Hybrid Multi Agent-Neural Network Intrusion Detection with Mobile Visualization

Álvaro Herrero[1], Emilio Corchado[1], María A. Pellicer[1], and Ajith Abraham[2]

[1] Department of Civil Engineering, University of Burgos
C/ Francisco de Vitoria s/n, 09006 Burgos, Spain
{ahcosio, escorchado}@ubu.es
[2] Norwegian University of Science and Technology, Norway
ajith.abraham@ieee.org

Abstract. A multiagent system that incorporates an Artificial Neural Networks based Intrusion Detection System (IDS) has been defined to guaranty an efficient computer network security architecture. The proposed system facilitates the intrusion detection in dynamic networks. This paper presents the structure of the Mobile Visualization Connectionist Agent-Based IDS, more flexible and adaptable. The proposed improvement of the system in this paper includes deliberative agents that use the artificial neural network to identify intrusions in computer networks. The agent based system has been probed through anomalous situations related to the Simple Network Management Protocol.

Keywords: Multiagent Systems, Artificial Neural Networks, Unsupervised Learning, Projection Methods, Computer Network Security, Intrusion Detection.

1 Introduction and Previous Work

As it is known, the rapid growing of computer networks and the interconnection among them has entailed some security problems. New security failures are discovered everyday and there are a growing number of bad-intentioned people trying to take advantage of such failures. It is unquestionable that organizations need to protect their systems from these intruders and consequently, new network security tools are being developed. The most widely used tool of this kind is firewalls but Intrusion Detection Systems (IDSs) are becoming more and more popular. They monitor the activity of the network with the purpose of identifying intrusive events and can take actions to abort these risky events.

A wide range of techniques have been used to build IDSs. On the one hand, there have been some previous attempts to take advantage of agents and Multiagent Systems (MAS) in the field of Intrusion Detection (ID), as for example [1], [2], [3]. It is worth mentioning the mobile-agents approach [4], [5]. On the other hand, some different machine learning models – including Data Mining techniques and Artificial Neural Networks (ANN) – have been successfully applied for ID, as for example [6], [7], [8], [9], [10].

Additionally, some other AI techniques have been combined (such as genetic algorithms and fuzzy logic [11], genetic algorithms and K-Nearest Neighbor (K-NN)

[12] or K-NN and ANN [13] among others) in order to face ID from a hybrid point of view. In some cases they provide intelligence to MAS. This paper proposes the use of a dynamic multiagent architecture employing deliberative agents capable of learning and evolving with the environment. Some of the agents contained in this architecture are known as CBR-BDI agents [14] because they integrate the BDI (Believes, Desires and Intentions) model and the Case-Based Reasoning (CBR) paradigm. These agents may incorporate different identification or projection algorithms depending on their goals. In this case, an ANN will be embedded in such agents to perform ID in computer networks.

The use of embedded ANN in the deliberative agents of a dynamic MAS let us take advantage of some of the properties of ANN (such as generalization) and agents (reactivity, proactivity and sociability) making the ID task possible. The overall architecture of the system as well as its different components are described as follows: Section 2 outlines the concept of CBR-BDI agent, section 3 describes the overall architecture of the system, and section 4 presents some conclusions and future work.

2 CBR-BDI Architecture

The Mobile Visualization Connectionist Agent-Based IDS (MOVICAB-IDS) [10], [15], [16] has been designed to detect anomalous situations taking place in a computer network. In this paper, some of the agents of the MOVICAB-IDS MAS have been implemented as CBR-BDI agents including an ANN for ID. The multiagent architecture of MOVICAB-IDS is depicted in Fig 1. CBR-BDI agents [17], [18] use CBR systems [19] as a reasoning mechanism, which allows them to learn from initial knowledge, to interact autonomously with the environment, users and other agents within the system, and to have a large capacity for adaptation to the needs of its surroundings.

MOVICAB-IDS includes as a novelty, in this paper, deliberative agents using a CBR architecture, that allows them to respond to events, to take the initiative according to their goals, to communicate with other agents, to interact with users, and to make use of past experiences to find the best information to achieve goals. The proposed CBR-BDI agents work at a high level with the concepts of Believes, Desires and Intentions (BDI) [20]. CBR-BDI agents have learning and adaptation capabilities, what facilitates their work in dynamic environments. Different models can be embedded in the four steps of the CBR reasoning process. In this work, these agents use an ANN to identify intrusions in computer networks. The following section outlines the architecture and reasoning process of the CBR-BDI agents used for ID.

