UMAS Learning Requirement for Controlling Network Resources

Abdullah Gani[1], N.Abouzakhar [2], Gordon Manson [3]
Centre for Mobile Communication Research (C4MCR)
Dept of Computer Science
The University of Sheffield
Regent Court, 211 Portobello St, Sheffield S1 4DP
United Kingdom

Abstract - This paper presents an intelligent User Manager Agent System (UMAS) in which it has a capability of making a management decision for balancing the network load with the users' requests for accessing network resources. A conventional users' request for accessing network resources depends on rigid rules setting and it can affect the overall performance of network services. UMAS provides additional measure in controlling the network services availability and responsiveness. In providing a different level of services to users, UMAS is required to perform an appropriate learning activity that can furnish an input for a decision of time allocation for a single session. This paper demonstrates the use of Neuro Fuzzy Logic for performing the learning that will be integrated into the UMAS.

Key words: Intelligent Agent, Knowledge Engineering, Neuro Fuzzy Logic

1 Introduction

The term of learning was originally used to describe the process of human learning and is always being associated with capturing knowledge through understanding and applying it for changing the behaviour towards desirable outcomes. Pavlov and Skinner's theory on learning that based on experiments were conducted on animal in order to understand animal learns. However, it is believed that a sequential type-operating machine of computer can be 'taught' like in Pavlov's experiment where dogs were used as a subject. Soft computing as a domain provides a foundation for artificial intelligence techniques to be further fully explored and integrated into a 'thinkable' machine.

In order to achieve a 'thinkable' machine that can be used to solve many real-world problems, learning for machine needs to be defined and all the requirements are identified and provided. Educationists like Weiss and Bloom believe that learning is a process of applying captured knowledge towards desirable behavioural outcomes. It is clear that learning must be associated with behaviour changes, knowledge acquisition and memory storage. These three fundamental elements must be a basis for any learning to be applied to.

In the next section, the learning concept will be presented, followed by the UMAS description that outlines the functions of UMAS has.

2 Learning Definitions

One of the main features for the UMAS to be distinctive from a conventional program is the ability of learning. Semantically learning is a process of acquiring knowledge for the desired behavioural outcomes. This only can be achieved if a 'learner' has some degrees of reasoning in which knowledge is used for comparison and logic computation. However, for an agent of the UMAS to do learning it means that it is capable of making a correct decision upon users' requests for network resources. Given a program, P, and some input x, a normal program would produce the same result, y. $P(x) = y$ for every request. In the case of agents in the UMAS, the program can alter its initial state, q, so that the decision is made on the account of several knowledge domain and accordingly. Thus, $P(x|q) = y$, where y is the result of applying program, P, to input, x , given the initial state, q [1].

Bigus [1] states several forms of learning that can take place. The most common learning that obviously takes place is a rote learning, in which an example is given and agent copies the example and

exactly reproduces the behaviour. In the case of UMAS, the agent will produce the result of the training data. For example, simulation results are presented to an agent to generate a new situation and later use the presented results for responding to a new situation occurrence. Another form of learning is parameter or weighted adjustment. In this type, important factors are known but the weight of their impacts on final result is unknown. This is a basis for neural network learning. Induction is a process of learning by example in which extraction of important characteristics of the problem is carried out for generalizing an input.

Another type of learning is clustering, chunking or abstraction of knowledge. It is about detecting a common pattern and generalise to a new situation. By chunking ten cases into one more general case, storage and searching time can be improved. Clustering looks at high-dimensional data and has similarity that bases on some criterion [1].

3 Learning Paradigms

Another approach to learning is called a machine learning (ML) that has at least three major paradigms of learning – supervised, unsupervised and reinforcement. In a supervised learning, agent is trained by showing it examples of the problem states or attributes along with the desired output or action. Agent will make prediction and if differs from the desired output, agent is adjusted or adapted to produce the correct output. This process is repeated several times until the agent makes accurate classifications or predictions. Leaning of this type requires historical data from databases, sensor logs, trace logs as a training and example data and learning algorithms that normally used are back propagation neural network and decision tree.

