10

Artificial Neural Network for Cooperative Distributed Environments

Mauricio Paletta
Universidad Nacional Experimental de Guayana (UNEG)
Venezuela

1. Introduction

Distributed systems have become increasingly common because they offer significant computational power and are cost-effective and scalable. Moreover, collaboration between users that are part of these distributed systems improves efficiency and effectiveness for a better utilization of this computational power. Because of this, new specific collaboration/cooperative models for distributive systems are needed for enabling effective collaboration/cooperation between users of these dynamic environments or Cooperative Distributed Environments (CDEs). A CDE is then an environment in which multiple users in remote locations participate in shared activity aiming to achieve a common goal. Most of the CDE work towards providing reliable, customized and QoS guaranteed dynamic computing environments for end-users. The success of achieving this goal in proper time (efficiency) and/or to obtain the higher quality of results (effectiveness) depends on implementing an appropriate collaboration model that should include learning abilities necessary for the use of the previous experience acquired (with situations that occurred in the past) in order to improve new required collaborations.

On the other hand, according to CSCW (Computer Supported Cooperative Work) awareness is a useful concept used to achieve cooperation and collaboration in CDE as it increases communication opportunities (Matsushita & Okada, 1995). A collaborative process is leaded by five processes (Kuwana & Horikawa, 1995) (Malone & Crowston, 1994): 1) co-presence, that gives the feeling that the user is in a shared environment with some other user at the same time; 2) awareness, a process where users recognize each other´s activities on the premise of co-presence, for instance “What are they doing?”, “Where are they working?”; 3) communication; 4) collaboration which together with communication permit users to collaborate between each other for accomplishing the tasks and common goals; and 5) coordination which is needed to resolve the conflicts towards effective collaboration.

In the same order of ideas, in CSCL (Computer Supported Collaborative Learning), awareness plays an important role as it promotes collaboration opportunities in a natural and efficient way (Ogata & Yano, 1998) and improves effectiveness of collaborative learning. In this matter, Gutwin et al identified the following types of awareness (Gutwin et al, 1995): social, task, concept, workspace, and knowledge.

Moreover, SMI (Spatial Model of Interaction) (Benford & Fahlén, 1993) is one of the awareness models proposed with the purpose to obtain any knowledge of the immediately closer world in collaborative virtual environments. It is based primarily on the use of a variety of mechanisms that were defined for this model and to steer the interaction in a virtual environment. These are the concepts of medium, aura, focus, nimbus and awareness. The concept of awareness in this context, more explicitly awareness of interaction, is defined for quantifying the degree, nature and quality of the interaction between the elements of the environment.

This chapter focuses on the use of Artificial Neural Networks (ANNs) as an option to provide learning abilities to a collaborative model to meet goals mentioned above. First, this chapter makes a state of the art on this relationship between CDE and ANN specifically as it relates to the use of ANN to learn to collaborate and/or to improve collaboration. The chapter, then, develops a particular strategy based on the concept of awareness of interaction derived from SMI. The reason for using this strategy is because for cooperative tasks to be successful in CDEs they require from users to be known. Before this process is achieved it is important to know which users are more suitable in the system to cooperate with, as well as which tools are needed to achieve the common goal in the system in a cooperative way. In this regard, awareness allows users to be aware of others' activities each and every moment. Information about others' activities combined with their intentions and purposes could be used to improve cooperation in CDEs.

Finally, this chapter also includes the results of a recent research that was carried out which combine the concepts of awareness of interaction and ANN applied in a particular model known as AMBAR (Awareness-based learning Model for distriButive collAborative enviRonment). Some particular comments related with the ANN used in AMBAR are included in a different section of this chapter.

2. A summary of the state of the art

There are two different categories of works related with ANNs and CDEs: 1) those in which CDE is used to improve the ANN performance; and 2) those where an ANN is used aiming to improve certain processes relative to the CDE. This chapter is more oriented to the second category of researches.

2.1 Improving ANNs by using CDEs

Related with improving the ANN performance by using a CDE, Garcia et al (Garcia et al, 2002 and 2005) proposed a cooperative co-evolutionary model for the evolution of neural network topology and weights. Cooperative co-evolution is a recent paradigm in evolutionary computation that allows the effective modelling of cooperative environments. In a first work, authors proposed MOBNET (Garcia et al, 2002) that evolves subcomponents that must be combined in order to form a network, instead of whole networks. The problem of assigning credit to the subcomponents is approached as a multi-objective optimization task. The subcomponents in a cooperative co-evolutionary model must fulfil different criteria to be useful, these criteria is usually conflicted with each other. In this work authors show how using several objectives for every subcomponent and evaluating its fitness as a multi-objective optimization problem, the performance of the model is highly competitive. MOBNET is compared with several standard methods of classification and with other neural network models showing the best overall performance of all classification methods applied. It also produces smaller networks when compared to other models. Moreover, the basic idea underlying MOBNET is extensible to a more general model of co-evolutionary computation, as none of its features are exclusive of neural networks design.

In a second work of the same authors (Garcia et al, 2005), a general framework is proposed for designing neural network ensembles by means of cooperative co-evolution. The authors state that although theoretically, a single neural network with a sufficient number of neurons in the hidden layer would suffice to solve any problem, in practice many real-world problems are too hard to construct the appropriate network that could solve this problems. In such problems, neural network ensembles are a successful alternative. Nevertheless, the design of neural network ensembles is a complex task. The model proposed in this work has two main objectives: first, the improvement of the combination of the trained individual networks; second, the cooperative evolution of such networks, encouraging collaboration among them, instead of a separate training of each network. Authors concluded that the performance of the model is better than the performance of standard ensembles in terms of generalization error, as well as the size of the obtained ensembles that is also smaller.

On the other hand, another example of improving ANN by using CDE is related with implementing ANNs on a parallel or distributed platform to improve the training performance. Some works related with this subject are (Calbert & Guan, 2005), (Kiran, 2009) and (Wesley-Smith, 2006).

2.2 Improving CDEs by using ANNs

One of the fields where ANN is used aiming to improve some area related with distributed environments has to do with analyzing and monitoring different distributed sources of voluminous data, multiple compute nodes, and distributed user community. So that a data mining technology designed for distributed applications is required. The field of Distributed Data Mining (DDM) deals with this problem mining distributed data by paying careful attention to the distributed resources. A complete bibliography of DDM-related publications can be consulted in (Liu et al, 2006).

Another field to mention in this section is related with cooperative learning systems based on ANNs. In this regard Cristea and Florea (Cristea & Florea, 1999) present a cooperative distance learning system based on the emerging paradigm of intelligent human-computer interaction in which the group of learners is assisted by artificial agents with active role in the learning process. In this research, the tutor in the system may be a human or an artificial agent and the system offers several learning modalities that combine the traditional style of tutorial learning with the “problem based” approach. Moreover, cooperative learning is achieved either by interaction between the student and the tutor or interaction inside the group of learners.

On the other hand, an approach for the optimization of the job scheduling in large distributed systems, based on a self-organizing neural network is presented in (Newman & Legrand, 2000). In this approach, the dynamic scheduling system should be seen as adaptive middle layer software, aware of current available resources and making the scheduling decisions using the past experience. Another example of using neural network in a problem related with collaboration can be consulted in (Blanchard & Frasson, 2002). In this matter, authors present an architecture aiming to address the collaboration in a learning activity to create groups among students. Authors used a neural network algorithm to obtain homogenous groups. In (Yildiz, 2006) author described a load balancing approach by using graph partitioning and ANNs. The aim of this work is to integrate the successful load balancing decisions of graph partitioning algorithms with the efficient decision making mechanism of ANNs. The author affirms that the results obtained by him showed that using ANNs to make efficient load balancing can be very beneficial due to the fact that, once it is trained enough, the ANN may load the balance as good as graph partitioning algorithms or even more efficiently.

An interesting work, different than the others previously mentioned, is the proposal of a complex neural network model of user behaviour in distributed systems (Shelestov et al, 2007). This model reflects both dynamical and statistical features of user behaviour and consists of three components: 1) the on-line model that reflects dynamical features by predicting user actions on the basis of previous ones; 2) the off-line model which is based on the analysis of statistical parameters of user behaviour; and 3) the change detection module which is intended for trends analysis in user behaviour. In both on-line and off-line models neural networks are used to reveal uncharacteristic activity of users.

Regarding the context of awareness and recognizing the current context of a user or device, authors in (Mayrhofer & Radi, 2007) present an approach based on general and heuristic extensions to the growing neural gas algorithm classifier which allow its direct application for context recognition. The authors here used context awareness features for automatically classifying sensor data to recognize user or device context.

Perhaps the most currently related work that discusses the subject of using ANNs to improve some aspect of distributed environments is AMBAR (Awareness-based learning Model for distriButive collAborative enviRonment) (Paletta & Herrero, 2010a), and more specifically in its particular element called CAwANN (Collaborative Distributed Environment by means of an Awareness & Artificial Neural Network strategy) (Paletta & Herrero, 2009f). Both will be explained in detail in this chapter.

3. Cooperative environments by using spatial model of interaction

The aim of this section is to present a proposal to represent cooperative environments based on the SMI.

3.1 Spatial model of interaction (SMI)

SMI is, perhaps, the most well-known awareness model for multi-user environments. This model was developed between 1991 and 1993 by Professor Steve Benford at Nottingham University's School of Computer Science and Information Technology, Lennart E. Fahlén at The Swedish Institute of Computer Science (SICS) and John Bowers at The Royal Institute of Technology (KTH) in Stockholm (Sweden).

