NEURAL: A Self-organizing Routing Algorithm for Ad Hoc networks

Vicente E. Mujica V., Dorgham Sisalem and Radu Popescu-Zeletin
Fraunhofer FOKUS Institute. Berlin, Germany.
{mujica,sisalem,zeletin}@fokus.fraunhofer.de

Abstract

This paper evaluates a self-organizing routing protocol for Ad Hoc network, called the NEUron Routing ALgorithm (NEURAL). NEURAL has been designed taking into account the learning and self-organizing abilities of the brain. More precisely, it was inspired by the synapses process between neurons, when a signal is propagated. Basically, the most significant characteristic of NEURAL is the uniform distribution of the information around the node's location based on the current changes in its neighborhood. Using a 2-hop acknowledgment mechanism, local information is monitored in order to be used for route selection method, classification procedures and learning algorithms.

1 Introduction

Inspired by the biological nervous system, Artificial Neural System (ANS) and neural networks are being applied to study a wide variety of problems in the areas of engineering and business [7, 22, 18]. A neuron is the individual computational element that makes up most artificial neural system models [11]. A neuron presents three major parts called *dendrites*, the cell *body* and a single *axon*. *Dendrites* are nerve fibers that are connected to the cell *body* or soma, where the nucleus is located. Extending from the cell body is a single long fiber called the *axon*. At the ends of the neuron, terminal branches, synaptic junctions or *synapses* connect the axon with the dendrites of the following neurons [16]. In a ANS system, the information is propagated between neurons using electrical stimulation along dendrites. High stimulation signal produces an output to the other neighbor neurons and so the information takes the right way to the destination, where a reaction will occur. Otherwise, low stimulation signal will be blocked by the neurons and the information will be not transported. The *synapse* is defined as the communication process between neurons, when a signal is propagated. This means that the information is forwarded using electrical stimulations along dendrites. The modification of synaptic weights provides the traditional method for the design of neural networks. Such an approach is the closest to linear adaptive filter theory, which is already well established and successfully applied in such diverse fields as communications, control radar, sonar, seismology and biomedical engineering [14].

The characteristics described above are desirable in the context of Ad Hoc networks. The first association is the *synapse* process between neurons as the capacity of a processing element to communicate with others (Routing). Thus, the amount of neighbors around a node can be represented as a probabilistic weight value or "synaptic weight". Nodes are free to move randomly and organize themselves arbitrarily. This means that the ad hoc network's topology changes rapidly and unpredictably as well as synaptic weights are changed in the local environment. The advantage of Artificial Intelligent algorithms for control problem in complex systems is that the weights can be found by examining the performance of a network as controller rather than by providing correct control signals for various input data [24].

The goal behind this paper is described a novel routing Algorithm called NEuron Routing ALgorithm (NEURAL), which it has been inspired by the synapses process. The design of NEURAL is based on three main phases, which apply some algorithms used in the area of neural networks. The *Pre-processing* phase involves a classification rule for Pattern Recognition called the K-Nearest Neighbor Rule. Afterwards, the *Route Discovery* phase considers an self-organizing algorithm based on the Kohonen model. And finally, the *Learning* phase employs a Trust and Reputation mechanism, which it has been integrated to the extension of the Kohonen model.

The rest of the paper is organized as follows: Section 2 introduces a background about Classification, Self-organizing and Artificial Intelligent System theory. The NEURAL architecture was developed in section 3 using the performance of three main modules. The conjuction of these modules provide tools to accomplish the implementation for a network simulator in section 4, and finally, conclusions in section 5.

2 Background of Artificial Neural Systems

2.1 Preprocessing for Pattern Recognition

The K-Nearest Neighbor Rule (K-NNR) [4] is proposed as simple classifiers algorithm which it is usually applied in the pattern recognition area. K-NNR assigns an object of unknown class to the plurality class among the K labeled "training" objects that are closer to it. Closeness is usually defined in terms of a metric distance on the Euclidean space with the input measurement variables as axes [21]. K-NNR Rule definition is enhanced as follows: Let x denote the center point of a small hypersphere with volume $\mathfrak{V}$, which its radius grows until contains exactly K points irrespective of their class label. Based on this sample of K points, K_{d} points belong for example to the class C_{d}. So that the calculation of the class-conditional density for the point x is given by

$$p(x|C_{\mathrm{d}}) = \frac{K_{\mathrm{d}}}{M_{\mathrm{d}} \cdot \mathfrak{V}} \tag{1}$$

