determine parameter weighting scheme (if required)
- Identify decision outcomes (outputs)
- Establish learning rate
- Determine evaluation criteria

An interdisciplinary team of computer scientists and subject matter experts in the fields associated with the simulation should be consulted to define or examine the process, identify the critical parameters and validate the decision logic. This measure is very important regardless of the simulation application. Depending on the application, sociologists, psychologists, and military scientists may also be important members of the interdisciplinary team [3]. Properly and completely defining the decision process can ultimately reduce the requirement in the simulation for certain human-in-the-loop interactions with the CGO, thus providing a tremendous benefit to the end user of the simulation.

3.2. Task 2: Create the Model

In this task the FAM neural network model is created off-line, prior to simulation run-time. The objective of this task is to ensure the artificial neural network acquires, and adequately uses information that was gathered during first task. Figure 1 contains a flow chart that describes this task.

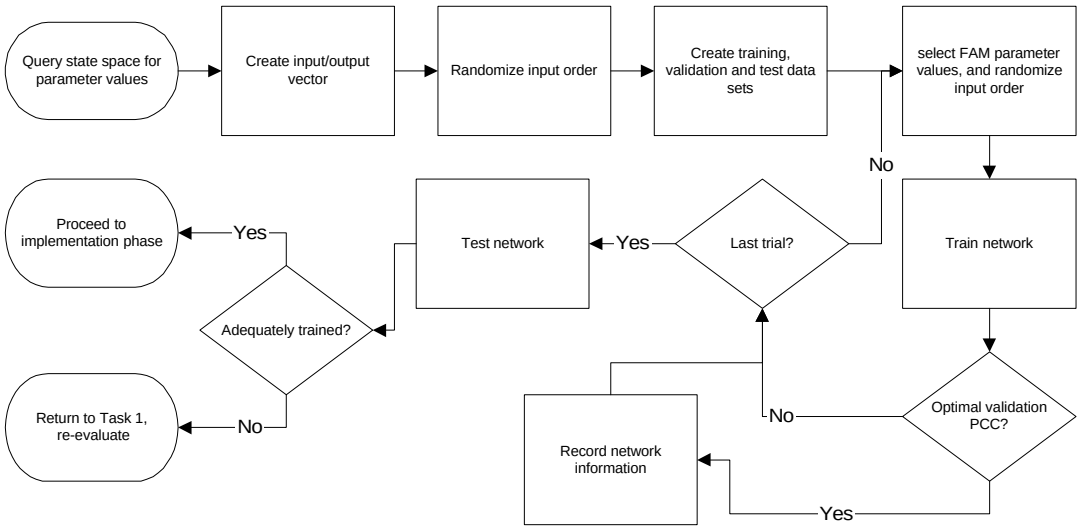

Figure 1. Required steps to create fuzzy ARTMAP neural network model

If the simulation currently has a set of if-then-else rules, or some other logic structure for the decision episode that is well defined, and underlying logic has been validated, then this information can be extracted from the simulation computer code and used to create the data required to train the FAM network. Even so, computer code and logic should be scrutinized and reevaluated to ensure that information is complete, accurate and valid. On the other hand, if this decision structure is not present at all, or not adequately modeled in the simulation, then it must be developed before proceeding any further. Using the decision rules and logic structure, the modeler creates a set of training, validation and testing input data along with corresponding outputs. In many cases this is not a trivial task and requires some detailed effort.

The algorithm trains the network and produces a *psuedo-optimal* set of connection weights in the following manner. It selects the parameter values and input pattern order by using a validation set of data that provides the highest percent correct classification (PCC). The choice parameter b, and the vigilance parameter r, are set to values between 0 and 1 in 0.1 size increments. Several (10 to 15) randomly ordered training data sets are used for each pair of parameter values. Results from each trial are compared using separate validation data. The set of weights that produce the highest percent correct classification (PCC) are retained. If the user determines that the testing phase produces an acceptable level of performance, then this task is complete. However, if the results from the test data are not acceptable, it is necessary to return to Task 1 to re-evaluate the decision logic, relevant parameters, and ranges of those parameters to determine why the network is not producing acceptable results.

Results obtained using Fuzzy ARTMAP are highly dependent upon the data set. Therefore, an automated process is used so the user does not need to be an expert regarding the algorithm, and is not required to select any algorithm parameter values. Results for several benchmark databases have produced extremely good results. It was shown that with the automated process, the PCC for the test data was significantly higher than attempting to manually selecting b and r. It is also much less time consuming. The benchmark databases also showed a positive correlation exists between the validation PCC and test PCC. Therefore, optimizing the validation PCC, in general, produces a more desirable PCC for the test data. At the completion of task 2, the modeler has created the FAM neural network. A text file contains all required information regarding the size and structure of the network as well as the selected parameter values that will be used in the next step of the methodology.