Most of the main concepts and ideas of this model emerged from a project, called COMIC, which was a three-year [1992-1995] basic research action to investigate techniques and develop tools for large-scale real-world CSCW application developers (Benford et al, 1994). This project aimed to examine and overcome the practical and theoretical problems limiting effective CSCW product development at that time. One such problem is that simultaneous interaction between all objects is not computationally manageable in any large-scale environment. For this reason, it is important to determine which objects are capable of interacting with other given objects at any given time.

As its name suggests, SMI uses the properties of space as the basis for mediating interaction. It was proposed as a way to control the flow of information in the environment. It allows objects in the environment to govern their interaction through some key concepts: medium, aura (Fahlén & Brown, 1992), awareness, focus, nimbus, adapters (Benford & Fahlén, 1993) and boundaries (Bowers & Rodden, 1993). This model provides a synchronous method of controlling how users and objects make themselves known to the world and how the world is aware of them. It has been driven by a number of objectives (Benford & Fahlén, 1993):

- Scalability: It is based on the concept of aura. Each object has an aura for each medium (visual, audio, text...) in which it can interact, because the aura defines the volume of space within which this interaction is possible. The use of aura facilitates scaling to many users by limiting the number of object interactions that must be considered. This number will be governed by the extent of the object auras and by the population density of the space.
- Interactions: The SMI assumes a space populated by potentially communicating objects. These objects may represent anything: human users or data in a database, for example. The space itself may have any form, for example, a three-dimensional Cartesian space, an abstract higher-dimensional space or a graph. The SMI provides a framework for these objects to manage their interaction, and communication with every pair of objects in the environment. A key component of this management of interaction is the use of the space itself. Thus by controlling their position, orientation, distance, etc., the objects are able to modify their interaction and communication (Greenhalgh, 1994).

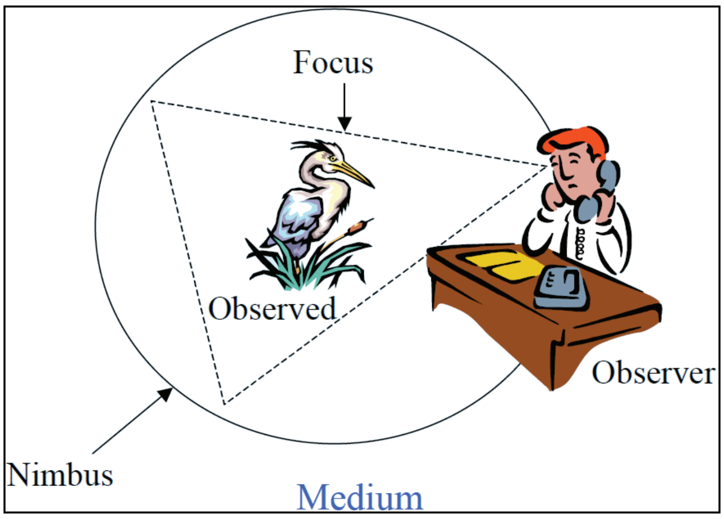


Fig. 1. Key concepts in The Spatial Model of Interaction (Herrero, 2003)

As it was mentioned before, the model itself defines five linked concepts: medium, awareness, aura, focus and nimbus (see Fig. 1 and 2):

- Medium: A prerequisite for useful communications is that two objects have a compatible medium in which both objects can communicate. This medium might include audio, video, graphics and text.
- Awareness: It is the main concept involved in controlling interaction between objects. It quantifies the degree, nature or quality of interaction between two objects. One object's awareness of another object quantifies the subjective importance or relevance of that object. The awareness relationship between every pair of objects is achieved on the basis of quantifiable levels of awareness between them (Benford & Fahlén, 1992) and it is unidirectional and specific to each medium (Benford & Fahlén, 1993).
- Aura: In 1992, Fahlén and Bowers defined aura as the sub-space which effectively bounds the presence of an object within a given medium and which acts as an enabler of potential interaction (Fahlén & Brown, 1992). Once aura has been used to determine the potential for object interactions (see Fig. 2), the objects themselves are subsequently responsible for controlling these interactions. "When two auras collide, interaction between the objects in the medium becomes a possibility" (Benford & Fahlén, 1993).

- Focus: In each particular medium, it is possible to delimit the observing object's interest. This idea was introduced by S. Benford in 1993 as "The more an object is within your focus the more aware you are of it" (Benford & Fahlén, 1993), and it was called Focus.
- Nimbus: similar to what was mentioned above, it is possible to represent the observed object's projection in a particular medium. This area is called Nimbus: "The more an object is within your nimbus the more aware the object is of you".

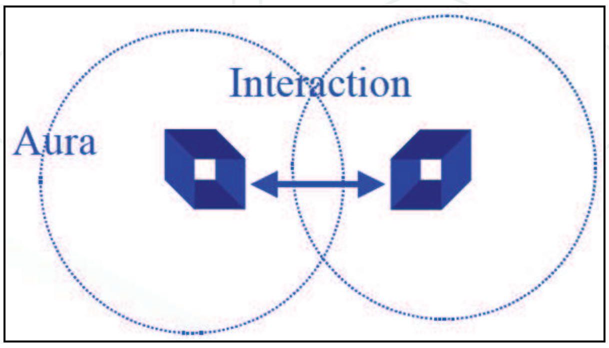


Fig. 2. Collision of two objects' auras (Herrero, 2003)

Therefore, awareness between objects in a given medium is manipulated via Focus and Nimbus, requiring a negotiation process. Considering, for example, A's awareness of B, the negotiation process combines the observer's (A's) focus and the observer's (B's) nimbus. In the words of Benford and Fahlén: "The level of awareness that object A has of object B in medium M is some function of A's focus on B in M and B's nimbus on A in M". For a simple discrete model of focus and nimbus, there are three possible classifications of awareness' values when two objects are negotiating unidirectional awareness (Greenhalgh, 1997):

- Full awareness: The awareness that object A has of object B in a medium M is "full" when object B is inside A's focus and object A is inside B's nimbus (Fig. 3).
- Peripheral awareness: The awareness that object A has of object B in a medium M is "peripheral" when object B is outside A's focus but object A is inside B's nimbus, or object B is inside A's focus but object A is outside B's nimbus (Fig. 4).
- No awareness: An object A has no awareness of object B in a medium M when object B is outside A's focus and object A is outside B's nimbus (Fig. 5).

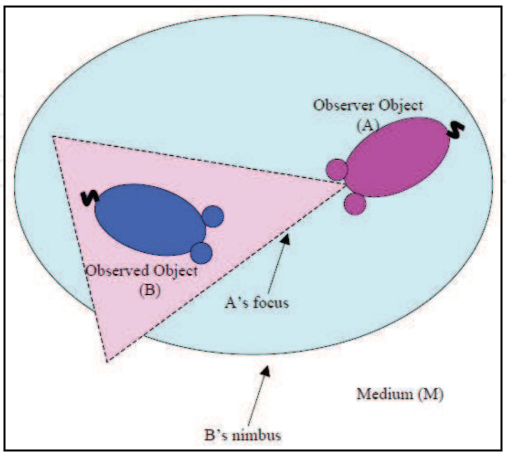


Fig. 3. Full awareness (Herrero, 2003)

In the SMI an object can control its awareness in different ways (Benford & Fahlén, 1993) by modifying its own auras, focus and nimbus:

- Implicitly: By moving and changing direction within the space and hence its aura, focus and nimbus.
- Explicitly: By directly modifying the parameters which define aura, focus and nimbus.

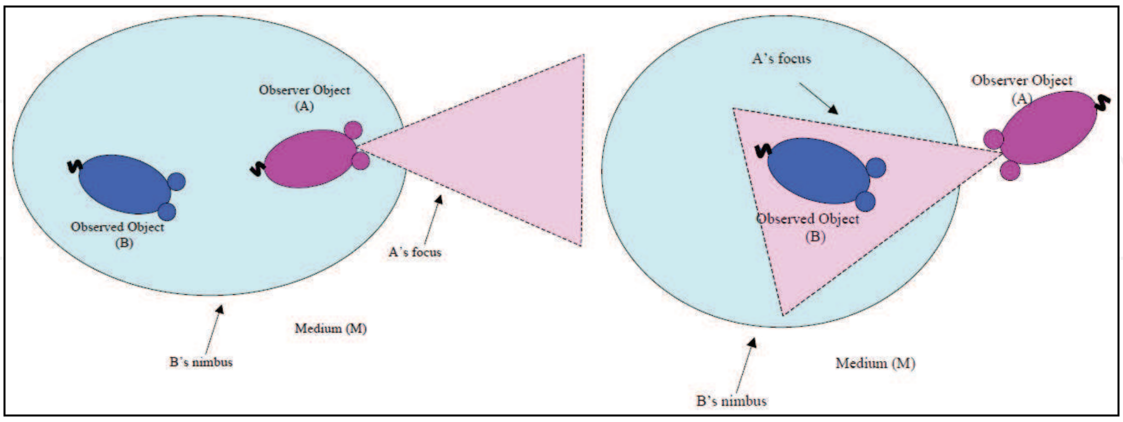


Fig. 4. Peripheral awareness (Herrero, 2003)

Additionally, aura, focus and nimbus may be manipulated through Boundaries in space. Boundaries have more importance in structuring social interaction (Bowers & Rodden, 1993). Boundaries are also a way of structuring space and influencing awareness (Bowers & Rodden, 1993). Therefore, boundaries "divide space into different areas and regions and provide mechanisms for marking territory, controlling movement, and influencing the interactional properties of space" (Benford et al, 1995). It is possible to identify several kinds of boundaries:

- Obstructive: The boundary blocks the property in question (movement, aura, focus, and nimbus).
- Conditionally obstructive: The obstruction can be removed when some condition is obeyed.
- Transforming: The boundary alters the property in some way.
- Non-obstructive: The boundary has no effect on the property.