3 MOVICAB-IDS Architecture

MOVICAB-IDS has been constructed as an IDS for distributed computer networks. The proposed MAS incorporates different types of agents. Some of them are reactive agents while others are CBR-BDI agents. The extended version of the Gaia methodology [21] has been applied, and some roles and protocols where identified after the Architectural Design Stage. Examples of protocols are the NegotiateAnalysis protocol (when new data is ready for analysis, an analyzer is chosen) and the ChangeAnalysisConfig protocol (when the configuration of the analysis has be changed, the new configuration is sent to the CONFIGURATIONMANAGER). The Detailed Design Stage concluded that there is a one-to-one correspondence between roles and agent classes in this system, so the agent classes finally identified are: Sniffer, Preprocessor, Analyzer, ConfigurationManager, Coordinator and Visualizer. The outcomes of Gaia methodology [21], [22] are modelled by AUML [23]. Six agents have been developed in this study to improve previous versions of MOVICAB-IDS. They all are listed, and special attention will be placed to the ANALYZER CBR-BDI agent.

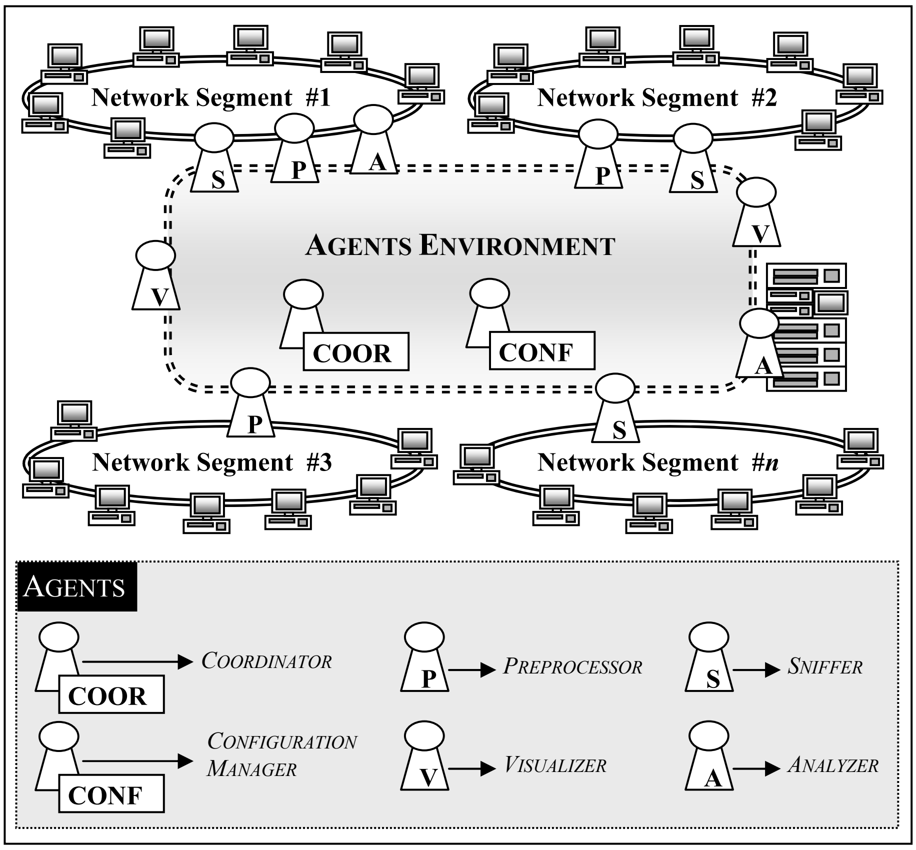

Fig. 1. MOVICAB-IDS Structure

Sniffer

This reactive agent is in charge of capturing traffic data. The continuous traffic flow is captured and split into segments in order to send it through the network for further process. Finally, the readiness of the data is communicated. One agent of this class is located in each of the network segments that the IDS has to cover (from 1 to *n*).

Additionally, there are cloned agents (one per network segment) ready to substitute the active ones if they fail because these agents are the most critical ones. Nothing could be done if traffic data is not captured.