The second paradigm is unsupervised learning in which agents need to recognise similarities between inputs or to identify the features in the input data. Data that is presented to the agents is partitioned into groups by clustering or segmenting. The process of clustering or segmenting continues until the same data is placed in the same group. Common attributes of the data are extracted and technique that normally used is Kohonen Map. Finally, the third paradigm is reinforcement learning that is used when explicit input/output pairs of training data are not available. Reinforcement learning also is used when there is a sequence of inputs and the desired output is only known after the specific sequence occurs. The process of identifying the relationship between a series of input values and a later output values is called temporal credit assignment. It is the most realistic learning even though taking longer time and less efficient.

The UMAS will use the Machine Learning (ML) approach of acquiring knowledge for learning purposes.

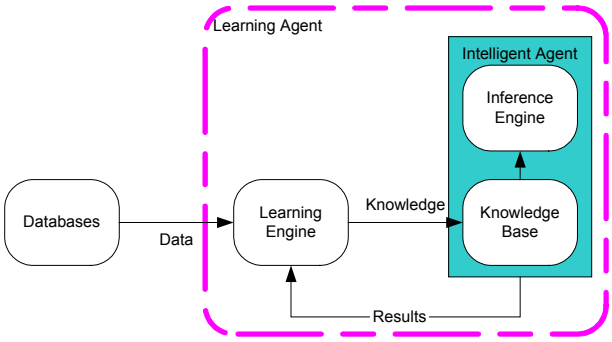

Figure 1: Overview of the General Machine Learning Approach

There are a number of learning algorithms and each has a specific suitability for different kind of learning. UMAS will perform the learning activities upon the data that was collected on three parameters – policy, profile and network states [2].

4 UMAS Description

Generically, UMAS is an interface between users and networks for the purpose of controlling network resources that are requested by users. Figure 2 shows a high level diagram of UMAS with the example of network resources. The Router and server are examples of network resources that are critical in the use of network applications.

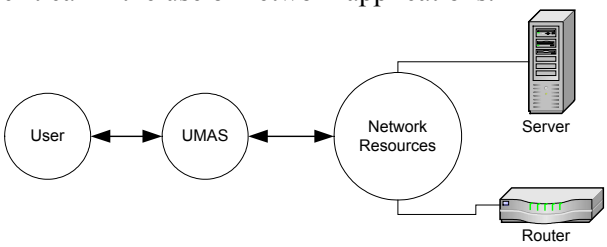

Figure 2: High Level Diagram of UMAS

Based on three sets of data, UMAS determines the access time for a specific session of a particular user. This measure is needed so that network resources particularly bandwidth can be allocated for only legitimate purposes. For example, if a number of users accessing network resources are relatively small compared to available resources,

the least critical applications are allowed to be accessed. However, if the number of users increased, then the least critical application of the session will be discarded from the network.

When a user logs in to the network, the UMAS will read the profile file of the user and use the knowledge base of the network. A system administrator creates a Profile file and it contains a collection of user settings such as desktop attributes and allowed network connections. A Knowledge base is a repository for learned knowledge of the use of network elements such as router, bandwidth, servers and applications. This knowledge is critical in the process of granting access to the network application request because it helps the UMAS to decide the best option for a network to be utilised for a particular session. The decision making process for determining a session period will be updated periodically at the interval time of 100 ms., which is believed to be sufficient for establishing communication between UMAS with other network objects.

The UMAS consists of several agents that each is assigned with a specific task. Figure 3 illustrates the components of the UMAS in which each component has a specific task to be performed and governed by the manager that resides in the UMAS.