The unconditional density can be similarly estimated from

$$p(x) = \frac{K}{M \cdot \mathfrak{V}} \tag{2}$$

While the prior can be estimated using

$$p(C_{\mathrm{d}}) = \frac{M_{\mathrm{d}}}{M} \tag{3}$$

Finally based on the Baye's theorem to give

$$p(C_{\mathrm{d}}|x) = \frac{p(x|C_{\mathrm{d}}) \cdot p(C_{\mathrm{d}})}{p(x)} = \frac{K_{\mathrm{d}}}{K} \tag{4}$$

The eqn. 4 is known as the K-nearest-neighbor classification rule. It involves finding a hypersphere around the point **x** which contains K points (independent of their class), and then assigning **x** to the class according to the largest number of representatives inside the hypersphere [4].

2.2 Kohonen Model for Self-Organizing Systems

The Kohonen model [17] employs a two-dimensional neuron layer. This layer is innervated by d input fibers (axons), which carry the input signal. This signal excites or inhibits the neurons of the layer via synaptic connections. We consider conditions under which the excitation of the neurons is restricted to a spatially localized region in the layer. The location of this region is then determined by those neurons that respond most intensively to the given stimulus. The mathematical formulation of the kohonen model is shown taking into account the three processes to define a Self-organizing system such as: *broadcasting of the input, selection of the winner, and adaptation* of the models in the spatial neighborhood of the winner.

2.2.1 Broadcasting of the input

An incoming signal pattern or input vector **x** ($\mathbf{x} = x_1, x_2, \ldots\ldots, x_{\mathrm{m}}$) represents the average activities x_l of the individual incoming fibers $l = 1, 2..., m$. The *strength* of the synapse or synaptic weight vector **w** between the neuron ***i*** and *m* neighbors is denoted by $\mathbf{w}_{\mathrm{i,j}} = w_{\mathrm{i,1}}\ ,\ w_{\mathrm{i,2}}, \ldots\ldots, w_{\mathrm{i,m}}$. Suppose a incoming stimulus in neuron *i*. This neuron forms in its dendritic tree a weight sum as

$$net_{\mathrm{i}} = \sum_{\mathrm{j}=1}^{m} w_{\mathrm{i,j}} \cdot x_{\mathrm{j}} \tag{5}$$

2.2.2 Selection of the "winner"

The above equations describe an active neuron ***i***, in which the total excitation is concentrated within a singles and connected *cluster* of ***j*** consecutive neurons. Kohonen suggests an approximation (net_{i}) for eqn. 5 using the position of the maximum excitation on the basis external signal x_l alone. More precisely, net_{i} is determined from

$$net_{\mathrm{i}} = \sum_{\mathrm{j}=1}^{m} w_{\mathrm{i,m}} \cdot x_{\mathrm{m}} = \mathbf{max}\,(\sum_{\mathrm{j}=1}^{m} w_{\mathrm{i,j}} \cdot x_{\mathrm{j}}) \tag{6}$$

Eqn. 6 sums up the essence of the competition process among the neurons, where a particular neuron b satisfies this conditions and it is called the *wining neuron* for the input vector **x**.

2.2.3 Adaptation procedure

Helge et al. [15] enhanced an adaptation step for the kohonen model, in which every synaptic change is limited to a neighborhood zone about the excitation center (neuron *i*). In this zone, the synaptic connections are changed such that a subsequent re-occurrence of the same or a similar stimulus will lead to an increased excitation in the neighborhood.

2.3 Learning Process in Neural Networks

Fischler et al. [8] defined learning as follows: *Learning is a process by which the free parameters of a neural network are adapted through a process of stimulation by the environment in which the network is embedded. The type of learning is determined by the manner in which the parameter changes take place.* In other words an effective learning process has a close relationship with the interaction between environment and neurons, which are ideally part of the medium.

Sensing trust and local reputation allows calculation of the control output variable that it was described by Ritter et al. [15] in the extended model of Kohonen. The implementation of a reputation management framework is enhanced for computing system in order to facilitate a distributed and efficient mechanism for resource selection and learning such as in the research area of Grid computing systems [10, 9]. For example, Alunkal et al. [1] applied a reputation management service based on community experiences to classify, select, and tune the allocation of entities, including resources and services provided by people. Grid systems introduce the term of Trust or Reputation such as defined Azzedin [2]: *"Trust is the firm belief in the competence of an entity to act as expected such that this firm belief is not a fixed value associated with the entity but rather it is subject to the entity's behavior and applies only within a specific context at a given time"*. On the other hand, reputation's definition is assumed as follows: *"the reputation of an entity is an expectation of its behavior based on other entities' observations or information about the entity's past behavior within a specific context at a given time"*.