Preprocessor
After splitting traffic data, the generated segments must be preprocessed to apply subsequent analysis. For the shake of network traffic, it would be advisable to locate one of these reactive agents in the same host where a SNIFFER is located. By doing so, the high-volume raw data will not travel along the network. Once the data has been preprocessed, an analysis for this new piece of data is requested. The sending of this data will not overload the network as its volume is much smaller than the one of split data.

Analyzer
This is a CBR-BDI agent. It has got embedded a connectionist model within the adaptation stage of its CBR system that helps to analyze preprocessed traffic data. This connectionist model is called Cooperative Maximum Likelihood Hebbian Learning (CMLHL) [24]. It extends the Maximum Likelihood Hebbian Learning (MLHL) model [25] that is a neural implementation of Exploratory Projection Pursuit (EPP). This agent generates a solution (or achieve its goals) by retrieving a case and analyzing the new one using a CMLHL network. Each case incorporates several features, as can be seen in Table 1.

Table 1. Representation of case features. Classes: P (Problem description attribute) and S (Solution description attribute).

Class	Feature	Type	Description
P	Segment length	Integer	Total segment length (in ms).
P	Network segment	Integer	Network segment where the traffic comes from.
P	Date	Date	Date of capturing.
P	#source ports	Integer	Total number of source ports.
P	#destination ports	Integer	Total number of destination ports.
P	#protocols	Integer	Total number of protocols.
P	#packets	Integer	Total number of packets.
P	Protocol/packets	Array	An array (of variable length depending on each dataset) containing information about how many packets of each protocol there are in the dataset.
S	#Iterations	Integer	Number of iterations.
S	Learning rate	Float	Learning rate.
S	p	Float	CMLHL parameter.
S	Lateral strength	Float	CMLHL parameter.

As it is known, the CBR life cycle consists of four steps: retrieval, reuse, revision and retention [19]. The techniques and tools used by the Analyzer agent to implement these steps are described in the following paragraphs.

Retrieval stage: when a new analysis is requested, the ANALYZER agent tries to find the most similar case to the new one. Euclidean distance is used to find the most similar case in the multidimensional space defined by the features characterizing each dataset (see problem description features in Table 1).

Reuse Stage: once the most similar case has been found, its solution is reused. This solution consists of the values of the parameters used to train a connectionist model (see solution description features in Table 1). This model is CMLHL [24]. It extends the MLHL model [25] that is a neural implementation of EPP. The classical statistical method of EPP [26] provides a linear projection of a data set onto a set of basis vectors which best reveal the interesting structure in data. MLHL identifies interestingness by maximising the probability of the residuals under specific probability density functions which are non-Gaussian.

CMLHL extends the MLHL paradigm by adding lateral connections [24], which have been derived from the Rectified Gaussian Distribution [27]. The resultant net can find the independent factors of a data set but does so in a way that captures some type of global ordering in the data set. Considering a *D*-dimensional input vector (**x**), and an *Q*-dimensional output vector (**y**), with W_{ij} being the weight (linking input *j* to output *i*), then CMLHL can be expressed as:

1. Feed-forward step:

$$y_i = \sum_{j=1}^{D} W_{ij} x_j \ ; \ i = 1,\dots,Q \cdot \tag{1}$$

2. Lateral activation passing:

$$y_i(t+1) = \left[y_i(t) + \tau(b - Ay)\right]^{+} ; \ i = 1,\dots,Q \, . \tag{2}$$

3. Feedback step:

$$e_j = x_j - \sum_{i=1}^{Q} W_{ij} y_i \ ; \ j = 1,\dots,D \cdot \tag{3}$$

4. Weight change:

$$\Delta W_{ij} = \eta . y_i . sign(e_j) | e_j |^{p-1} ; \ i = 1,\dots,Q ; \ j = 1,\dots,D \cdot \tag{4}$$

Where: η is the learning rate, τ is the "strength" of the lateral connections, *b* is a bias parameter, *p* a parameter related to the energy function and *A* is a symmetric matrix used to modify the response to the data. The effect of this matrix is based on the relation between the distances separating the output neurons.

A set of trainings (for the same CMLHL model with a combination of parameter values varying in a specified range) is proposed by tacking into account the distance between the new case and the most similar one. That is, if they are quite similar, a reduced set of trainings are going to be performed. On the contrary, if the most similar case is far away from the new one, a great number of trainings are going to be generated.