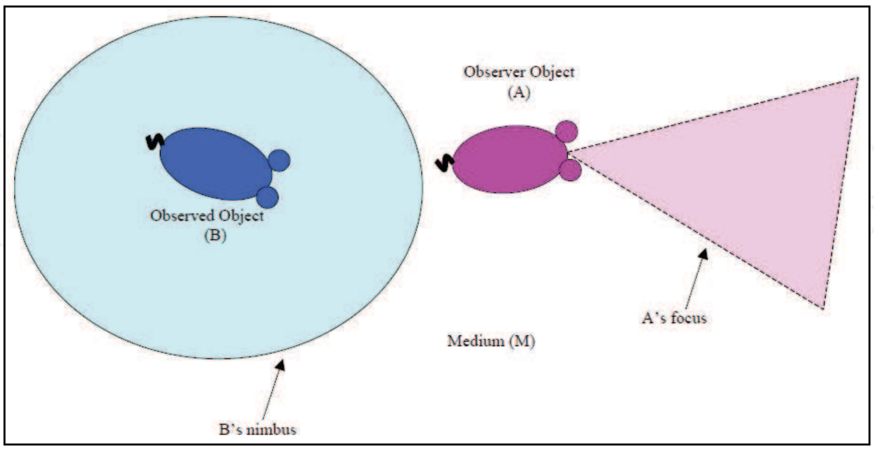


Fig. 5. No awareness (Herrero, 2003)

3.2 Awareness of interaction and artificial neural network

Based on the concepts of the SMI model previously mentioned, Herrero et al proposed an interesting adaptation related with CDEs (Herrero et al, 2007a & 2007b). Some minor changes were then proposed by Paletta & Herrero in (Paletta & Herrero, 2008 & 2009a). A distributed environment E contains a set of n nodes N_i ($1 \le i \le n$) and r different types of resources R_j ($1 \le j \le r$) that nodes can indifferently give. These resources can be shared as a collaborative mechanism among different nodes. The following concepts are defined:

1. N_i.*Focus*(R_j): It can be interpreted as the subset of the space (distributed environment) on which N_i has focused his attention aiming to interact or collaborate, according to the resource R_j.
2. N_i.*NimbusState*(R_j): Indicates the current grade of collaboration that N_i can give over R_j. It could have three possible values: *Null*, *Medium* or *Maximum*. If the current grade of collaboration given by N_i about R_j is not high, and this node could collaborate more over this resource, then N_i.*NimbusState*(R_j) will get the *Maximum* value. If the current grade of collaboration given by N_i about R_j is high but N_i could improve the collaboration over this service, then N_i.*NimbusState*(R_j) would be *Medium*. Finally, N_i.*NimbusState*(R_j) will be *Null* if N_i cannot offer R_j or if it cannot collaborate any more with this service.
3. N_i.*NimbusSpace*(R_j): Represents the subset of the distributed environment where N_i aims to establish the collaboration over R_j.
4. R_j.*AwareInt*(N_a, N_b): This concept quantifies the degree of collaboration over R_j between a pair of nodes N_a and N_b. It is manipulated via *Focus* and *NimbusSpace*, and and requires a negotiation process. Following the awareness classification introduced by Greenhalgh (Greenhalgh, 1997), values of this concept could be *Full*, *Peripheral* or *Null*. It is calculated using (1).

$$R_j.AwareInt(N_a, N_b) = \begin{cases} N_b \in N_a.Focus(R_j) \wedge N_a \in N_b.NimbusSpace(R_j),\ Full \\ (N_b \in N_a.Focus(R_j) \wedge N_a \notin N_b.NimbusSpace(R_j)) \vee \\ (N_b \notin N_a.Focus(R_j) \wedge N_a \in N_b.NimbusSpace(R_j)),\ Peripheral \\ \text{Otherwise},\ Null \end{cases} \quad (1)$$

5. N_i.*TaskResolution*(R_1,…,R_p): N_i requires collaboration with all R_j ($1 \le j \le p$) to solve a specific task T.
6. N_i.*CollaborativeScore(R_j)*: Determines the score for R_j to collaborate in N_i. It is represented with a value within [0, 1]. The closer the value is to 0 the hardest it will be for N_i to collaborate with the necessary R_j. The higher the value is (closer to 1) the completer will the willingness to collaborate be.

Trying to use an ANN to learn specific situations of the CDE, and therefore take decisions at the basis of these situations, depends on the ability to represent and inform the ANN about the current state of the CDE. Based on the fact that current CDE conditions could be represented by the concepts of N_i.*Focus*(R_j), N_i.*NimbusState*(R_j) and N_i.*NimbusSpace*(R_j), and from these concepts it is possible to obtain the corresponding R_j.*AwareInt*(N_a, N_b), it is possible to identify the following variables:

1. A value *Nst* $\in$ [0,1] representing N_a.*NimbusState*(R_j) that is further interpreted/ represented in (2).
2. A value *AwI* $\in$ [0,1] representing R_j.*AwareInt*(N_a, N_b) that is further interpreted/ represented in (3).
3. A value *Foc* that is equal to 1 if $N_b \in N_a$.*Focus*(R_j). If $N_b \notin N_a$.*Focus*(R_j) then the entry is 0.

With these variables and depending of the searched goal, it is possible to define parts of a pattern that can be used as an input in an ANN so that the ANN might learn different scenarios related with the current CDE. Some examples can be seen in the case study presented in the next section, specifically in the topic related with the heuristic-based learning strategies.

$$Nst = \begin{cases} 1, N_a.NimbusState(R_j) = Maximum \\ 0.5, N_a.NimbusState(R_j) = Medium \\ 0, N_a.NimbusState(R_j) = Null \end{cases} \tag{2}$$

$$AwI = \begin{cases} 1, R_j.AwareInt(N_a, N_b) = Full \\ 0.5, R_j.AwareInt(N_a, N_b) = Peripheral \\ 0, R_j.AwareInt(N_a, N_b) = Null \end{cases} \tag{3}$$

The next section explains in detail the AMBAR model, as a case study based on the concepts presented in this section.

4. AMBAR: A case study

AMBAR was proposed as a learning collaboration agent-based model for distributed environments endowed with heuristic-based strategies. This was done aiming to take into account the information of awareness' collaborations occurring in the environment for achieving the most appropriate future awareness situations. AMBAR is structured by the following elements (see Fig. 6):

1. The awareness representation and collaborative process.
2. An architecture (SOFIA) used for designing the intelligent agents known as IA-Awareness.
3. A negotiation mechanism to deal with saturated conditions.
4. A mutual exclusion strategy to synchronize the use of critical sections.
5. A load-balancing strategy (CAwaSA).
6. A communication protocol that allows agents to exchange messages and hence interact with each other.
7. Heuristic-based learning strategies (CAwANN).

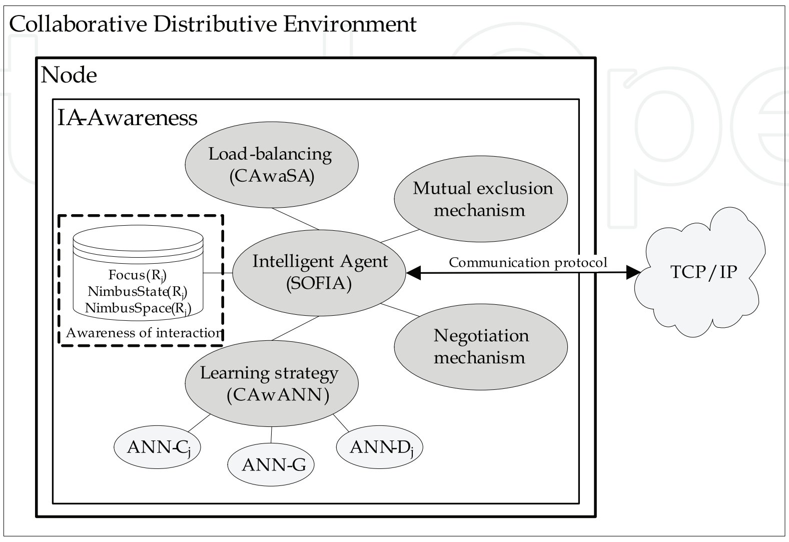


Fig. 6. The AMBAR structure

4.1 Awareness representation and collaboration process

Awareness representation is defined as was explained in Section 3.2. Any node N_a in the distributive environment is endowed with an IA-Awareness agent, that has the corresponding information about *E*, i.e.: $N_a.Focus(R_j)$, $N_a.NimbusState(R_j)$ and $N_a.NimbusSpace(R_j)$ for each R_j. The collaborative process in the system follows these steps:

1. N_b must solve a task *T* by means of a collaborative task-solving process making use of the resources $R_1,\ldots, R_p$, so that, it generates a $N_b.TaskResolution(R_1,\ldots, R_p)$.
2. N_b looks for the current conditions to calculate the values associated to the key concepts of the model (*Focus*, *NimbusState* and *NimbusSpace* related to the other nodes), given by $N_i.Focus(R_j)$, $N_i.NimbusState(R_j)$ and $N_i.NimbusSpace(R_j)$ $\forall i$, $1 \le i \le n$ and $\forall j$, $1 \le j \le r$. This information is used to decide the most suitable node with which to collaborate related with any resource R_j (by using the load-balancing strategy CAwaSA). Nodes in this environment respond to requests for information made by N_b. This is done through the exchange of messages between agents (by using the communication protocol). As a final result of this information exchange the model will calculate the current awareness levels given by $R_j.AwareInt(N_i, N_b)$ as well as the collaboration score $N_b.CollaborativeScore(R_j)$.
3. For each resource R_j ($1 \le j \le p$) included in $N_b.TaskResolution(R_1,\ldots, R_p)$, N_b selects the node N_a whose $N_a.CollaborativeScore(R_j)$ is the most suitable to start the collaborative process (greatest score). Then, N_a will be the node in which N_b should collaborate on resource R_j.
4. Once N_a receives a request for cooperation, it updates its *Nimbus* (given by $N_a.NimbusState(R_j)$ and $N_a.NimbusSpace(R_j)$). In like manner, once N_a has finished collaborating with N_b it must update its *Nimbus*.