3 Evaluating a modular architecture for NEURAL

The conjunction of three phases, which consider algorithms normally applied in the area of neural networks, provide robust and efficient tools to be implemented in NEURAL. The modular architecture in fig. 1 shows these modules as *Pre-processing*, *Route Discovery* and *Learning*. In the Pre-processing module, the K-Nearest-Neighbors Rule (K-NNR) is employed in order to provide classification rule to sense continuous changes in the network. The Route Discovery module computes a self-organizing routing algorithm using the Kohonen model. Finally, the performance of a trust mechanism is carried up in the learning module.

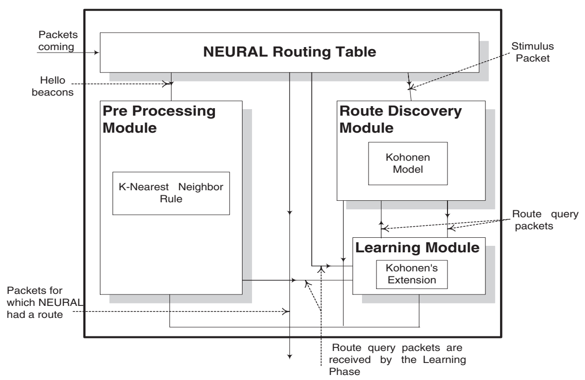

Figure 1. Schematic modular Architecture for NEURAL

3.1 Preprocessing Module

Based on sending hello request and reply packets during an interval of time (pre-processing interval), the preprocessing module adapts the K-Nearest-Neighbors Rule sensing the continuous topology changes of the network. K-NNR disseminates environmental changes using the activation stage (A_s). The activation stage, which it is configured using the *activation threshold* (K value from K-NNR), represents an estimation of the local density based on the 1-hop information. The value of this stage consists in two behaviors such as

Excitatory Behavior: nodes belonging to excitatory class are in a crowded area with high probability of interconnection (synapses). In our protocol, nodes in the excitatory class are identified by the the activation stage (A_s) equal to +1.

Inhibitory Behavior: In the case of an inhibitory class, areas are identified in Ad Hoc networks by isolated clusters. Nodes present a low synaptic capacity (*synaptic weight*) or poor communication ability with their neighbors. The activation stage (A_s) is set to -1.

For the pre-processing phase in NEURAL, we propose as flooding technique the "Post-synapse" schema, where this name is related with the communication process between neurons after that a signal arrived to the receptor neuron. Post-synapse describes the combination of a probabilistic schema applied in a generic epidemic protocol [12] together with the random query processing delay from the ZRP [13]. For example, nodes with an excitatory behavior present high local density, thereby they should manage low values of broadcasted probability (ProbBrcst) in order to avoid flooding due to useless broadcast packets. Otherwise, nodes , which belong to the inhibitory class, need to improve their connectivity in the environment using high values of ProbBrcst to spread their location for other neighbors. In sum, each nodes performs the post-synapse algorithm as follows.

$$\begin{array}{l} if\ (A_s = +1)\ then\ ProbBrcst = Excitatory\_Brcst \\ else\ ProbBrcst = Inhibitory\_Brcst \end{array} \quad (7)$$

where $Excitatory\_Brcst < Inhibitory\_Brcst$.

3.2 Route Discovery Module

NEURAL applies in the discovery phase a self-organized feature map called the Kohonen model, which it is used to select the next route in the MANET network based on a *competitive learning* procedure. In order to be consequent with the mathematical formulation of the Kohonen model in sec 2.2, the Route Discovery phase in NEURAL is subdivided in three steps.

3.2.1 Broadcasting

In this step, signals or inputs are broadcasted around MANET with the purpose of configuring input vectors from the Kohonen model. Suppose that the *processing element "i"* is going to select the following route between *j* neighbor nodes ($j \in \mathbb{N} : j > 0$). First, the two inputs vectors (the input vector $x_\mathbf{i}$ and synaptic vector $w_\mathbf{i,j}$) are initialized based on the information of the *local activation zone*. The local activation zone represent the diameter of a 2-hop distance (Neighborhood around the node *"i"*). The neighborhood is defined by "first" neighbors (n_j), which are *j* nodes around the *processing element i*, and "second" neighbors (N_m) that are *m* neighbors from the "first" neighbors. So that the information about the *local activation zone* is obtained by the node *i* broadcasting *"Stimulus"* messages to the "first" and "second" neighbors, respectively. Node *i* takes into account the relationship between distance with neighbors and the communication capacity of its neighbors. In NEURAL, the round trip time (RTT) is represented as the input vector $x_\mathbf{i}$ and the synaptic weight ($w_\mathbf{ij}$) is assigned to the number of "second" neighbors (N_m). The capacity of node to communicate with others neighbors is called *synaptic weight*. With the purpose of normalize vectors $x_\mathbf{i}$ and $w_\mathbf{ij}$, the following expressions are implemented for the node *i*.