Revision Stage: the CMLHL model is trained with the new dataset and the combination of parameters values generated in the reuse stage. When the new projections (the outputs of the CMLHL model for each combination) of the dataset are ready, they are shown to the human user (the network administrator typically) through the VISUALIZER agent. The user has to choose one of these projections as the best one; the one that provides the clearest snapshot of the traffic evolution.

Retention Stage: when a projection is chosen by the user, the ANALYZER agent stores a new case containing the dataset-descriptor and the solution (parameter values used to generate this projection) in the case base for future reuse (See Table 1). ANALYZER agents share their case bases.

The ANALYZER is clearly the most resources-consuming class of agents. The amount of computational resources needed to analyze the continuous data coming from different network segments is extremely high. To overcome this demand, ANALYZER agents can be located in high-performance computing cluster or in most common machines (as can be seen in Fig. 1).

ConfigurationManager

It is worth mentioning the importance of the configuration information. The processes of data capture, split, preprocess and analysis depends on the values of several parameters, as for example: packets to capture, segment length, features to extract... All this information is managed by the CONFIGURATIONMANAGER reactive agent, that is in charge of providing this information to the SNIFFER, PREPROCESSOR and ANALYZER agents.

Coordinator

There can be several ANALYZER agents (from 1 to *m*) but only one COORDINATOR. The latter is in charge of sharing the analysis work out among the former. In order to improve the efficiency and perform a real-time processing, the preprocessed data must be dynamically and optimally assigned. This assignment is performed taking into account both the capabilities of the machines where ANALYZER agents are located and the analysis demands (amount and volume of data to be analysed).

Visualizer

This is an interface agent. At the very end of the process, the analyzed data is presented to the network administrator (or the person in charge of the network) by means of a functional and mobile visualization interface. To improve the accessibility of the system, the administrator may visualize the results on a mobile device (as can be seen in Fig. 2), enabling informed decisions to be taken anywhere and at any time. Depending on the platform where the information will be shown, the offered visualization facilities will be different.

4 Results, Conclusions and Future Work

MOVICAB-IDS identifies anomalous situations due to the fact that these situations do not tend to resemble parallel and smooth directions (normal situations) or because their high temporal concentration of packets. It can be seen in Fig. 2, where a port sweep situation has been identified (Group 1) and visualized in a mobile platform. On the other hand, a more advanced visualization is offered in Fig. 3 for a different data set. In this case, it is easy to notice some different directions (Groups A and B) to the normal data ones. Also, the density of the packets is higher for these anomalous groups.

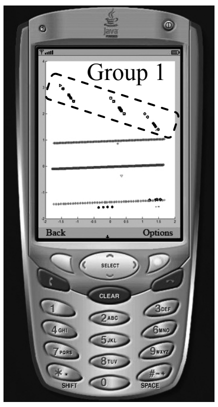

Fig. 2. Mobile Visualization

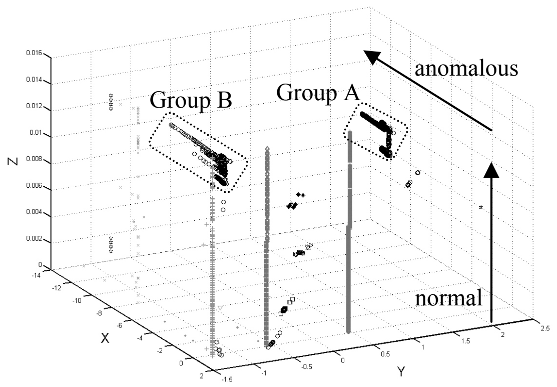

Fig. 3. Advanced Visualization

The effectiveness of MOVICAB-IDS in facing some anomalous situations has been widely demonstrated in previous works [10], [15], [16]. The projections obtained [by CMLHL overcome that obtained by Principal Component Analysis (PCA) [10], Maximum Likelihood Hebbian Learning [28] and Auto Associative Back Propagation Networks [28]. Figures 4 and 5 show a comparison of projections obtained by CMLHL (Fig. 4) and PCA (Fig. 5) for the same data. As can be easily seen, CMLHL is able to identify the port sweeps (Groups 1 and 2) contained in the dataset while PCA is not.