The IA-Awareness agent, that each node in the system has, is designed to take into account the following considerations/features:

1. While each node may have different agents / processes, the IA-Awareness is the one that handles and manages the collaboration process; moreover, it learns to collaborate. In this sense, any need for cooperation from a source that is currently running on the node needs to communicate through the IA-Awareness service $TaskResolution(R_1,\ldots, R_p)$. In response to this service, IA-Awareness returns a list of *p* nodes, one for each resource R_j, better suited to collaborate with the current node in relation with the corresponding R_j.
2. There are services (abilities) that report on current levels of $Focus(R_j)$, $NimbusState(R_j)$ and $NimbusSpace(R_j)$ for a specific resource R_j.
3. Once all the necessary information is achieved, the search for the most suitable nodes to collaborate related with any R_j is done by using the service $FindSuitableNodes(R_1,\ldots, R_p)$.
4. When conditions on the environment are not appropriated enough to establish a collaboration process ($N_i.NimbusState(R_j) = Null$ for most of the N_i, R_j), the nature of the node N_b ,initiating a collaborative process to answer a $N_b.TaskResolution(R_1,\ldots, R_p)$, can lead to having no options, so that N_b can start a negotiation process that allows for N_b to identify new candidates to collaborate with. The detection of this saturated conditions is accomplished by using the service $IsOverloaded(N, R)$.
5. The initiation and completion of the collaboration associated with the resource *R* is achieved through the implementation of services $StartCollaboration(R)$ and $EndCollaboration(R)$.

4.2 The agent architecture

SOFIA (SOA-based Framework for Intelligent Agents) (Paletta & Herrero, 2009d and 2009e) is the architecture used to design the IA-Awareness agents used in AMBAR. It focuses on the design of a common framework for intelligent agents with the following characteristics: 1) it merges interdisciplinary theories, methods and approaches, 2) it is extensible and open as to be completed with new requirements and necessities, and 3) it highlights the agent´s learning processes within the environment. SOFIA's general architecture contains four main components (see Fig. 7):

1. The Embodied Agent (IA-EA) or the "body": It is a FIPA-based structure (FIPA, 2002b) because it has a Service Directory element which provides a location where specific and correspondent services' descriptions can be registered. The IA-EA encloses the set of services related to the abilities of sensing stimuli from the environment and interacting with it.
2. The Rational Agent (IA-RA) or the "brain": This component represents the agent's intelligent part and therefore, it encloses the set of services used by the agent to implement the process associated with these abilities. It is also a FIPA-based structure.
3. The Integrative/Facilitator Agent (IA-FA) or the "facilitator": It plays the role of simplifying the inclusion of new services into the system as well as the execution of each of them when it is necessary. The basic function of the IA-FA is to coordinate the integration between the IA-SV and the rest of the IA components. This integration is needed when a new service is integrated with the IA and therefore it is registered into the corresponding Service Directory, even when an existing service is executed.
4. The IA Services or "abilities" (IA-SV): It is a collection of individual and independent software components integrated to the system (the IA) which implements any specific ability either to the IA-EA or the IA-RA.

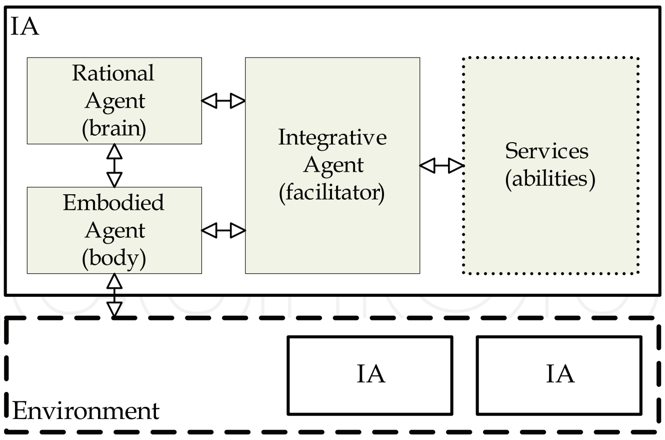


Fig. 7. The SOFIA general architecture

4.3 The negotiation mechanism

The negotiation mechanism included in AMBAR consists of three elements (Paletta & Herrero, 2010b) (see more details below in Section 4.6): 1) a heuristic algorithm used for deciding the most suitable node to initiate negotiation based on current conditions; 2) a heuristic method to accept/decline a need for collaboration during a negotiation; 3) a protocol for exchanging messages between agents.

The basic idea is to find / identify a node *N* that might will be a potential candidate to negotiate with, taking into account the possibility of making changes in its *Nimbus* in relation with the resource *R*. *N* can then collaborate with this node in relation with *R*. "Potential" means that *N* accepts i.e. the negotiation is successful.

4.4 The mutual exclusion mechanism

The nature of a node initiating a collaborative process to answer a *TaskResolution*($R_1,\ldots,R_p$), provokes a change in the conditions of the collaboration levels of the environmental nodes involved in the process. Since this information is required by the process of taking action, the levels of collaboration between the nodes turn into a critical section, so that a mutual exclusion mechanism is required. The strategy used in AMBAR is a variation of the Naimi-Tréhel's token-based algorithm (Naimi et al, 1996). In the AMBAR token-based approach (Paletta & Herrero, 2009b), the token travels with a queue Q which has the nodes that require the exclusive use of the critical section and haven't been able to satisfy that need.

4.5 The load-balancing strategy

Having a set of *p* resources R_j ($1 \leq j \leq p$). For each resource a particular node must identify the most suitable other node in the environment with which to collaborate according to the corresponding resource. "Most suitable" means that it should consider the following assumptions:

1. The node N_b that seeks collaboration should be on the *Focus* of the node N_a that needs to be identified, i.e. $N_b \in N_a.Focus(R_j)$.
2. The score of collaboration given by $N_a.CollaborativeScore(R_j)$ must indicate the full readiness to collaborate on R_j (value equal or close to 1).
3. The selection must be done so that there will be a load-balancing process distributed equally among all possible nodes with which to collaborate. This should take into account the current environment conditions given by $N_i.NimbusState(R_j)$ and $N_i.NimbusSpace(R_j)$ $\forall i,j$ $1 \leq i \leq n$, $1 \leq j \leq r$.
4. The answer must be given in a dynamic way and in a reasonable amount of time (preferably at the same time as the request is generated).

The strategy used to solve this problem is based in the Simulated Annealing technique (Kirkpatrick, 1984) (Metropolis et al, 1953) which is a generalization of a Monte Carlo method that searches for a minimum in a more general system forming the basis of an optimization technique to solve combinatorial and other problems. This strategy is called CAwaSA (Collaborative Distributive Environment by means of Awareness and SA) and its results can be found in (Paletta & Herrero, 2009c).

4.6 The communication protocol

Messages for AMBAR-based agent interaction are defined according to the FIPA performative (FIPA, 2002a) and used for: 1) querying the current conditions of each node in the environment given by its Focus/Nimbus; 2) performing the mutual exclusion mechanism; 3) performing the negotiation mechanism; and 4) informing the initiation and completion of the collaboration associated with a particular resource. There are a total of ten different messages.

4.7 The heuristic-based learning strategies

An IA-Awareness has learning abilities and in AMBAR the element that has these abilities are known as CAwANN. This strategy combines Neural-Gas (NGAS) (Martinetz & Schulten, 1991), Radial Based Function Network (RBFN) (Lingireddy & Ormsbee, 1998) (Shahsavand & Ahmadpour, 2005) and Multi-Layer Perceptron (MLP) (Haykin, 1998) ANN-based models aiming to cover different aspects in the learning capabilities of AMBAR: 1) a supervised-based method for learning for learning how to collaborate based on levels of awareness; 2) an unsupervised-based method for selecting a potential candidate to negotiate on saturated conditions; and 3) a supervised-based method to learn the decision whether or not a node must change the information that describes its current conditions related with collaboration.

Just as a quick reminder, NGAS is a Vector Quantization (VQ) (Kohonen et al, 1984) (Makhoul et al, 1985) (Nasrabadi and Feng, 1988) (Nasrabadi & King, 1988) (Naylor & Li, 1988) technique with soft competition between the units. VQ is the process of quantizing n-dimensional input vectors to a limited set of n-dimensional output vectors usually generated by clustering a given set of training vectors. The goal of clustering is to reduce large amounts of raw data by categorizing it in smaller sets of similar items. On the other hand, radial based functions were originally developed to discuss problems involving the adaptation of irregular topographic contours through a series of geographic data. ANN's based on this technique (RBFNs) are among the best choices in models out there as an alternative to achieve excellent results in alignment of data caused either by stochastic or deterministic functions (Jin et al, 2001). Finally, MLP is one of the most used neural models for implementing a variety of problems.

