Computational Intelligence and Active Networks

BEHZAD MOSHIRI and MAHDI JALILI-KHARAAJOO
CIPCI, ECE Department
University of Tehran

P.O. Box: 14395/1355, Tehran, Iran

Abstract: - In this paper, some application of computational intelligence techniques in active networking technology well be presented. Computational intelligent techniques, e.g., neural networks, fuzzy systems, neuro-fuzzy systems, and evolutionary algorithms have been successfully applied for many engineering problems. The introduction of active networking adds a high degree of flexibility in customizing the network infrastructure and introduces new functionality. Therefore, there is a clear need for investigating both the applicability of computational intelligence techniques in this new networking environment, as well as the provisions of active networking technology that computational intelligence techniques can exploit for improved operation.

Key-Words: - Active Networks, computational intelligence, fuzzy logic, neural networks.

1 Introduction

The events in the area of computer networks during the last few years reveal a significant trend toward open architecture nodes, the behavior of which can easily be controlled. This trend has been identified by several developments [1,2]. Active Networks (AN), a technology that allows flexible and programmable open nodes, has proven to be a promising candidate to satisfy these needs. AN is a relatively new concept, emerged from the broad DARPA community in 1994–95 [1,3,4]. In AN, programs can be "injected" into devices, making them active in the sense that their behavior and the way they handle data can be dynamically controlled and customized. Active devices no longer simply forward packets from point to point; instead, data is manipulated by the programs installed in the active nodes (devices). Packets may be classified and served on a per-application or per-user basis. Complex tasks and computations may be performed on the packets according to the content of the packets. The packets may even be altered as they flow inside the network.

Computational Intelligence (CI) techniques have been used for many engineering applications [5-10]. CI is the study of the design of intelligent agents. An intelligent agent is a system that acts intelligently: What it does is appropriate for its circumstances and its goal, it is flexible to changing environments and changing goals, it learns from experience, and it makes appropriate choices given perceptual limitations and finite computation. The central scientific goal of computational intelligence is to understand the principles that make intelligent behavior possible, in natural or artificial systems. There are some concepts of CI like: fuzzy sets and systems, neural networks, neurofuzzy networks and systems, genetic algorithms, evolutionary algorithms, and etc.

Due to highly nonlinear behavior of telecommunication systems and uncertainty in the parameters, using the CI and Artificial Intelligence (AI) techniques in these systems has been widely increased in recent years [11-17]. CI and AI techniques can be used for the design and management of communication networks. In the network-engineering context, such techniques have been utilized to attack different problem. A thorough study of application of CI in traditional networks, such as, IP, ATM, Mobile networks can be found in the literature [16-19]. In this paper, the application of CI and AI techniques for active networks technology will be studied. CI can be employed to control prices within the market or be involved in the decision process. The environment we provide can act as a testbed for attacking several other control problems in a similar fashion.

2 Active networking technology

Active networking technology signals the departure form the traditional store-and-forward model of network operation to a store-compute-and-forward mode. In traditional packet switched networks, such as the Internet, packets consist of a header and data. The header contains information

such as source and destination address that is used to forward the packet to the next element that is closer to the destination. The packet format is standardized and processing is limited to looking up the destination address in the routing tables and copying the packet to the appropriate network port. In active networks, packets consist not only of header and data but also of code. This code is executed on the active network element upon packet arrival. Code can be as simple as an instruction to re-send the packet to the next network element toward its destination, or perform some computation and return the result to the origination node. Additionally, it is possible for these packets to install code whose lifetime exceeds the time that is needed for the active packet to be processed. Software modules that are installed in this fashion are called active extensions. Active extensions facilitate for software upgrades, new protocol implementations, system and network monitoring agents. Other potential applications that need control functionality to be installed on demand are also made possible. This is a major breakthrough compared to the current situation where network elements come with a set of configurable, yet pre-installed options at the time the element is shipped. The install new functions, one has to bring the infrastructure off-line to manipulate its functionality. The architecture of an active node is shown in Fig. 1, based on the DARPA active node reference architecture [20].

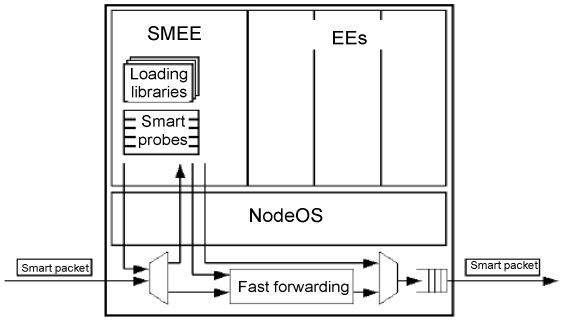

Fig. 1. Architecture of an Active Node.