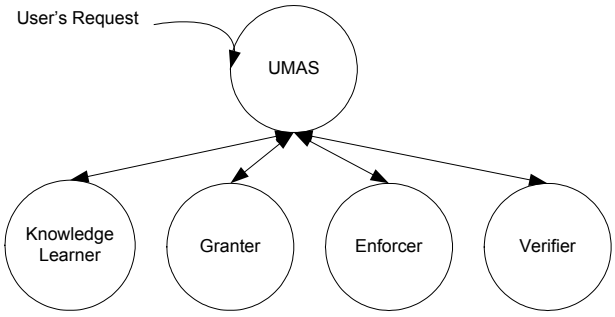

Figure 3: UMAS components

The Knowledge Learner is responsible for capturing a set of variables and these are stored in the repository for future retrieval. When a request has been received by the UMAS for accessing a particular application, it needs to be processed with the consideration of network resource availability and the criticality of the application to the organisation's objectives. The UMAS will forward the message of request to the Granter for a decision whether to approve or otherwise. In deciding on the request, the Granter relies on information about the users that are currently logging in to the network and the network resource usage in order to determine if a potential approval will be detrimental to the performance of the network. The Granter will use Fuzzy set data supplied by the Knowledge Learner for computing the time that practically can be allocated for the session. The decision taken by the Granter then will be forwarded to the UMAS for execution.

The component of Enforcer is responsible for executing the decision by communicating with the routers to inform them of changing packet priority. At the same time, the Granter will set the clock on and start counting. However, the Granter needs to inform the Verifier as well because it is responsible for verifying the decision by checking the user profile with the server. If the allocated time is 'correct' then the user can have such application to be accessed otherwise a new computation of time will be asked. Of course, the UMAS needs to be informed about any decision taken by both the Granter and the Verifier, as it needs to inform the user too [3]

5 UMAS Learning

The UMAS adopts the supervised learning notion in which training data is supplied to the Granter for deciding the allocation of time. However, the size of the training data sets are relatively large and contain a number of sets, the learning processes are conducted in stages. This is due to the dependency of learning input to the previous learning outcomes. A unit of processor handles a particular set of training data and the outcomes are used for a next learning process.

The goal of UMAS learning is to minimise the error rate of the time allocation decision by the Granter. This can be achieved by training the data through a number of attempts. The smaller value of the error rate, the better it would. The learning will base on the following neuro diagram.

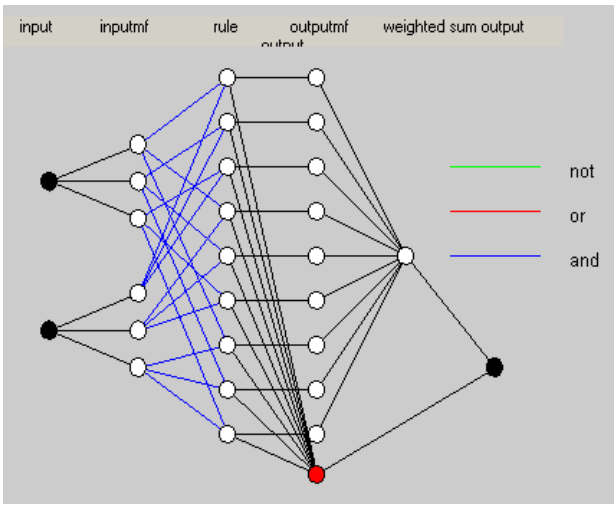

Figure 4 : Neuro diagram

Figure 4 shows only two inputs for the training data will be used. This is because the limitation of ANFIS to do the learning.

6 UMAS Learning Simulation

The simulation has been conducted using Matlab ver 5.3. The purpose of simulation was to establish rules with minimum error rate that will be integrated into the UMAS learning mechanisms. In order this, the training data that was collected using Windump and Windows Network Monitor tools was converted into numeric format before being put into use. This is because ANFIS, the learning algorithm of neuro fuzzy logic can only accept numeric representations and limited input [4][5][6]. For that reason, the simulation was based on two set of inputs – application criticality and user priority. Both inputs were given a similar scale of 1 – 10. A smaller number for application criticality denotes the higher criticality it was.