4.7.1 Learning levels of collaboration

This process is about learning the association between the current status of the environment, given by $N_i.Focus(R_j)$, $N_i.NimbusState(R_j)$ and $N_i.NimbusSpace(R_j)$, and the levels of collaboration obtained from that specific situation, given by the $N_i.CollaborativeScore(R_j)$ ($\forall i$, $1 \leq i \leq n$; $\forall j$, $1 \leq j \leq r$).

The ANN used in this solution, called ANN-C, has 2 inputs and 1 output (see Fig. 8). The output relates to the learned value of $N_a.CollaborativeScore(R_j)$. The inputs correspond to the following items:

1. *Nst* as is indicated in Section 3.2.
2. *AwI* as is indicated in Section 3.2.

To differentiate one resource from another, given the fact that each node can have a different treatment in the levels of collaboration, the IA-Awareness has a different ANN-C_j element for each resource R_j. This is an important aspect because:

1. Each resource can be trained separately from the rest.
2. The training process is less complex and, therefore, it is expected to obtain a higher quality in the response given by each ANN-C_j.
3. The model is expansible because new ANNs can be added when a new resource has to be incorporated into the environment.
4. Each node has a particular set of ANN-C_j, i.e. IA-Awareness of each node trains and uses certain ANNs according to the particular handling a node wants to give to each resource R_j, making the collaboration model more flexible.

Since each ANN-C_j has two inputs, each of them with three possible values, there is a total of nine possible patterns for training by combining the *Nst* values (0, 0.5, and 1) with the

AwI values (0, 0.5, and 1). Moreover, this strategy is implemented by using the MLP and RBFN models (see Section 5 for details). The basic idea is to train both ANNs and each time the ANNs are consulted about a specific situation, one of them will choose from the two possible responses taking that which originates from the ANN that has achieved a minor error in the training process (it is represented with a "?" in Fig. 8). The ANN-C_j training is performed automatically after new patterns have been stored on the IA-Awareness agent.

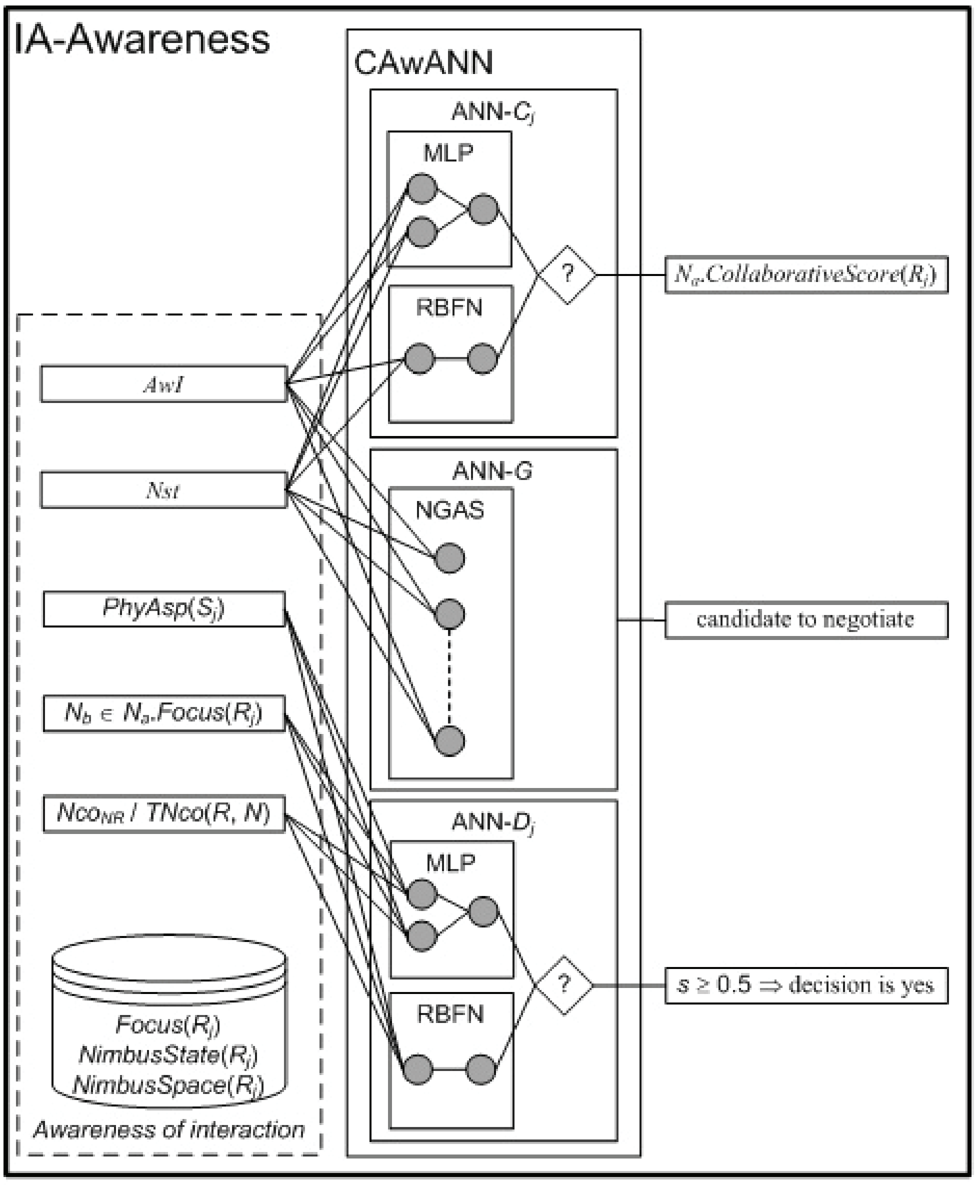


Fig. 8. CAwANN: The AMBAR learning strategy

4.7.2 Learning saturated conditions

The objective in this part of the process is to find / identify a node *N* that might be a potential candidate to negotiate with, taking into account the possibility to make changes in its *NimbusState* and *NimbusSpace* in relation with the resource *R*. *N* can then collaborate with the identified *N* node in relation with *R*. "Potential" means that the negotiation is successful, i.e. that *N* accepts.

To achieve this goal a competitive-learning-based strategy was defined aiming to correlate current information of the nodes in the distributive environment based on clusters. Therefore a NGAS-based algorithm is used. The decision that the node must make consists on identifying the node closest to the hyper-plane defined by the space given by the current environment conditions. In other words, it is necessary to determine the winning unit by testing the NGAS with the environment. The goal of this learning process is 1) to cluster the input data into a set of partitions such as the intra-cluster variance which remains small compared with the inter-cluster variance, and 2) to estimate the probability of the density function. This clustering scheme seems possible as we expect a strong correlation among the awareness information involved.

In CAwANN the NGAS-based ANN used is identified as ANN-*G*. The input vector is defined as the same as the ANN-C_j (being *Nb* the node that requires collaboration on a set of services and therefore the one that sends the $N_b.TaskResolution(R_1,\ldots,R_p)$, for each $N_a \neq N_b$):

1. *Nst* as is indicated in Section 3.2.
2. *AwI* as is indicated in Section 3.2.

Therefore, the input vectors for this problem have 2*n* elements, being *n* the number of nodes in the environment. If $N_a = N_b$ then $Nst = AwI = 0$. Patterns for learning are obtained either by those scenarios that have been stored during the dynamics of the distributed environment, or by an automatic generation, mostly random.

4.7.3 Learning the decision to alter the current condition

The latter case of ANN-based learning strategy is similar to that used with the ANN-C_j. In this case it has one ANN-D_j for each resource R_j. There are three inputs and one output. The output $s \in [0, 1]$ represents the decision i.e. it is accepted if $s \geq 0.5$, and declined otherwise. Inputs are as follows:

1. A value $PhyAsp(S_j) \in [0, 1]$ that indicates the level of physical availability of the resource R_j. This physical aspect is related to the current conditions of the resource and the node's physical ability to actually be able to collaborate on the basis of this resource. For example, the maximum size of a service requests queue, the physical feature of a hardware related with the resource (CPU, memory, communications ports, and others), among others.
2. A value equal to 1 if $N_b \in N_a.Focus(R_j)$, being N_b the node that requires the decision, and N_a the node that should make the decision. If $N_b \notin N_a.Focus(R_j)$ then the entry is 0.
3. A value equal to *NcoNR* / *TNco*(*R*, *N*) which represents a logical aspect that deals with the relationship that N_b might had had in the past regarding a collaboration process. The idea is to reward those nodes N_b who collaborated in the past with N_a and are now requiring collaboration with N_a. *NcoNR* is the number of times a node *N* (node that requires the decision) has collaborated with the current node (node that should make the decision) related to resource *R*. *TNco*(*R*, *N*) is calculated by following (4).

$$TNco(R,N) = \begin{cases} \sum_{j=1}^{r} Nco_{Nj}, & \sum_{j=1}^{r} Nco_{Nj} \neq 0 \\ \text{random}(Nco_{NR},1), & \text{otherwise} \end{cases} \tag{4}$$

To obtain the information related to the physical aspect associated with the resource, IA-Awareness has the ability to query the corresponding current values. It is represented by a

value between [0, 1] which expresses the percentage of current use of the node in relation to *R*. A value equal to 1 which means that *R* is being used at its maximum capacity (it is saturated).
Furthermore, each node *N* keeps the nxr matrix *Nco* whose elements Nco_{ij} matches the number of times the node N_i ($1 \le i \le n$) has collaborated with *N* in relation to the resource R_j ($1 \le j \le r$). This information as well as $N.Focus(R_j)$ are used to calculate the logical aspects needed for taking a decision.
The training of the ANN-D_j is performed automatically after a new pattern has been stored on the IA-Awareness agent. An ANN-D_j is considered trained when a proper number of patterns has been stored and used. As happens with the ANN-C_j, the ANN-D_j are trained and used by using both MLPs and RBFNs strategies.