$$x_\mathbf{i} = \exp^{\frac{-RTT_\mathrm{j}}{\tau_\mathrm{j}}}, \quad m = 1, 2...j \tag{8}$$

$$w_\mathbf{ij} = \frac{N_\mathrm{m}}{\tau_\mathrm{N}}, \quad m = 1, 2...j \tag{9}$$

Where RTT_j is the Round Trip Time vector of the *j*th "first" neighbors and the vector N_m denotes the number of "second" neighbors for the each *j*th node. In addition, τ_j and τ_N are the maximal values of the RTT and N_m observed in the *local activation zone* of the neuron *i*.

3.2.2 Selection of the "winner"

The "winner" is the node with the optimal "properties" to forward a packet to the destination. The node, which is searching a route, represents the central "stimulus" in the *local activation* zone. Thus, "first" neighbors initiate a competition to be selected as "winner". So that rules of this competition are based on the Kohonen model (See sec. 2.2). The best-matching neuron (winner) is selected by the maximum excitation signal $net_\mathrm{i,j}$, which it is given by eqn. 6 considering input vectors ($x_\mathbf{i}$ and $w_\mathbf{ij}$) from the broadcasting action.

3.2.3 Adaptation

Nodes that are topographically close in the array up to a certain geometric distance will activate each other to learn something from the neighborhood. With the purpose of implementing a control output variable, an trust mechanism is adapted in the *learning* phase based on the extension of Kohonen's model.

3.3 Learning module

The goal of this module is provide a learning parameter to be applied for control actions. Figure 2 illustrates a *learning* system architecture for NEURAL. Trust calculation is processed into three layers.

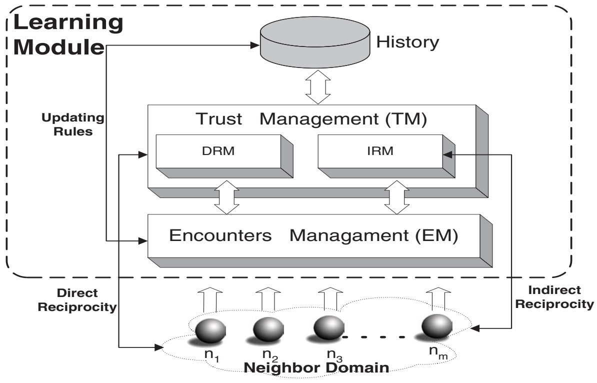

Figure 2. Information Layered Structure for the Learning Module

The lowest layer refers to the Encounter Management (EM). "Encounters" events are represented in NEURAL as the number of packets sent to neighbors in a certain period of time. The Trust Management (TM) corresponds the second layer of the learning architecture. It computes the trust value as a conjunction between a direct reputation module (DRM) and indirect reputation module (IRM). Direct reputation is calculated base on direct reciprocity with the neighbor domain. Direct reciprocity is related as the feedbacks between agents (I.e. nodes a_i and a_j) based on previous experiences. To compute the direct reputation $R^D_{j,i}(c)$, the number of previous encounters (e_prev) and the previous defect actions (α_prev) are given by the History-Table (HTable). A defect actions is described as number of packet lost detected and finally the reputation value is given by

$$R^D_{j,i}(c) = \frac{\alpha_\mathrm{prev}}{e_\mathrm{prev}} \tag{10}$$

Monitoring of the encounter event as well as defect action are carried up by the Encounter Management through the history table. In addition, the Trust Management calculates reputation values according to the information stored in the HTable. Destination nodes frequently monitor the number of packet losses and sent back to the source "reputation" packets.

4 NEURAL performance using a network simulator

In this section, we address the issue for configuring the NEURAL routing protocol to provide the best performance for particular networks. For this purpose, the network simulator *ns-2* [3] was employed to simulate the NEURAL module architecture described in the section 3. The network simulator *ns-2* is an object-oriented and discrete even-drive network simulator that in the recent year it has incorporated powerful tools, protocols and modules in the area of ad hoc networks as well as wireless networks.