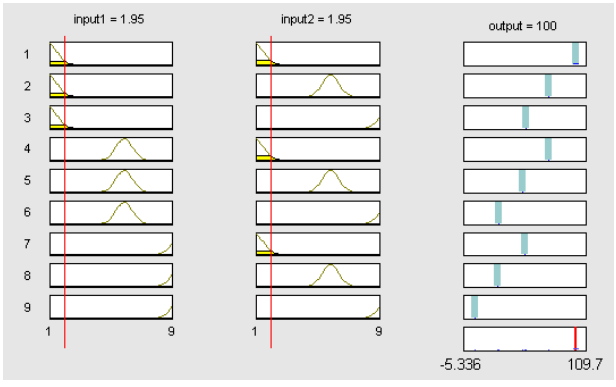

Figure 5: Generated rules

Figure 5 illustrates the rules that have been automatically created by the ANFIS, which shows the allocated time is 100% if the application criticality and user priority are high. On the other hand, the allocated time drops substantially if both inputs are decreasing. However, because of the criticality is given higher weighting, any changes in the values significantly affect the outcome of the allocated time.

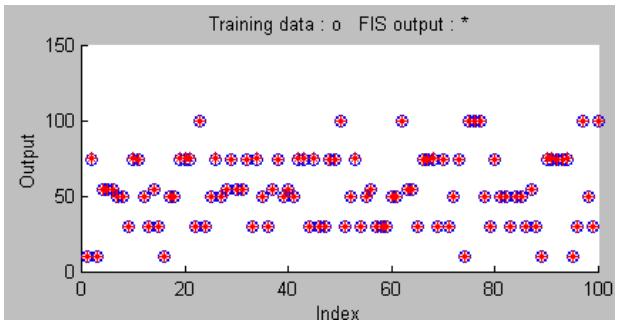

Figure 6: Results of the simulation

Figure 6 is the results of the simulation in which clearly shown that all data were distributed into three clusters – high, medium and low. Most the data were scattered in the medium interval and with minimum error rate. The results of simulation indicate that learning had taken place in which error was 0.0000158

7 Conclusion

The undertaken simulation had generated a set of rules with minimum error rate that can be integrated into the UMAS agents' development. In addition, the outcome of the simulation can be used for the Granter to decide the best allocation of session time for a user.

The UMAS will have the capability of making decisions intelligently based on the current situation of the network and the user's roles through the undertaken learning. This will enable the network resources to be allocated for achieving the main aims of network establishment and the organisation's goals. At the same time, users can receive appropriate levels of network services, in term of application availability and satisfied network responsiveness. As the UMAS has learning capability, the Knowledge Base will be gradually getting larger and this will lead to higher intelligence of the UMAS [7].

The idea of UMAS can be applied into another network management activities such as traffic management, Quality of Service and as a 'teacher' for training an agent that is newly introduced to the network.

Reference

[1] J.P. Bigus, Jennifer Bigus, *Constructing Intelligent Agents Using Java*, 2nd.ed. John Wiley, 2001.

[2] L.L.Peterson and B.S.Davie, *Computer Networks – A System Approach.* Morgan Kaufmann, 2000.

[3] Abdullah Gani, et.al., "The Roles of Intelligent User Manager Agent for Controlling an Access to Network Resources," presented at *3rd Annual PostGraduate Symposium The Convergence of Telecommunications, Networking and Broadcasting*, John Moore Univ., Liverpool, UK, 2002.

[4] Michael Knapik and Jay Johnson, Developing Intelligent Agents for Distributed Systems, Exploring Architecture, Technologies, and Applications, 1998, McGraw-Hill.

[5] Eral Cox, The Fuzzy Systems Handbook, A Practitioner's Guide to Building, Using, and Manipulating Fuzzy Systems, Second Edition, 1999, AP Professional.

[6] Adrian A. Hopgood, Intelligent Systems for Engineers and Scientists, Second Edition, 2001, CRC Press LLC.

[7] V. R. B. Rudi Studer, Dieter Fensel, "Knowledge Engineering : Principles and methods," *Data and Knowledge Engineering*, vol. 25, pp. 161-197, 1998.