4.7.4 Evaluation

In summary, and as it can be seen in Fig. 8, CAwANN strategy is the combination of the ANN-C_j ($1 \le j \le r$), the ANN-*G*, and the ANN-D_j ($1 \le j \le r$) previously explained. It was implemented using Java. Its evaluation was conducted in a TCP/IP-based LAN (Local Area Network) which assumes that each node (PC) can directly communicate with any other node. The experimentation was conducted by simulating different scenarios aiming to rate the capability of the method used for managing the growth of the nodes in the different conditions of the environment. The scenarios were defined by changing the quantity of nodes/PCs *n* (agents) as well as the number of resources *r* according to $n \in \{4, 8\}$ and $r \in \{2, 6, 10\}$. Therefore 6 different scenarios were simulated: 1) $n = 4$, $r = 2$; 2) $n = 4$, $r = 6$; 3) $n = 4$, $r = 10$; 4) $n = 8$, $r = 2$; 5) $n = 8$, $r = 6$; and 6) $n = 8$, $r = 10$. Moreover:

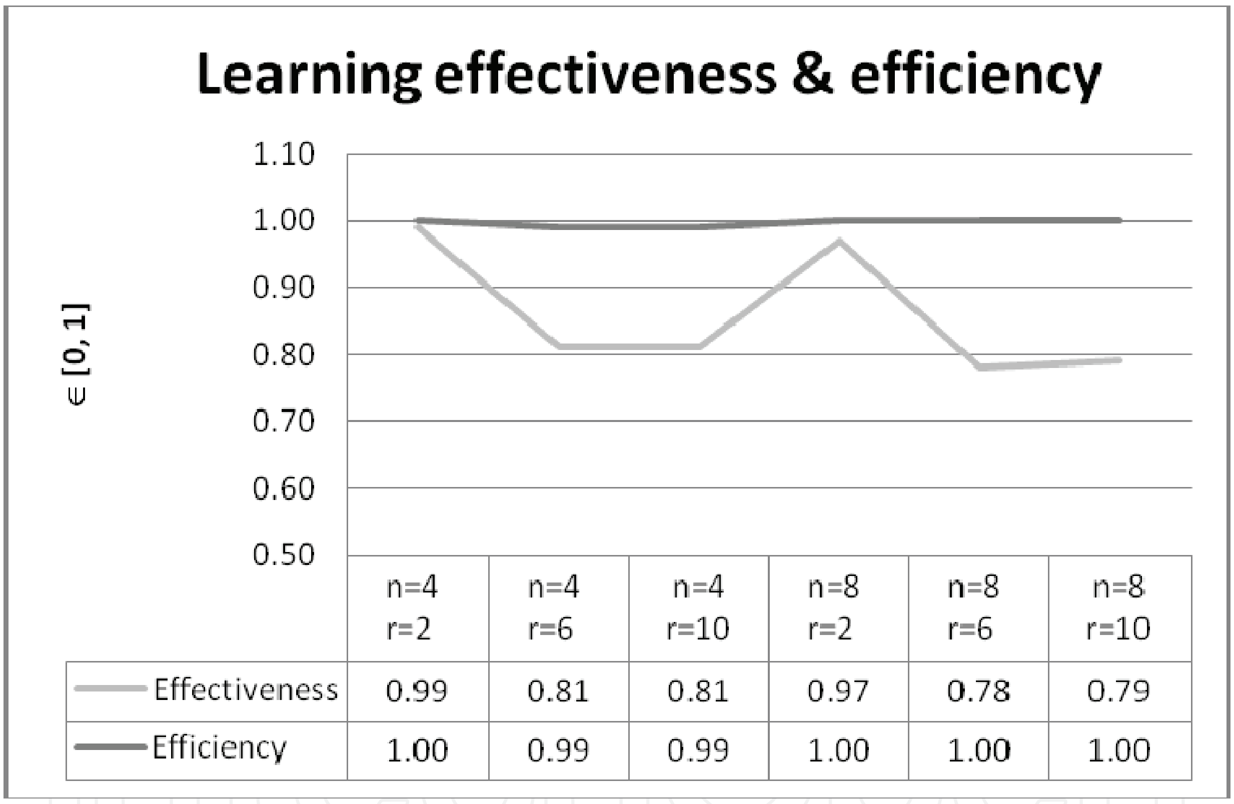
1. The hardware platform of the PCs was the same for all the nodes: Intel T2600 (2.16 GHz) with 2 GB RAM.
2. The initial condition of the distributed environment for each scenario ($N_i.Focus(R_j)$, $N_i.NimbusState(R_j)$ and $N_i.NimbusSpace(R_j)$; $1 \le i \le n$; $1 \le j \le r$) was randomly defined by considering the following: one node belongs to the *Focus* of another node with a probability of 0.75 and to the *Nimbus* with a probability of 0.85.
3. All N_b nodes execute an automatic process that generates $N_b.TaskResolution(R_1,\ldots,R_p)$ by randomly selecting the involved resources from the 50% of the total resources in the scenario.
4. $PhyAsp(R_j)$, $\forall j\ 1 \le j \le r$ were randomly initialized.
5. The parameters used for configuring the NGAS-based ANNs (i.e. the ANN-G) are the following: $\varepsilon(0) = 1.58$; $\varepsilon(T) = 0.02$; $\rho(0) = 5.59$; $\rho(T) = 0.07$. In fact, a genetic program was used to find the best configuration to deal with this problem. The network was trained with T = 40.000 signals.
6. The MLP-based ANNs were configured as follows: the transfer function used is the sigmoid; additional to the input and output units the topology has one hidden layer with two units; the learning rate is equal to 0.125; momentum (Phansalkar & Sastry, 1994) is used with a rate equal to 0.9.
7. The RBFN-based ANNs were configured by using a genetic program to find the best configuration to deal with this problem. Especially regard to the range of initialization of the connections' weights and the learning factors. Only one hidden unit is used.

Aiming to measure the effectiveness (θ) and efficiency (ξ) of the learning strategy, expressions (5) and (6) were defined respectively. Note that both measures (θ, ξ) are positive values in [0, 1] where 1 is the maximum effectiveness and efficiency. Where:

- PSC: is the percentage of successful collaborations based on the number of resources in which there was positive response from a node to collaborate with, in relation to the total quantity of resources in which collaboration was required.
- PSN: is the percentage of successful negotiations made in saturated conditions, based on the number of negotiations that receive a positive response from a node requesting to change its current saturated conditions in relation to the total attempts made.
- ATL: is the mean duration in seconds of the learning process.
- ATC: is the average time of collaboration in seconds calculated since *TaskResolution*($R_1,\ldots,R_p$) starts until it ends.

$$\theta = (PSC + PSN) / 200 \tag{5}$$

$$\xi = 1 - ATL / ATC \tag{6}$$



	n=4 r=2	n=4 r=6	n=4 r=10	n=8 r=2	n=8 r=6	n=8 r=10
Effectiveness	0.99	0.81	0.81	0.97	0.78	0.79
Efficiency	1.00	0.99	0.99	1.00	1.00	1.00

Fig. 9. Effectiveness and efficiency of CAwANN obtained from experimentation

Table 1 shows the measures obtained after a simulation of 120 minutes for each scenario, and Fig. 9 shows the effectiveness and efficiency related with these measures. According to these results it is possible to make the following observations and/or conclusions:

1. The efficiency remains stable at a high value. Therefore the learning process shows to be faster.
2. Both effectiveness and efficiency have a similar trend of behavior.
3. The variation in the number of nodes hasn´t a particular tendency to improve or worsen the effectiveness.
4. The variation in the number of resources has a tendency to undermine the effectiveness.
5. The average effectiveness is 0.86 and the average efficiency is 1.00.

Measure	*n*=4 *r*=2	*n*=4 *r*=6	*n*=4 *r*=10	*n*=8 *r*=2	*n*=8 *r*=6	*n*=8 *r*=10
PSN	100.00	68.13	64.29	93.75	78.13	100.00
PSC	98.96	94.24	97.70	99.89	78.08	57.90
ATT	2.47	5.86	9.58	6.73	167.38	445.15
ATL	0.01	0.02	0.02	0.01	0.03	0.02
θ	0.99	0.81	0.81	0.97	0.78	0.79
ξ	1.00	0.99	0.99	1.00	1.00	1.00

Table 1. Measures obtained from experimentation

It is important to stress that, due to the fact that it is a learning-based mechanism from past situations, it is assumed that, as there is much more to learn, the metrics associated with it must be improved.

4.8 An example

This section follows an example of using AMBAR, and therefore CAwANN in real applications. This example is related with prediction of banking fraud. It represents a distribute system in which financial and national security institutions intervene with the purpose of detecting possible bank frauds during a financial transaction. The main objective of this application is to uncover bank frauds before these occur. This process benefits certain financial institutions, and their respective clients, that are related to transactions that involve money withdrawals and can be susceptible to fraud. For example, assuming that the cashing of a fraudulent check inside a bank when the information presented, included the signature, is false or non valid, goes undetected before the cashing of the check then there will be a loss of money that can put the financial institution and its clients in jeopardy.