4.1 Simulation scenario configuration

To make effective the convergence of Route Discovery and Learning phases in NEURAL, the optimal configuration of the pre-processing phase was evaluated using different scenarios. First, each mobile host has an omni-directional antenna having unity gain with a nominal radio range of 250 m. The random waypoint model [5] is selected as mobility model in a rectangular field with nodes' speed uniformly between cero and a maximum value of 20 $m.s^{-1}$. Number of sources and the sending rate are varied to study different loads for each configuration. To evaluate metrics, we consider the study of the packet delivery ratio, routing overhead and the average delay such as in [13, 20, 5]. Packet delivery ratio represents the fraction of control packet delivered to the destination. In addition, the total of routing packets transmitted in the simulation are measured using the routing overhead. And finally, the average delay shows the average one-way latency observed between transmitting and receiving a packet. It is important to mention that for each configuration, reported measurements are the mean of 10 runs with different random seeds.

4.2 Evaluating the impact of useless broadcast in NEURAL

To evaluate the performance of the Post-synapse algorithm, 20 nodes are configured using 20 seconds of pause time and a pre-processing interval of 5 seconds. So that every 5 seconds a hello packets is broadcasted for each nodes to start the pre-processing phase. Simulation was run during 150 seconds taking into account 5 different transition times: 15, 25, 50, 75 and 100 seconds.

The efficiency of the Post-synapse algorithm is first examined based on the delivery ratio for reply packets during diverse transition times. Figure 3 shows the delivery ratio for reply packets when the Post-synapse algorithm is applied to broadcast the hello messages of the pre-processing phase. So that percentages in the figure represent the "ProbBrcst" from the Post-synapse algorithm. For example, considering a ProbBrcst of 100% means that 100% of the hello messages were broadcasted to neighbors.

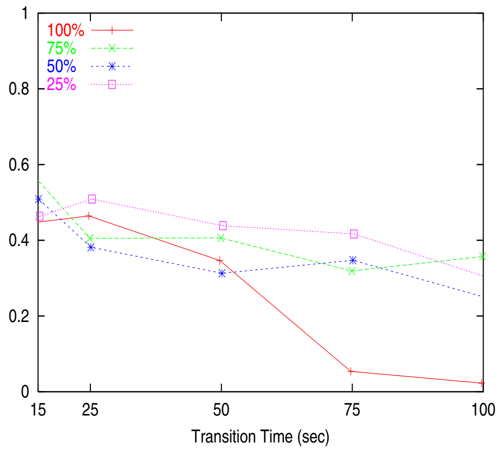

Figure 3. Delivery ratio for reply packets using Post-synapse algorithm

The real performance of the algorithm can be checked when the transition time is over 50 seconds. I.e., broadcasting all the hello messages in the pre-processing phase (ProbBrcst=100%), the control traffic (represented by hello and reply packets) introduces more collisions and packets loss into the wireless channel. Thus, the delivery ratio is abruptly decreased under 0,1 when the transition time rises. Otherwise, the packet delivery ratio show stable behavior with values between 0.38 and 0.4, when the broadcast probabilities were set to 25% and 75%, respectively. In this study, the routing overhead is related to the control traffic generated by the hello messages and reply packets in the pre-processing phase. Based on results depicted in fig. 4, we observe that the pre-processing phase overloads the network performing without a broadcast query algorithm. As example, metrics with 100% of Broadcast Probability shown around 2400, 2100 and 1300 routing packets for 100, 75 and 50 seconds of transition time, respectively. Thus, routing protocols that involve ack/nack mechanism to improve reliability, unfortunately, also tend to compromise their scalability by heavily loading the network. However, the performance of the Post-synapse algorithm as query mechanism allows to avoid query implosion in the network. In our case, Fig. 4 demonstrates that the overhead can be reduced until around 30% for the different transition times.

Taking into account the efficient performance of the Post-synapse algorithm over the control overhead (Fig. 4) in the network, we concentrate on improving the delivery ratio observed in the fig. 3. For this purpose, a random query processing delay (RQPD) was employed within the configuration of the Post-synapse algorithm. More precisely, RQPD mechanism addressed the problem of "simultaneous" reply messages by spreading out the packet with a random delay. Specifically, each node schedules a random delay prior to broadcast the packet [13].