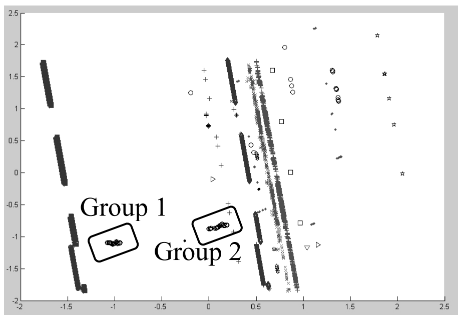

Fig. 4. CMLHL Projection

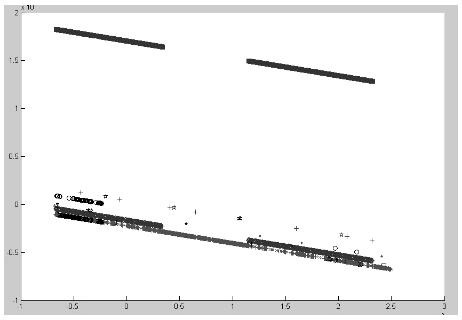

Fig. 5. PCA Projection

As a conclusion, we could say that this paper presents an improved MAS that incorporates CBR-BDI agents with embedded ANN. This improved version of MOVICAB-IDS gives the following advantages:

– Scalability: new agents (both SNIFFER and ANALYZER) can be dynamically added at any time.
– Failure tolerance: backup instances of some agents can be ready to run as soon as the working instances fail, showing a proactive behaviour.
– Real-time processing: by splitting the data and allowing the system to process it in different processing units (agents located in different machines).
– Mobile visualization: the visualization task can be performed in a wide variety of devices (as it is shown in Fig. 2).

Future work will focus on improving the dynamical assignment of analysis in order to take advantage of the computational resources in a more efficient way and on studying different distributions and learning rules for the data analysis. New neural models or adaptations of the existent ones would be tried.

Acknowledgments. This research has been partially supported by the MCyT project TIN2004-07033 and the project BU008B05 of the JCyL.

References

1. Spafford, E.H., Zamboni, D.: Intrusion Detection Using Autonomous Agents. Computer Networks: The Int. Journal of Computer and Telecommunications Networking 34(4), 547-570 (2000)
2. Hegazy, I.M., Al-Arif, T., Fayed, Z.T., Faheem, H.M.: A Multi-agent Based System for Intrusion Detection. IEEE Potentials 22(4), 28-31 (2003)
3. Dasgupta, D., Gonzalez, F., Yallapu, K., Gomez, J., Yarramsettii, R.: CIDS: An agent-based intrusion detection system. Computers & Security 24(5), 387-398 (2005)
4. Wang, H.Q., Wang, Z.Q., Zhao, Q., Wang, G.F., Zheng, R.J., Liu, D.X.: Mobile Agents for Network Intrusion Resistance. In: APWeb 2006. LNCS, vol. 3842, pp. 965-970. Springer, Heidelberg (2006)
5. Deeter, K., Singh, K., Wilson, S., Filipozzi, L., Vuong, S.: APHIDS: A Mobile Agent-Based Programmable Hybrid Intrusion Detection System. In: Mobility Aware Technologies and Applications. LNCS, vol. 3284, pp. 244-253. Springer, Heidelberg (2004)
6. Laskov, P., Dussel, P., Schafer, C., Rieck, K.: Learning Intrusion Detection: Supervised or Unsupervised? In: Roli, F., Vitulano, S. (eds.) ICIAP 2005. LNCS, vol. 3617, pp. 50-57. Springer, Heidelberg (2005)
7. Liao, Y.H., Vemuri, V.R.: Use of K-Nearest Neighbor Classifier for Intrusion Detection. Computers & Security 21(5), 439-448 (2002)
8. Sarasamma, S.T., Zhu, Q.M.A., Huff, J.: Hierarchical Kohonenen Net for Anomaly Detection in Network Security. IEEE Transactions on Systems Man and Cybernetics 35(2), 302-312 (2005)
9. Zanero, S., Savaresi, S.: Unsupervised Learning Techniques for an Intrusion Detection System. In: Proc. of the ACM Symposium on Applied Computing. pp. 412-419 (2004)
10. Corchado, E., Herrero, A., Sáiz, J.M.: Detecting Compounded Anomalous SNMP Situations Using Cooperative Unsupervised Pattern Recognition. In: Duch, W., Kacprzyk, J., Oja, E., Zadrozny, S. (eds.) ICANN 2005. LNCS, vol. 3697, pp. 905-910. Springer, Heidelberg (2005)