According to the experience suffered by financial institutions located in countries where this type of fraud is common, the majority of these fraudulent cases are carried out by the same group of thieves. If this is true then it is possible to try to uncover the fraud through the identification of people that has committed this type of fraud before. It is also known that in the majority of cases thieves won't come back to the same financial institution that was robbed so as not to be identified by the employees that work in the institution or by the surveillance systems employed.

In the same order of ideas, recently people involved in frauds is identified automatically by the use of biometrical techniques and the existence of data bases that associate this person with any other biometrical structure like finger prints, facial features and signature just to name a few. These data bases can be used from public offices of national security (such as police departments and other departments related) and financial institutions that have adopted biometrical techniques as part of their technological solution for banking fraud. By using biometrical techniques this institutions hold biometrical information of all of the financial institution's clients and of the people that, at one time, committed fraud in the institution. With this in mind the following inquires are presented:

- How to have online access to the data bases of national public offices such as police departments?
- How can different financial institutions that cannot stand the biometric technology benefit themselves?

- How different financial institutions can share data bases so that thieves that are registered and identified in a certain community can be easily identified trying to commit the same crime in another community?

To solve these questions and to satisfy the main objective of the problem previously described a distributive collaborative environment is developed designed to be used by financial institutions and national public offices. For this, it is possible to use AMBAR. The collaborative distributive environment *E* is formed by the nodes N_i that represent the following items:

1. Financial institutions that use biometric technology.
2. Communities that do not possess biometric technology.
3. National and State Public Offices that are in possession of data bases of current thieves.
4. Public institutions that do not manage biometric technology but require it as an activity to satisfy particular objectives.

The resources R_j are equivalent to their own purposes for the usage of the biometrical technology (enroll, identify and certify) that are associated to each one of the technologies (finger print, face and signature). Related to this the next 9 resources are encountered:

- R_1, R_2, R_3: enroll, identify and certify a finger print respectively.
- R_4, R_5, R_6: enroll, identify and certify a face respectively.
- R_7, R_8, R_9: enroll, identify and certify a signature respectively.

It is interesting to mention that if another biometric metric, different from finger print, face or signature, existed it is only necessary to add to the collaborative distributive environment new resources that are able to support the different abilities for enrolling, identifying and certifying the new metrics.

Within the same order of ideas, when it is said that $N_3.\mathit{Focus}(R_2) = \{N_1, N_4\}$ this is really indicating that when the system associated to N_3 has the necessity of identifying a finger print (R_2), N_3 could then ask N_1 and N_4 for collaboration. N_1 and N_4 are supposed to be nodes that possess data bases of finger prints and the abilities to achieve R_2. The expression $N_4.\mathit{NimbusState}(R_2) = \mathit{Medium}$ indicates, not only that N_4 possesses the ability to make R_2, but also, in that specific moment, N_4 is already collaborating in the making of this ability and is able to make it for longer. Speaking of $N_4.\mathit{NimbusSpace}(R_2) = \{N_3\}$ it is essential to make clear that in that specific moment N_4 is working on a process of identification of finger prints (R_2) due to a request of collaboration made by N_3.

Learning capabilities with CAwANN in this particular example means:

1. To learn the better options to ask for collaboration (enroll, identify and certify a finger print, a face, or a signature) needed for doing financial transactions. “Better” in this case means trying to detect and avoid a possible fraud.
2. To learn about better options for selecting a potential candidate to negotiate on saturated conditions and try to avoid delays in the completion of the financial transaction.
3. To learn the decision of whether or not a node (financial institution, society, public institution or office) must change the information that describes its current conditions related with collaboration.

5. Comments related to the ANN

This section was developed aiming to present some aspects related to the way in which the ANN was used in the case study presented in the previous section. One of those aspects deals with the use of both models MLP and RBFN in those cases where a supervised ANN is required. Since not all problems can be treated the same way there is always the question: "what model should be used to treat a particular problem?" the advantage of using both models is to exploit the benefits of both of them to obtain the best results.

The idea related to the previous subject is very simple: There are two ANNs, one of them related to MLP and the other to RBFN. Both ANNs are designed so that the inputs and outputs are the same. Both ANNs are trained at the same time and using the same patterns and the last training error obtained is saved. When an answer is required to a specific situation, both ANNs are consulted with the same input and only that ANN whose last training error is smaller is chosen as the right output.

On the other hand, depending on the problem and specifically on the number of different input patterns, the ANNs can be trained automatically by counting the number of situations that have occurred in the environment related to that problem, every new situation corresponds to a new pattern to be considered for the following training. When the count reached a predetermined number then training is working automatically. For supervised ANN a non ANN-based alternative to obtain the answer it is required to define the training pattern completely.

Finally, and due to the fact that CDE are dynamically continually changing, having multiple small ANNs is better than having only a larger ANN. As a first consequence, the training process is less complex and, therefore, it is expected to obtain a higher quality response. Moreover, changing on the environment can be handled properly by adding / removing the necessary o corresponding ANNs.

6. Conclusion

Collaborative Distributed Environments (CDEs) and Artificial Neural Networks (ANNs) can relate to each order to improve either the ANN performance or any other process relative to the distributed environment. In this regard, theoretical aspects of the Spatial Model of Interaction (SMI), and particularly in the awareness concept are used in this chapter to define an ANN-based learning strategy aiming to improve cooperation/collaboration in CDEs. In fact, the awareness concept quantifies the degree of collaboration needed or occurring over a particular resource between a pair of nodes in the environment. This information, as well as the information related with the CDE current condition, can be used as the input of any ANN defined to deal with a particular situation or problem in the CDE.

As a case study, the chapter presents some details of AMBAR (Awareness-based learning Model for distriButive collAborative enviRonment), and more specifically, the particular element called CAwANN (Collaborative Distributed Environment by means of an Awareness & Artificial Neural Network strategy), which has been designed with the objective to learn from different types of awareness as well as from previous collaborations that were carried out in the environment to foresee future collaborative/cooperative scenarios. The results obtained show that the learning strategy has an average efficiency of 100% and an average effectiveness of 86%.

7. References

Benford, S.D. & Fahlén, L.E. (1992) Focus, Aura and Awareness, *Proceedings of 5th Multi-G Workshop*, KTH, Stockholm, Sweden, December 1992.

Benford, S.D. & Fahlén, L.E. (1993). A Spatial Model of Interaction in Large Virtual Environments, *Proceedings of 3rd European Conference on Computer Supported Cooperative Work*, Kluwer Academic Publishers, pp. 109-124.

Benford, S.D.; Bowers, J.M.; Fahlén, L.E.; Mariani, J. & Rodden, T.R. (1994). Supporting Co-operative Work in Virtual Environments, *The Computer Journal*, Vol. 37, No. 8, Oxford University Press, pp. 653-668.

Benford, S.; Bowers, J.; Fahlen, L.; Grrenhalgh C. & Snowdon D. (1995). User Embodiment in Collaborative Virtual Environments. *Proceedings of ACM Conference on Human Factors in Computing Systems* (CHI'95), Denver, Colorado, USA, pp. 242-249.

Blanchard, E. & Frasson C. (2002) Designing a Multi-agent Architecture based on Clustering for Collaborative Learning Sessions, *Proceedings of International Conference in Intelligent Tutoring Systems*, LNCS, Springer Verlag, Biarritz, France.

Bowers, J. & Rodden, T. (1993). Exploding the Interface: Experiences of a CSCW Network *Proceedings of Conference on Human Factors in Computing Systems: INTERACT '93 and CHI '93*, pp. 255-262, Amsterdam, The Netherlands, April 1993.

Calbert, D. & Guan, J. (2005). Distributed Artificial Neural Network Architectures. *Proceedings of the 19th International Symposium on High Performance Computing Systems and Applications*, IEEE Computer Society, Washington, DC, USA, pp. 2-10.

Cristea, P. & Florea, A. (1999). Artificial Intelligence and Neural Network Tools for Cooperative Learning., *Proceedings of International Workshop on Interactive Computer Aided Learning – Tools and Application* (ICL99), Villach, Austria,

Fahlen, L. & Brown, C. (1992). The Use of a 3D Aura Metaphor for Computer Based Conferencing and Teleworking, *Proceedings of the 4th Multi-G Workshop*, Stockholm, pp. 69-74.

FIPA (2002a) FIPA ACL Message Structure Specification, *Foundation for Intelligent Physical Agents*, SC00061, Geneva, Switzerland. Available on "http://www.fipa.org/specs/fipa00061/index.html"

FIPA (2002b) FIPA Abstract Architecture Specification, *Foundation for Intelligent Physical Agents*, SC00001, Geneva, Switzerland. Available on "http://www.fipa.org/specs/fipa00001/index.html"

Garcia, N.; Hervás, C. & Muñoz, J. (2002). Multi-objective cooperative coevolution of artificial neural networks (multi-objective cooperative networks), *Neural Networks*, Vol. 15, No. 10, Elsevier Science Ltd, pp. 1259-1278.

Garcia, N.; Hervás, C. & Ortiz, D. (2005). Cooperative coevolution of artificial neural network ensembles for pattern classification, *IEEE Transactions on Evolutionary Computation*, Vol. 9, No. 3, pp.271-302.

Greenhalgh, C. (1994). An Experimental Implementation of the Spatial Model, *Comic working paper COMIC-NOTT-4-15*, Department of Computer Science, The University of Nottingham, pp. 53-71.

Greenhalgh, C. (1997). Large Scale Collaborative Virtual Environments, *Doctoral Thesis, University of Nottingham.*

Gutwin, C.; Stark, G. & Greenberg, S. (1995). Support for Workspace Awareness in Educational Groupware, *Proceedings of Computer Supported Collaborative Learning* (CSCL '95), pp. 147-156.