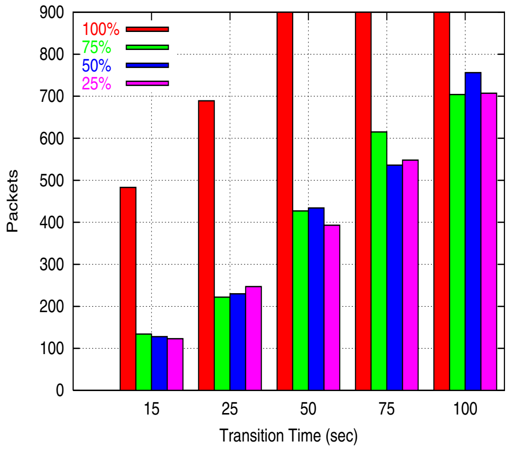

Figure 4. Number of routing packet sent using the Post-synapse algorithm

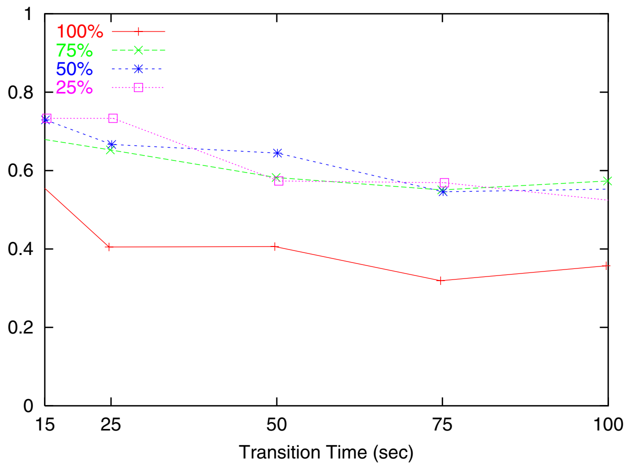

Figure 5. Delivery ratio for reply packets using the modified Post-synapse algorithm

Furthermore, we investigate the performance of the Post-synapse algorithm in the process of sending back a reply packets. In this case, nodes that receive a hello packet response to the source according to acknowledged probability (ProbAck). To verify the above modification in our algorithm, we consider the last simulation scenario with the following configuration parameters: a RQPD of 100 ms such proposed in [13] and ProbBrcst of 75%. With almost the same delivery ratio, the modified Post-synapse algorithm improved over 50% the performance for diverse transition times. For example in fig. 5, using 50 secs of transition time the best delivery ratio (0.65) was achieved with a acknowledged probability of 50%. Otherwise, when nodes response each received hello message (ProbAck=100%), redundancy appears as well as collisions increased in the wireless channel. For this case the delivery ratio was 0.4. In sum, the modified Post-synapse algorithm that applies a probabilistic behavior to send and response hello messages allows to increase the delivery ratio with a low total routing overhead such as fig. 6 illustrated.

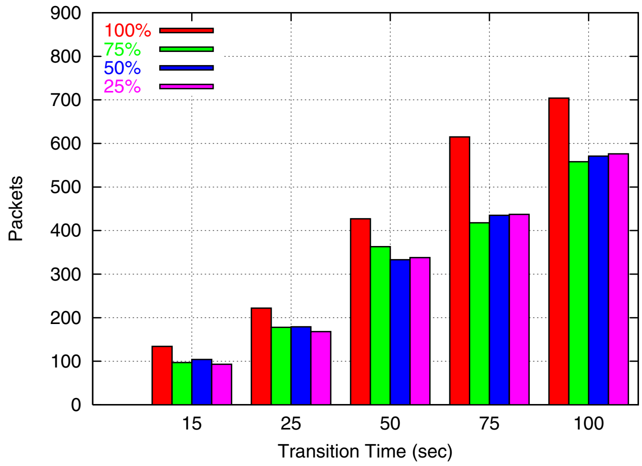

Figure 6. Number of routing packets sent using the modified Post-synapse algorithm

4.3 Study of the scalability of the preprocessing phase in NEURAL

In this section is furthermore proposed to improve scalability by segmenting the network into clusters, which are each managed by the performance of the K-Nearest-Neighbors Rule (K-NNR) in NEURAL. More precisely, we investigate the optimal configuration of the *activation threshold* or K value based on the algorithm described in section 2.1. For this configuration, 50, 100 or 200 nodes were placed in a field of 1300 x 500 meters. Pre-processing interval of 10 secs and pause time of 20 seconds.