11. Sindhu, S.S.S., Ramasubramanian, P., Kannan, A.: Intelligent Multi-agent Based Genetic Fuzzy Ensemble Network Intrusion Detection. In: Neural Information Processing. LNCS, pp. 983-988. Springer, Heidelberg (2004)
12. Middlemiss, M., Dick, G.: Feature Selection of Intrusion Detection Data Using a Hybrid Genetic Algorithm/KNN Approach. In: Design and application of hybrid intelligent systems. IOS Press. 519-527 (2003)
13. Kholfi, S., Habib, M., Aljahdali, S.: Best Hybrid Classifiers for Intrusion Detection. Journal of Computational Methods in Science and Engineering 6(2), 299 - 307 (2006)
14. Carrascosa, C., Bajo, J., Julián, V., Corchado, J.M., Botti, V.: Hybrid Multi-agent Architecture as a Real-Time Problem-Solving Model. Expert Systems with Applications: An International Journal 34(1), 2-17 (2008)
15. Corchado, E., Herrero, A., Saiz, J.M.: Testing CAB-IDS through Mutations: on the Identification of Network Scans. In: Proc. of the Int. Conf. on Knowledge-Based and Intelligent Information & Engineering Systems. LNAI, vol. 4252, pp. 433-441 (2006)
16. Herrero, A., Corchado, E., Sáiz, J.M.: MOVICAB-IDS: Visual Analysis of Network Traffic Data Streams for Intrusion Detection. In: Corchado, E., Yin, H., Botti, V., Fyfe, C. (eds.) IDEAL 2006. LNCS, vol. 4224, pp. 1424-1433. Springer, Heidelberg (2006)
17. Corchado, J.M., Laza, R.: Constructing Deliberative Agents with Case-Based Reasoning Technology. International Journal of Intelligent Systems 18(12), 1227-1241 (2003)
18. Pellicer, M.A., Corchado, J.M.: Development of CBR-BDI Agents. International Journal of Computer Science and Applications 2(1), 25 - 32 (2005)
19. Aamodt, A., Plaza, E.: Case-Based Reasoning - Foundational Issues, Methodological Variations, and System Approaches. AI Communications 7(1), 39-59 (1994)
20. Bratman, M.E.: Intentions, Plans and Practical Reason. Harvard University Press, Cambridge, M.A. (1987)
21. Zambonelli, F., Jennings, N.R., Wooldridge, M.: Developing Multiagent Systems: the Gaia Methodology. ACM Transactions on Software Engineering and Methodology 12(3), 317-370 (2003)
22. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. Autonomous Agents and Multi-Agent Systems 3(3), 285-312 (2000)
23. Bauer, B., Müller, J.P., Odell, J.: Agent UML: A Formalism for Specifying Multiagent Software Systems. International Journal of Software Engineering and Knowledge Engineering 11(3), 1-24 (2001)
24. Corchado, E., Fyfe, C.: Connectionist Techniques for the Identification and Suppression of Interfering Underlying Factors. Int. Journal of Pattern Recognition and Artificial Intelligence 17(8), 1447-1466 (2003)
25. Corchado, E., MacDonald, D., Fyfe, C.: Maximum and Minimum Likelihood Hebbian Learning for Exploratory Projection Pursuit. Data Mining and Knowledge Discovery 8(3), 203-225 (2004)
26. Friedman, J.H., Tukey, J.W.: A Projection Pursuit Algorithm for Exploratory Data-Analysis. IEEE Transactions on Computers 23(9), 881-890 (1974)
27. Seung, H.S., Socci, N.D., Lee, D.: The Rectified Gaussian Distribution. Advances in Neural Information Processing Systems 10, 350-356 (1998)
28. Herrero, A., Corchado, E., Gastaldo, P., Zunino, R.: A Comparison of Neural Projection Techniques Applied to Intrusion Detection Systems. In: Sandoval, F., Prieto, A., Cabestany, J., Graña, M. (eds.) IWANN'2007. LNCS, vol. 4507, pp. 1138-1146. Springer, Heidelberg (2007)