The conventional node hardware and software provide the basic forwarding, routing, and signaling capability of the node. Packets typically pass through the normal forwarding fast path of the node. Packet filters detect *smart packets* that need active processing, which are demultiplexed and sent up the active path. The NodeOS provides operating system services for the active node. Execution Environments (EEs) provide the language and execution environment for active code. Active Applications (AAs) execute to provide active services. The Smart Environment for Network Control, Monitoring and Management (SENCOMM) is an EE that provides an environment for the execution, monitoring, and control of AAs for node management (MAAs), which is shown by SMEE in the figure.

In active networks, functionality can be deployed in the infrastructure dynamically, and can be easily removed or replaced. This offers more flexibility in deploying early implementations (without having to stall on the standardization process), protocol bridged (that translates between different revisions/generations of a service as in the active bridge), and most importantly, services themselves: users are free to customize the network infrastructure to fit their needs, when such needs emerge. This means that, network operation, forms the low layers of the architecture up to the application layer, and could be dynamically customized to provide CPU and packet scheduling to suit application needs. Value-added services can be installed on network elements by appropriately authorized users to create a true open-service market. From a business perspective, one could even think of scenarios where a network operator might want to rent out a router or vine a whole network infrastructure, or contract another organization to manage the network. Secondly, the "en-to-end argument" , one of the Internet's key design principles, being attributed both to its success and some of the most serious problems, is being questioned by the active networking paradigm. In its original manifestation, the idea was to have the least possible functionality (and thus complexity) within the network and push the intelligence to the end-systems.

3 Computational intelligence

Computational Intelligence (CI) [7,10,19] is an area of fundamental and applied research involving numerical information processing (in contrast to the symbolic information processing techniques of Artificial Intelligence (AI)). Nowadays, CI research is very active and consequently its applications are appearing in some end user products. The definition of CI can be given indirectly by observing the exhibited properties of a system that employs CI components [10]:

A system is *computationally intelligent* when it deals only with numerical (low-level) data, has a pattern recognition component, and does not use

knowledge in the AI sense; and additionally, when it (begins to) exhibit

- computational adaptivity;
- computational fault tolerance;
- speed approaching human-like turnaround;
- error rates that approximate human performance.

The major building blocks of CI are artificial neural networks, fuzzy logic, neurofuzzy systems and evolutionary computation. In the following, these topics will be briefly reviewed.

3.1. Artificial neural networks

Artificial neural network (ANN) computing is an approach that simulates the human brain's neurons. This machine learning technology has ability to process massive parallel data and recognize patterns [21]. The ability of neural network model to approximate the noisy and incomplete data sample helps explain its current popularity [22]. Over the last decade, the application of ANN has solved many problems like production/operation, finance, human resource, accounting, marketing, and engineering. Moreover, Backpropagation Neural Network (BPN) is one of the most popular learning paradigms of the network models [21]. There are many input and output nodes called processing elements in ANN. The outputs are connected to the inputs through connection weights [23]. With these interactive nodes and weights, ANN learns and adjusts until they achieve optimal situation.

3.2. Fuzzy sets and systems

Fuzzy Logic, invented in 1965 by Prof. Lotfi A. Zadeh [25], is a branch of mathematics that allows a computer to model the real world the same way that people do. Unlike computers, people are not always precise. People think and reason using linguistic terms such as "hot" and "fast", rather than in precise numerical terms such as "100 degrees" and "70 miles per hour." In addition, people can make "shades of gray" decisions, rather than absolute "black and white" or "yes/no" decisions. Fuzzy Logic provides a computer with the capability to make the same kinds of classifications and decisions that people do. The most common use of Fuzzy Logic is in fuzzy expert systems.

The way Fuzzy Logic works is through the use of *fuzzy sets*, which are different from traditional sets [26,27]. A set is simply a collection of objects. Traditional sets impose rigid membership requirements upon the objects within the set. An object is either completely in the set, or it is not in the set at all. Another way of saying this is an object is a member of a set to degree 1 (completely in the set) or 0 (not in the set at all). Usually, fuzzy sets and granular computing, promote top-down design approaches. In particular, we first capture the skeleton of the problem and subsequently refine the solution by moving into processing smaller, more detailed information granules. When dealing with fuzzy sets, all processing is carried out at the level of such information granules.