Haykin, S. (1998). Neural Networks: A Comprehensive Foundation; *Prentice Hall*; ISBN: 0-132-73350-1.

Herrero, M.P. (2003). A Human-Like Perceptual Model for Intelligent Virtual Agents, *Doctoral Thesis, Universidad Politécnica de Madrid*.

Herrero, M.P.; Bosque, J.L. & Pérez, M.S. (2007a). Managing dynamic virtual organizations to get effective cooperation in collaborative grid environments, *Proceedings of On the Move to Meaningful Internet Systems 2007*: OTM 2007 Workshops, Lecture Notes in Computer Science, Vol. 4804, Springer, pp. 1435-1452.

Herrero, M.P.; Bosque, J.L. & Pérez, M.S. (2007b). An agents-based cooperative awareness model to cover load balancing delivery in grid environments, *Proceedings of On the Move to Meaningful Internet Systems 2007*: OTM 2007 Workshops, Lecture Notes in Computer Science, Vol. 4805, Springer, pp. 64-74.

Jin, R.; Chen, W. & Simpson, T.W. (2001). Comparative studies of metamodelling techniques under multiple modeling criteria. *Struct Multidiscip Optim*, pp. 1-13.

Kiran, K. (2009). Parallelized Backpropagation Neural Network Algorithm using Distributed System. Master Thesis, *Computer Science and Engineering Department Thapar University*, Patiala, 147004.

Kirkpatrick, S. (1984). Optimization by simulated annealing: Quantitative Studies, *Statistical Phy*, Vol. 34, No. 5-6, pp. 975-986.

Kohonen, T.; Mäkisara, K. & Saramäki, T. (1984). Phonotopic maps- insightful representation of phonological features for speech recognition. *Proceeding of 7th International Conference on Pattern Recognition*, pp. 182-185.

Kuwana, E. & Horikawa, K. (1995). Coordination Process Model - based Design for Synchronous Group Task Support System, *Technical Reports of Information Processing Society of Japan*, Groupware, No. 13, pp. 1-6.

Lingireddy, S. & Ormsbee, L.E. (1998). Neural networks in optimal calibration of water distribution systems. *Artificial Neural Networks for Civil Engineers: Advanced Features and Applications*, pp. 53-76.

Liu, K.; Kargupta, H. & Ryan, J. (2006). Distributed data mining bibliography. Release, pp: 1-7. http://www.biostat.wustl.edu/archives/html/s-news/2003-09/msg00165.html.

Makhoul, J.; Roucos, S. & Gish, H. (1985). Vector quantization in speech coding. *Proceeding of IEEE 73*; IEEE Computer Society, pp. 1551-1588.

Malone, T.W & Crowston, K. (1994). The interdisciplinary study of coordination, *ACM Computing Surveys*, Vol. 26, No. 1, pp. 87-119.

Martinetz, T.M. & Schulten, K.J. (1991). A neural gas network learns topologies. In: Kohonen T, Mäkisara K, Simula O, Kangas J (Eds.). *Artificial Neural Networks*, pp. 397-402.

Matsushita, Y. & Okada, K. (Ed.) (1995). Collaboration and Communication, Distributed collaborative media series 3, Kyoritsu Press.

Mayrhofer, R. & Radi, H. (2007). Extending the Growing Neural Gas Classifier for Context Recognition, *Proceedings of Computer Aided Systems Theory* – EUROCAST'07, pp. 920-927.

Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller A.H. & Teller E. (1953). Equations of state calculations by fast computing machines, *J Chem Phy*, Vol. 21, No. 6, pp. 1087-1091.

Naimi, M.; Trehel, M. & Arnold, A. (1996). A log (N) distributed mutual exclusion algorithm based on path reversal, *Parallel Distributed Computing*, Vol. 34, No. 1, pp. 1-13.

Nasrabadi, N.M. & Feng, Y. (1988). Vector quantization of images based upon the Kohonen self-organizing feature maps. *Proceedings of IEEE International Conference on Neural Networks*; IEEE Computer Society, pp. 1101-1108.

Nasrabadi, N.M. & King, R.A. (1988). Image coding using vector quantization: A review. *IEEE Trans Comm*, Vol. 36, No. 8, pp. 957-971.

Naylor, J. & Li, K.P. (1988). Analysis of a Neural Network Algorithm for vector quantization of speech parameters; *Proceedings of First Annual INNS Meeting*, NY; Pergamon Press, pp. 310-315.

Newman, H.B. & Legrand, I.C. (2000). A self-organizing neural network for job scheduling in distributed systems, *Proceedings of Advanced Computing and Analysis Techniques in Physics Research: VII International Workshop* (ACAT 2000). AIP Conference Proceedings, Vol. 583, pp. 113-115.

Ogata, H. & Yano, Y. (1998). Knowledge Awareness: Bridging Learners in a Collaborative Learning Environment, *International Journal of Educational Telecommunications*, Vol. 4, No. 2, pp. 219-236.

Paletta, M. & Herrero, M.P. (2008). Learning cooperation in collaborative grid environments to improve cover load balancing delivery, *Proceedings of IEEE/WIC/ACM Joint Conferences on Web Intelligence and Intelligent Agent Technology*, IEEE Computer Society E3496, pp. 399-402.

Paletta, M. & Herrero, M.P. (2009a). Foreseeing cooperation behaviors in collaborative grid environments, *Proceedings of 7th International Conference on Practical Applications of Agents and Multi-Agent Systems* (PAAMS'09), Vol. 50, Springer, pp. 120-129.

Paletta, M. & Herrero, M.P. (2009b). A Token-Based Mutual Exclusion Approach to Improve Collaboration in Distributed Environments, *Proceedings of 1st International Conference on Computational Collective Intelligence – Semantic Web, Social Networks & Multiagent Systems* (ICCCI 2009), N.T. Nguyen, R. Kowalczyk, S.M. Chen (Eds.); Lecture Notes in Artificial Intelligence Vol. 5796, Springer, pp. 118-127.

Paletta, M. & Herrero, M.P. (2009c). An awareness-based simulated annealing method to cover dynamic load-balancing in collaborative distributed environments, *Proceedings of 2009 IEEE/WIC/ACM International Conference on Intelligence Agent Technology* (IAT 2009); R. Baeza-Yates et al (Eds.), IEEE Computer Society, pp. 371-374.

Paletta, M. & Herrero, M.P. (2009d). Towards fraud detection support using grid technology, *Multiagent and Grid Systems - An International Journal 5*, IOS Press, pp. 311-324.

Paletta, M. & Herrero, M.P. (2009e). Awareness-based learning model to improve cooperation in collaborative distributed environments, *Proceedings of 3rd International KES Symposium on Agents and Multi-agents Systems Technologies and Applications* (KES-AMSTA 2009), A. Håkansson et al. (Eds.), Lecture Notes in Artificial Intelligence, Vol. 5559, Springer, pp. 793-802.

Paletta, M. & Herrero, M.P. (2009f). An awareness-based artificial neural network for cooperative distributed environments, *Proceedings of International Work Conference on Artificial Neural Networks* (IWANN 2009), J. Cabestany et al (Eds.), Lecture Notes in Computer Science Vol. 5517, Springer, pp. 114-121.

Paletta, M. & Herrero, M.P. (2010a). Collaboration in Distributed Systems by means of an Awareness-based Learning Model, *Recent Patents on Computer Science* (CSENG), Vol. 3, No. 2, Bentham Science Publishers, pp. 1-21.

Paletta, M. & Herrero, M.P. (2010b). A MAS-based Negotiation Mechanism to deal with saturated conditions in Distributed Environments, *Proceedings of Second International Conference on Agents and Artificial Intelligence* (ICAART 2010), J. Filipe, A. Fred, B. Sharp (Eds.), INSTICC Press, Vol. 2, pp. 253-256.

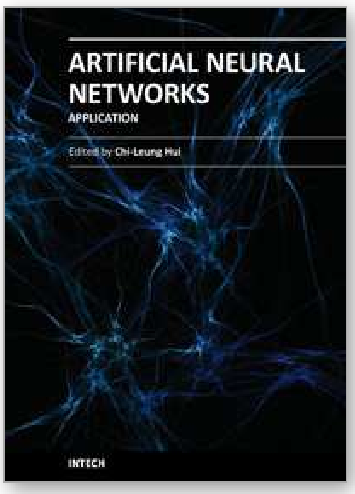
Phansalkar, V.V. & Sastry, P.S. (1994). Analysis of the Back-Propagation Algorithm with Momentum, *IEEE Transactions on Neural Networks*, Vol. 5, No. 3, pp. 505-506.

Shahsavand, A. & Ahmadpour, A. (2005). Application of Optimal RBF neural networks for optimization and characterization of porous arterials. *Comput Chem Engineering*, pp. 2134-2143.

Shelestov, A.; Skakun, S. & Kussul O. (2007). Complex Neural Network Model of User Behavior in Distributed Systems. *Proceedings of XIII-th International Conference Knowledge-Dialogue-Solutions*, Varna, Bulgaria, pp. 42-49.

Wesley-Smith, I. (2006). A Parallel Artificial Neural Network Implementation. *Proceedings of The National Conference On Undergraduate Research* (NCUR), Asheville, North Carolina, USA.

Yildiz, A. (2006). Resource-Aware Load Balancing System with Artificial Neural Networks. Master Thesis, *The Graduate School of Natural and Applied Sciences of Middle East Technical University*, Istanbul, Turkey.



ISBN 978-953-307-188-6
Hard cover, 586 pages
Publisher InTech
Published online 11, April, 2011
Published in print edition April, 2011

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821