3.3. Task 3: Implement the model

In this task the newly created ANN is used within the simulation to serve as the decision tool for the semi-automated force for the modeled decision episode. The previously existing rule-based decision process is no longer used, and is removed from the simulation code. During a simulation run when the CGO is confronted with a decision episode, the algorithm is invoked and, operating in real-time, the ANN provides the required decision. The decision logic for FAM neural network is diagramed in Figure 3. When the decision episode occurs, relevant input parameters for the network are queried to determine their current values. An input vector is created and presented to the performance phase of fuzzy ARTMAP. Two situations could occur at this time. A small chance exists that the network returns an "I don't know" response because an output pattern can not be found to correspond with the provided input pattern. This situation can be avoided almost entirely depending on the selection of the choice parameter, and if adequate training occurred during the information acquisition phase of the process. However, if it were to occur, then the choice parameter is reduced to zero, and the instance would be flagged as a "weak/new response". Reducing the choice parameter b, would force the network to associate the input vector with a previously established output category and guarantee a response from the network. The predominately more common situation is when the network provides a "known response" and passes the appropriate action to the CGO entity that responds accordingly in the simulation.

The outcome of this decision is later evaluated to determine if a *correct* decision was made. That is, to see if the outcome of the decision is favorable, or unfavorable from the perspective of the CGO. Again, there are two cases to consider. If the decision is determined to be the correct one, then the input/output pattern is presented to the training phase of fuzzy ARTMAP in order to strengthen the current set of weights within the network. However, if the decision is determined to be incorrect, then the algorithm checks "next best alternative". The network creates a list of feasible decisions. All alternatives that satisfy the vigilance criterion are prioritized based on the bottom-up weights from the FAM network. Fuzzy ARTMAP may produce only one feasible outcome, or it may list one or

more alternatives. If an alternate decision exists, then the decision is considered *weak*. This input vector with its new output is used to retrain the network. Therefore, when presented with this decision episode in the future, under the same circumstances, if the outcome was undesirable and the decision was weak, the CGO will be more prone to select the next best alternative decision. However, if no such feasible alternative exists, the decision is labeled

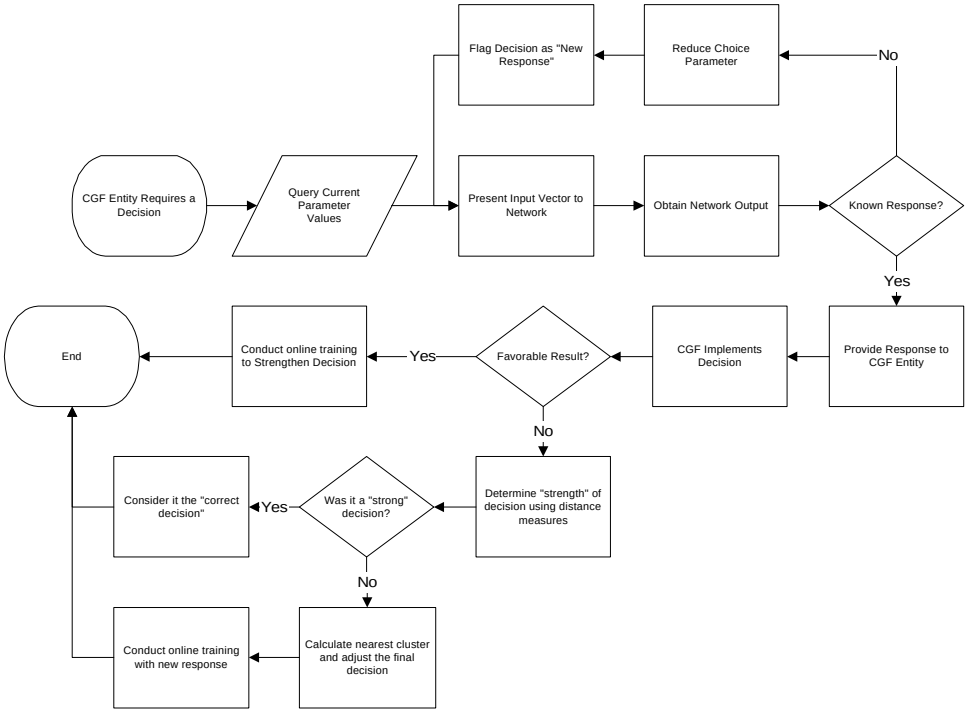

Figure 2. Decision logic for implemented fuzzy ARTMAP neural network model

strong. In this case the outcome for the decision remains unchanged in the future, even though the outcome of the decision was determined to be unfavorable. Note in this case however, the decision is not reinforced. That is to say, the input/output pair is not presented to the network for online, incremental training. The *learning* for the CGO is a result of the incremental training of the ANN based on the outcome of each decision episode. As a result, successful decisions are reinforced, and unsuccessful decisions are examined to determine if there are any feasible alternatives. If there is a logical alternate solution, measures are taken to give the network an opportunity to avoid making the same mistake given the same circumstances.