Based on broadcasting *Hello* messages, nodes compare the number of acknowledgments (N_{ack}) with the *activation threshold* (A_t). After this initial procedure each node presents an activation state according to the number of neighbors. Nodes belong to the excitatory class employ broadcast probabilities (ProbBrcst) of 25 % and acknowledged probability (ProbAck) of 50% for the hello and reply packets, respectively. Otherwise, a probability of 75% was applied in both cases for nodes in the inhibitory class. Simulations was run varying the activation threshold as 0, 1, 3, 5, 7 and 10. These values have been applied considering the performance of the Weighted Clustering Algorithm (WCA) in Ad Hoc networks. WCA [6] is a weight based distributed clustering algorithm which takes into consideration the number of nodes a clusterhead can handle ideally (without any severe degradation in the performance), transmission power, mobility, and battery power of the nodes. Studies (such as [23]) related to the optimal configuration of WCA proposed that each clusterhead can at most handle ten nodes as ideal degree. The load handled by a clusterhead is essentially the number of nodes supported by it. As we mention before, the maximal activation threshold to classify cluster belonging to the inhibitory class is 10 nodes.

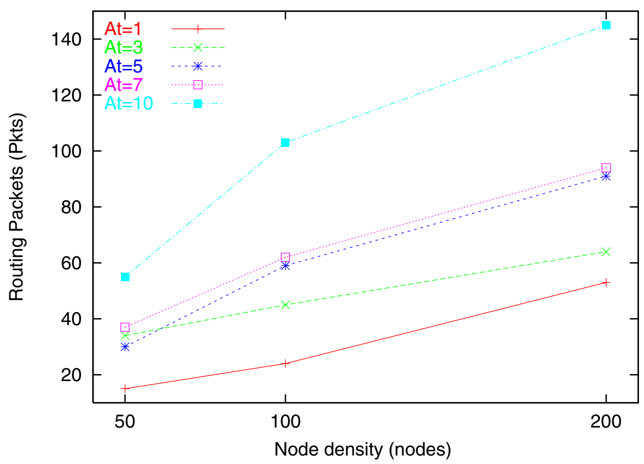

Figure 7. Scalability of the activation threshold

Fig. 7 illustrates the average of routing packets (hello and reply packets) per node. Metrics using activation threshold (A_t) equals cero represents that the K-Nearest-Neighbors Rule is not applied in the pre-processing modules. For this case, the lowest control overhead was obtained due to the algorithm considered that all nodes were within a crowded region. So that only 25% of the total hello packets were broadcasted to neighbors. Otherwise, based on a activation threshold of ten ($A_t = 10$), the K-Nearest-Neighbors Rule achieved to disseminate clusters with less than 10 nodes. These clusters were related with an inhibitory behavior broadcasting packets with a high probability (75%). For this reason, high control overhead results were obtained for each topology configuration using $A_t = 10$. Based on results for $A_t = 5$ and $A_t = 7$, the control packet loads the network with the same proportion. For these values, fig. 7 shows that the mean routing packets converged to 38, 59 and 95 packets per node for each configuration topology conformed with 50, 100 and 200 nodes, respectively. Finally, we note that the performance of the K-Nearest-Neighbors Rule has a better behavior when the activation threshold is seven ($A_t = 7$). Using this threshold the algorithm allows to classify clusters or inhibitory regions integrated with max. 7 nodes. Furthermore, results shown less overhead than $A_t = 10$ as the number of nodes increases. Metrics were compared with the performance of the weight clustering algorithm (WCA). Nuevo et. al reported in [19] around 32 routing packets using 50 nodes. In our case, the overhead was around 35 and 38 routing packets based on an activation threshold of 5 and 7, respectively.

5 Conclusions

In this paper, the design of a self-organizing routing algorithm called NEURAL is achieved using classification, adaptive and learning algorithms from the Artificial Neural System. This routing protocol is inspired by the synapses in the brain, in which neighbors neurons compete to propagate the signal. We contribute with the design of a modular architecture for the NEURAL protocol. Afterwards, simulations were carried up to provide the optimal configuration for the Pre-processing module in NEURAL. The pre-processing phase is related with a classification rule which operates broadcasting packets to a local region. Broadcast is widely used in sensor and ad hoc networks to disseminate information about environmental changes in the network. Therefore, it is essential to develop efficient broadcast protocols that are optimized for energy consumption and low control overhead. Thus, the “Post-synapse” algorithm was introduced in this paper as a query mechanism to avoid flooding due to useless broadcast packets in the NEURAL protocol.

The Pre-processing requires selection of the following parameters: periodic update interval (pre-processing interval), configuration of the the K-Nearest-Neighbors Rule (activation threshold) and searching of the optimal transient time to update routing tables in the simulations. These parameters will likely represent a tradeoff between the latency of the control information, delivery ratio for control packets and excessive communication overhead.