3.3. Neurofuzzy systems

Neurofuzzy systems are currently one of the "flavors of the month" in the neural network and fuzzy logic communities. They attempt to combine the structural and learning abilities of a neural network with the linguistic initialization and validation aspects of a fuzzy system [28,29]. Neurofuzzy networks are a particular type of fuzzy system that uses algebraic operators and continuous fuzzy membership functions, as it has been shown that these are generally the best for surface fitting problems. There are several types of neurofuzzy system, and these are categorized by the type of membership functions; the two most common are B-splines and Gaussians. By regarding these fuzzy systems as types of neural networks, the role of the membership function in determining the form of the decision surface is highlighted, rather than its accuracy in modeling the vagueness or uncertainty associated with a particular linguistic term. This interpretation also provides several possibilities for utilizing inductive learning-type techniques to produce parsimonious rule bases. Thus, the mathematical rigor associated with adaptive neural networks can be used to analyze the behavior of learning neurofuzzy systems and improve their performance as data-driven and human-centered approaches are being combined to improve the network's overall performance.

3.4. Evolutionary programming algorithms

Stochastic optimization algorithms [30-33], like evolutionary programming, genetic algorithms and simulated annealing, have proved useful in solving difficult optimization problems. In this context, a difficult optimization problem might mean: (1) a non-differentiable objective function, (2) many local optima, (3) a large number of parameters, or (4) a large number of configurations of parameters. Some important topics that can be discussed in this

section are: genetic algorithms, simulated annealing, ant colonies, and etc.
Whereas genetic algorithms include a variety of operators (for example, mutation, cross-over and reproduction), evolutionary programs use *only* mutation. As such, an evolutionary program can be viewed as a special case of a genetic algorithm.

4 Application of CI in AN

The features of active networks that are appealing to us in our attempt to utilize computational intelligence techniques are mainly:

- *Programmability:* We are able to enhance the network infrastructure on the fly. It is up to us to decide what functions we want to add, where to add them and when to do this. For example, we can easily replace algorithm A with algorithm B when cost or performance indicates such a need.
- *Mobility:* Our function does not need to be located on a single element throughout its lifetime. As is the agent paradigm, the programs we inject into the network might migrate from element to element to perform different tasks.
- *Distributivity:* Our function can be implemented as a distributed algorithm with agents performing tasks and exchanging information throughout the infrastructure.

All these are in sharp contrast with the state of the art, where control is either hard-coded into the software to the network elements that comprise the infrastructure or has to be performed through remote control interfaces (such as the configuration console, management and other protocols such as SNMP etc.).

4.1. Application of computational intelligence to providing an economic market approach to resource management

In this section, market-based resource management architecture for active networks will be adopted. Markets provide a simple and natural abstraction that is general enough to efficiently and effectively capture a wide range of different issues in distributed system control. The control problem is cast as a resource allocation problem, where resources are traded within the computational market by exchange of resource access rights. Resource access rights are simply credential that enables a particular entity to access certain resources in a particular way. These credentials result from market transactions, thus transforming an active network into an open service market. Dynamic pricing is used as a congestion feedback mechanism to enable applications to make policy controlled adaptation decisions. This system focuses on the establishment of contracts between user and application as well as between autonomous entities implementing the role of resource traders is emphasized. While this architecture is more oriented to end systems, the enhancements and the mechanisms described are well suited for use in active networks; Market-based control has also been applied to other problems such as bandwidth allocation, operating system memory allocation and CPU scheduling. The control problem can be redefined as the problem of profit or utility maximization for each individual player. The concept of a market is general enough to embrace all aspects of resource control in different contexts and different time scales. From a system engineering perspective, market-based control augments system-level, design-level and administrative-level control.

4.2. Application of computational intelligence for routing

Routing is one of the most fundamental and at the same time most complex control problems in networking. Its function is to specify a path between two elements for the transmission of a packet or the set-up of a connection for communicating packets. There are usually more than one possible paths and the routing function has to identify the one that is most suitable, taking into consideration factors such as cost, load, connection and path characteristics etc. In best effort networks, such as the Internet, routing is based on finding the shortest path, where the shortest path is defined as the path with the least number of "hops" between source and destination. For more advanced network services, such as the transmission of multimedia streams that require qualitative guarantees, routing considers factors such as connection requirements (end-to-end delay, delay variation, mean rate) and current (or future) network conditions. Furthermore, the information available to the decision process might be inaccurate or incomplete. Given the above, routing becomes a complex problem, with many aspects, including the perspective of a multi-objective optimization problem. Maximization of resource utilization or overall throughput, minimization of rejected calls, delivery of quality of service guarantees, fault-tolerance, stability, security consideration for administrative policy are just a few of the properties that are requirements for an

acceptable solution. Issues of active organization and an approach for quality of service routing restricted to the case of routing connections with specific bandwidth requirements. Our goal here is to address the routing problem using CI. The solution we propose involves the following components:

- Roaming agents are moving from element to element to collect and distribute information on network state. Another difference is that in our work the agents operate within the metaphor of the active network economy rather than as an ant colony where ants are born, feed, reproduce and die. At each element, they communicate with the Resource Brokers or other Roaming Agents to trade topology information and connectivity costs. For efficiency, roaming agents are small-sized programs that take the form of active packets to travel throughout the infrastructure.
- Routing agents, at each network element, are responsible for spawning roaming agents and are also the recipients of the information collected by them. Routing agents also act as resource brokers for connectivity. Routing agents rely on the CI engine for setting their operational parameters and processing the information collected from the roaming agents.
- The CI engine is a set of active extensions that include several subcomponents. These subcomponents from a generic library-like algorithmic infrastructure. For each problem, the Configuration and Information Base (CIB) is used to store information that is specific to the problem domain. This information evolves as the engine learns the problem space. The CI engine can be further populated on demand with active extensions; however the above structure ensures scalability and ease of use by providing a set of commonly used tools.

The components we have currently implemented for the CI engine are:

- An Evolutionary Fuzzy Controller (EFC), which clusters paths according to their current state characteristics. This effectively hides or exposes details on routing information thus effectively controlling information granularity. For example, if information is inaccurate it is encapsulated as fuzzy set membership, or if information irrelevant it is eventually dropped.
- A Stochastic Reinforcement Learning Automation (SELA) which, given a set of input parameters, internal state and a set of possible actions computed the beast action out of the set. Applied to the routing problem, given a set of paths, it computes the "best" route for a given set of connection constraints.
- An Evolutionary Fuzzy Time Series Predictor (EFTSP) that can be used for predicting traffic loads on network links based on past link utilization information.

5 Conclusion

In this paper, some applications of computational intelligence techniques in active networking technology were presented. One of these possible applications is the novel approach to routing in active networks, which promises to address different aspects of the problem, using the strengths of computational intelligence and the infrastructural provisions of active networks. Another interesting application area is in problems that have a lower feedback cycle, such as traffic shaping and policing.

References:

[1] D. L. Tennenhouse *et al.*, A Survey of Active Network Research, *IEEE Commun. Mag.*, vol. 35, no. 1, Jan. 1997, pp. 80–86.

[2] R. Boutaba, A. Polyrakis, Projecting Advanced Enterprise Network and Service Management to Active Networks, *IEEE Network • January/February*. pp.28-33, 2002.

[3] K. Psounis; Active Networks: Applications, Security, Safety, and Architectures, *IEEE Commun. Surveys*, vol. 2, no. 1, 1st qtr. 1999.

[4] J. M. Smith, Activating Networks: A Progress Report, *Comp.*, vol. 32 4, Apr. 1999, pp. 32–41.

[5] W. Pedrycz, *Fuzzy Sets Engineering*, CRC Press, Boca Raton, 1995.

[6] M. Jalili-Kharaajoo, Predictive Control of a Continuous Stirred Tank Reactor Based on Neuro-Fuzzy Model of the Process, *SICE Annual Conference*, Fukui, Japan, 2003, pp. 770-775.

[7] W. Pedrycz, *Computational Intelligence: An Introduction*, CRC Press, Boca Raton, 1997.

[8] M. Jalili-Kharaajoo and H. Ebrahimirad, Improvement of second order sliding mode control applied to position control of induction motors using fuzzy logic, LNAI (Springer Verlag), *Proc. IFSA2003*, Istanbul, Turkey, 2003.

[9] Moshiri, B., Jalili-Kharaajoo, M. and Besharati, F., Application of fuzzy sliding mode