3.4. Task 4: Verify and validate the model

Each application will have its own set of verification and validation criteria depending on the simulation, the decision being modeled, as well as the purpose and intended use of the simulation. Fuzzy ARTMAP provides a relatively straightforward explanation on why a particular classification occurred [4]. In many cases it is more important to know why a particular decision was made than the actual decision itself. A text file can easily be created in most simulations to record key information in each pertinent step. Weight changes can be traced to the input/output pair which enables the modeler to ensure that the model is producing sensible results. The algorithm makes it relatively straightforward to replay identical scenarios and examine the results off-line to determine if the network is producing similar results as the pre-existing rule-based decision tool. The learning process can also be examined to check it against current doctrine (or common practices depending on the application and simulation). Most importantly, this method can be used to observe the learning as it occurs in a controlled testing environment, void of any outside compounding influences that are often present in a complicated simulation program. Virtually every aspect of the algorithm can be monitored; effective learning rate, magnitude of weight changes, percent of correct classifications, as well as occurrences of "I don't know" responses. All of this retrievable information assists in the verification and validation procedures.

3.5. Task 5: Maintain the model

This task focuses on maintaining model accuracy and efficiency. Periodic checks ensure the decision process has not evolved too radically. It is possible in some cases, due to long periods of on-line learning that the algorithm *decays*, or does not generalize well for the *original* data used to train and test the algorithm. To avoid this, periodic retraining with the full data set will restore the network. In most cases, this retraining takes about 15 minutes and can be conducted off-line. During on-line training new nodes may be created in the network. These nodes can occur as the network grows to accommodate new situations that do not generalize well to the previous output values. Periodic maintenance may be required to "prune" the network of obsolete nodes that no longer are required by the ICGO. Although the algorithm operates extremely fast, and all indications show it will not be a computational strain during simulation run-time, it makes good sense to maintain maximum efficiency. By simple inspection, the obsolete nodes can be identified and safely removed from the network structure using various pruning techniques [5].

4. PRELIMINARY RESULTS

This methodology has been partially demonstrated using OneSAF Testbed Baseline (OTB), open architecture model for semi-automated forces developed by the U.S. Army, Simulation, Training, and Instrumentation Command (STRICOM). A relatively simple decision episode was modeled. The decision involves selecting the correct weapon system and munition for a CGO (in this case an M1 Abrahms tank) when confronted with an enemy target. Admittedly, this is a rather simple decision for the tank in the simulation and other, more complicated decisions will be modeled as well. However, the choice to model this decision episode was based on several factors. It is a decision that could benefit from an intelligence decision tool. That is to say, if the tank is repeatedly unsuccessful with its primary selection of weapon and munition against a particular target, it may benefit from selecting differently in the future. Also, the decision logic currently present in OTB for this decision episode is well documented, validated, and rather straightforward to extract. Additionally, since it is relatively confined to just a few functions within the OTB computer code, the ANN decision tool is will be relatively straightforward to implement.

The algorithm has shown to be very efficient. For the types of decisions normally considered for this methodology produces networks with anywhere from 8 to 30 committed nodes. For the examples in our preliminary research, the computer time required to conduct training, validation and testing in this manner generally takes less than 5-15 minutes using a 400 megahertz Pentium II personal computer. Other, larger databases such as the Pima Indian diabetes database [6] have taken as much as a 45 minutes. Once implemented into the simulation, the CPU time required for the performance phase requires much less than 0.25 seconds. We are poised to model other decision episodes in OTB using this methodology. Some that appear to be most suitable and appropriate include: target selection for direct fire weapon systems, identification of friend or foe and possibly some platoon command decisions regarding decisions whether or not to attack, defend or retreat. The OTB software development team is also hoping to use this methodology in to aid in developing tactics and doctrine for future combat systems for the Army.

5. CONCLUSIONS

This research provides modelers a methodology to implement an intelligent CGO into an existing simulation environment. It provides a means to incorporate more realistic CGOs that exhibit a more reasonable representation of the human decision making process. The algorithm can operate in a real-time simulation environment, and does not add a significant burden to either programming or computational requirements to the simulation. Finally, the methodology reduces the overall effort, in terms of man hours, lines of computer code and money to implement and maintain an effective CGO.

References

1. Mall, H., & Ourston, D., 1996, "Neural Networks and Bounded Neural Nets, Command Decision Modeling Technology Assessment," Compiled by the US Army Artificial Intelligence Center for the National Simulation Center, Fort Leavenworth, KS, TAA 5-1 to TAA 5-13.
2. Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., & Rosen, D. B., 1992, "Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps,"

appears *in IEEE Transactions on Neural Networks*, Vol. 3, No. 5, September 1992.
3. Pew, R. W., & Mavor, A. S. (Eds.), 1998, "Modeling Human and Organizational Behavior: Application to Military Simulations," Washington, DC: National Academy Press, 323-328.
4. Carpenter, G. A., & Tan, A., 1995, "Rule extraction: from neural architecture to symbolic representation," appears in Connection Science, Vol. 7, No. 1, 3-26.
5. Reed, R., 1993, "Pruning algorithms," appears in IEEE Transactions on NNs, Vol. 4, No. 5, 740-747.
6. Sigillito, V., 1990, "*Pima Indians Diabetes Database"*, Applied Physics laboratory, The Johns Hopkins University, laurel, MD., 9 May 1990. Original database owners: National Institute of Diabetes and Digestive Kidney Diseases.