Our simulation results show improved performance for the pre-processing module of the NEURAL protocol in terms of low control overhead and high delivery radio for control packets as the Post-synapse algorithm was employed. When evaluating scalability of the pre-processing phase, the control overhead was managed taking into account the efficient performance of the K-Nearest-Neighbors Rule as classification rule. We finally conclude that by focusing in these configuration parameters, the NEURAL routing protocol can be hopefully improved in the performance of the Route Discovery and Learning phases. The above results conduce to further researches such as evaluating NEURAL based on the performance of all its modules and comparison with the current routing algorithms.

References

[1] B. Alunkal, I. Veljkovic, and G. von Laszewski. Reputation-based grid resource selection. In *Workshop on Adaptive Grid Middleware*, New Orleans, USA, 1999.

[2] F. Azzedin and M. Maheswaran. Towards trust-aware resource management in grid computing systems. In *In Second IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID'02)*, pages 452–457, Berlin,Germany, May 2002.

[3] U. Berkeley and U. ISI. The network simulator ns-2, 1998. Part of the VINT project. Available from http://www.isi.edu/nsnam/ns.

[4] C. Bishop. *Neural Network for Pattern Recognition*. Oxford University Press, Great Britain, 2000.

[5] J. Broch, D. Maltz, D. Johnson, Y. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc networks routing protocols. In *Fourth annual ACM/IEEE international conference on Mobile computing and networking*, pages 85–97. ACM Press, 1998.

[6] M. Chatterjee, S. Das, and D. Turgut. Wca: A weighted clustering algorithm for mobile ad hoc networks. In *Journal of Cluster Computing (Special Issue on Mobile Ad hoc Networks)*, volume 5 N2, pages 193–204, april 2002.

[7] I. Cloete and A. Skabar. Feature selection for financial trading rules. In *Proceedings of 13th. European Simulation Symposium: Simulation in Industry*, pages 713–717, Marseille, France, 2001.

[8] M. Fischler and O. Firschein. *Intelligence: The eye, The brain, and The computer*. Addison-Wesley, Reading, Massachusetts, 1987.

[9] I. Foster and C. Kesselman. The grid: Blueprint for a new computing infrastructure. *Morgan Kaufmann Publisher*, 1999.

[10] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the grid: Enabling scalable virtual organizations. *Int'l Journal on Supercomputer Applications*, 2001.

[11] J. Freeman and D. Skapura. *Neural Networks: Algorithms, applications, and programming techniques*. Addison-Wesley, Reading, Massachusetts, 1992.

[12] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. An empirical study of epidemic algorithms in large scale multihop wireless networks, march 2002. Intel Research, IRB-TR-02-003.

[13] Z. Hass and M. Pearlman. The performance of query control schemes for the zone routing protocol. In *IEEE/ACM Transactions on networking*, volume 9 N4, pages 427–438, Aug. 2001.

[14] S. Haykin. *Adaptive Filter Theory*. Pretice-Hall, Englewood Cliffs, NJ, 1991.

[15] R. Helge, T. Martinetz, and K. Schulten. *Neural Computation and Self-Organizing Maps- An Introduction*. Addisonn-Wesley, New York, 1992.

[16] J. Hertz, A. Krogh, and R. Palmer. *Introduction to the theory of neural computation*. Addison-Wesley, Santa Fe Institute, USA, 1992.

[17] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[18] P. Lednicky and A. Meszaros. Neural network modeling in optimisation of continuous fermentation processes. *Bioprocess Engineering*, 18:427–432, 1998.

[19] J. Nuevo and J. Gregoire. Proposition of a hirearchical service distribution architecture for ad hoc networks based on the weight clustering algorithm. In *Proceeding of European Wireless 2004*, Barcelona, Spain, february 2004.

[20] C. Perkins, E. Royer, S. Das, and M. Marine. Performance comparison of two on-demand routing protocols for ad hoc networks. In *IEEE Personal Communications*, volume 8 N1, pages 16–28, february 2001.

[21] V. Ramos, P. Pina, and F. Muge. From feature extraction to classification: A multidisciplinary approach applied to portuguese granites, 1999. Available from http://psych.athabascau.ca/html/Psych402/Biotutorials/.

[22] A. Skabar and I. Cloete. Discovery of financial traading rules. In *Proc. Artificial Intelligence and Applications (AIA2001)*, pages 121–125, Marbella,Spain, 2001.

[23] D. Turgut, S. Das, R. Elmasri, and B. Turgut. Optimizing clustering algorithm in mobile ad hoc networks using genetic algorithmic approach. In *Proceeding of GLOBECOM 2002*, Taipei, Taiwan, november 2002.

[24] E. Weeks and J. Burgess. Evolving artificial neural networks to control chaotic systems. *Physical Review E, The American Physical Society*, 56:1531–1540, Aug. 1997.