control based on genetic algorithms to Building Structures, *in Proc. 9th IEEE Conference on Emerging Technology and Factory Automation*, 16-19 September, Lisbon, Portugal, 2003.
[10] J.C. Bezdek, What is computational intelligence? In M. Zurada, J., II, R. J. M., and Robinson, C. J., editors, *Computational Intelligence* Imitating Life, pages 1–12. IEEE Press, 1994.
[11] Ascia, G., Catania, V., Ficili, G., Palazzo, S., and Panno, D. (1997). A VLSI fuzzy expert system for real-time traffic control in ATM networks. *IEEE Transactions on Fuzzy Systems*, 5(1), pp.20–31, 1997.
[12] Catania, V., Ficili, G., Palazzo, S., and Panno, D. A fuzzy expert system for usage parameter control in ATM networks. In Proc. GlobeCom95, pp.1338–1342, 1995.
[13] Cheng, R.-G. and Chang, C.-J. Design of a fuzzy traffic controller for ATM networks. *IEEE/ACM Transactions on Networking*, 4(3), pp.460–469, 1996.
[14] Park, Y.-K. and Lee, G. Applications of neural networks in high-speed communication networks. *IEEE Communications Magazine*, 33(10), pp.68–74, 1995.
[15] Pitsillides, A., Pattichis, C., Stekercioglu, Y. A., and Vasilakos, A. Bandwidth allocation for virtual paths using evolutionary programming (EP-BAVP). In *Proceedings of the International Conference on Telecommunications ICT'97*, Melbourne, Australia. pp.1163–1168, 1997.
[16] J. Bigham, L. Rossides, A. Pitsillides and A., Sekercioglu, Overview of Fuzzy-RED in Di.-Serv Networks, *LNCS 2151*, pp.1-13, 2001.
[17] F. Zelezn, J. Zidgek and O. Stepankov, A Learning System for Decision Support in Telecommunications, *LNCS 2151*, pp.83-101, 2001.
[18] Sekercioglu A., Pitsillides A., Vasilakos A., Computation Intelligence in Management of ATM Networks: a Survey of Current State of Research , *Soft Computing (5)*, pp.279-285 2001.
[19] Pedrycz W., Vasilakos A., *Computational Intelligence in Telecommunication Networks*, CRC Press, Boca Raton, 2000.
[20] K. Calvert, ed. Architectural Framework for Active Networks. AN draft, AN Architecture Working Group, 1998.
[21] Wong,B.K., Lai, V.S. and Lam J. A bibliography of neural network business applications research: 1994-1998. *Computer and Operation Research 27*, pp.1045-1076, 2000.
[22] Glorfeld, L.W. and Hardgrave, B.C. An improved method for developing neural networks: the case of ecaluating commercial loan credit worthness. *Computer and Operations Research Vol. 23, No. 10.*, pp.933-944, 1996.
[23] D. Mukesh. Hate statistics? Try neural networks. *Chemical Engineering :H.W. Wilson – AST*, Vol. pp.96-104, 1997.
[24] C.L. Chen, D.B. Kaber and P.G. Dempsey, A new approach to applying feedforward neural networks to the prediction of musculoskeletal disorder risk. *Applied Ergonomics (31),* pp.269-282, 2000.
[25] Zadeh, L.A. Fuzzy Sets, *Information and Control*, vol. 8, no. 3, pp. 338-353, 1965.
[26] Yager, R., On a general class of fuzzy connectives, *Fuzzy Sets and Systems, 4*, pp.235-242, 1980.
[27] Jalili-Kharaajoo, M. and Besharati, F., Fuzzy variance analysis model, LNAI (Springer Verlag), *Proc. ISCIC2003,* Antalya, Turkey, 2003.
[28] B. Kosko. Neural Networks and Fuzzy Systems. Prentice-Hall, Englewood Cliffs, NJ,1992.
[29] J.S.R. Jang and C.T. Sun. Neuro-fuzzy modeling and control. *Proc. IEEE, 83(3),* pp.378-406, 1995.
[30] Fogel, D.B., A comparison of evolutionary programming and genetic algorithms on selected constrained optimization problems, *Simulation, 64 (6),* pp.1051-1055, 1995.
[31] Goldberg, D.E. Genetic Algorithm in Search, Optimization, and Machine Learning. Reading, MA: Addison-Wesley, 1989.
[32] Jalili-Kharaajoo, M. and Besharati, F., Intelligent Predictive Control of a Solar Power Plant with Neuro-Fuzzy Identifier and Evolutionary Programming Optimizer, *in Proc. 9th IEEE Conference on Emerging Technology and Factory Automation*, 16-19 September, Lisbon, Portugal, 2003.
[33] Fogel, D., *Evolving Artificial Intelligence*, Ph.D. thesis, University of California, San Diego, 1992.
[34] S.B. Cho, Fusion of neural networks with fuzzy logic and genetic algorithm, *Integrated Computer-Aided Eng., 9*, pp.363–372, 2002.
[35] T. Fukuda, Fuzzy-neuro-GA based intelligent robotics, in: *Computational Intelligence: Imitating Life*, J.M. Zurada, R.J. Marks II, C.J. Robinson, eds, IEEE Press, pp.252–263, 1994.
[36] Kasabov, N. and Watts, M., Genetic algorithms for structural optimization, dynamic adaptation and automated design of fuzzy neural networks, *in Proc. International Conference on Neural Networks ICNN'97*, Houston, pp.1232-1237, 1997.