RT-MOVICAB-IDS:
Addressing Real-Time Intrusion Detection

Álvaro Herrero[A], Martí Navarro[B], Emilio Corchado[C], and Vicente Julián[B]

[A]Department of Civil Engineering, University of Burgos, Spain
C/ Francisco de Vitoria s/n, 09006 Burgos, Spain
ahcosio@ubu.es
[B]Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia,
Camino de Vera s/n, 46022, Valencia, Spain
mnavarro@dsic.upv.es, vinglada@dsic.upv.es
[C]Departamento de Informática y Automática, University of Salamanca,
Plaza de la Merced s/n, 37008 Salamanca, Spain
escorchado@usal.es

Corresponding author: Prof. Emilio Corchado.
Address: Departamento de Informática y Automática, University of Salamanca, Plaza de la Merced s/n, 37008 Salamanca, Spain.
Tel: (+34) 616 44 9888. Fax: (+34) 923294514.

Abstract. This study presents a novel Hybrid Intelligent Intrusion Detection System (IDS) known as RT-MOVICAB-IDS that incorporates temporal control. One of its main goals is to facilitate real-time Intrusion Detection, as accurate and swift responses are crucial in this field, especially if automatic abortion mechanisms are running. The formulation of this hybrid IDS combines Artificial Neural Networks (ANN) and Case-Based Reasoning (CBR) within a Multi-Agent System (MAS) to detect intrusions in dynamic computer networks. Temporal restrictions are imposed on this IDS, in order to perform real/execution time processing and assure system response predictability. Therefore, a dynamic real-time multi-agent architecture for IDS is proposed in this study, allowing the addition of predictable agents (both reactive and deliberative). In particular, two of the deliberative agents deployed in this system incorporate temporal- bounded CBR. This upgraded CBR is based on an *anytime* approximation, which allows the adaptation of this Artificial Intelligence paradigm to real-time requirements. Experimental results using real data sets are presented which validate the performance of this novel hybrid IDS.

Research highlights: RT-MOVICAB-IDS, a novel hybrid IDS incorporating temporal control, is presented.> Temporal restrictions allow real time processing and system response predictability.> Experimental results using real data sets are presented which validate the performance.

Keywords: Hybrid Artificial Intelligent Systems, Unsupervised Learning, Artificial Neural Networks, Multi-agent Systems, Case-Based Reasoning, Computer Network Security, Intrusion Detection, Time Bounded Deliberative Process.

1 Introduction

A network attack or intrusion is an action that threatens to affect any of the three computer security principles: availability, integrity and confidentiality by exploiting, for example, Denial of Service, Modification, or Destruction vulnerabilities [1]. One of the most harmful points of attacks and intrusions, increasing the difficulty of protecting computer systems, is the ever-changing nature of attack technologies and strategies. For this reason among others, Intrusion Detection Systems (IDSs) have become a required asset in addition to the computer security infrastructure of most organizations. In the context of computer networks, an IDS can, in broad terms, be defined as a tool that is designed to detect suspicious patterns that may be related to a network or system attack. Intrusion Detection (ID) is therefore a field that focuses on the identification of attempted or ongoing attacks on a computer system or network.

A Real-Time (RT) response in this field (computer security) is very important as less than fifteen minutes are required by some distributed and coordinated attacks to stop a large area of the Internet from normal functioning [2]. As a consequence, response time [3] is a critical issue for most of the security infrastructure components of an organization. The importance of a fast, predictable and smart response increases in the case of IDSs as spending too much time on training is clearly inadequate for RT self-adaptive ID [4]. Furthermore, an automated response will be almost useless if triggered after a host is infected [5].

Systems that require a response before a specific deadline, as determined by their requirements, make it essential to monitor execution times. Each task must be performed by the system within a predictable timeframe, within which accurate execution of the given response must be guaranteed. This is the main reason for time-bounding the analytical tasks of IDSs. This temporal issue has been also addressed by the Artificial Intelligence (AI) community. Over recent years, AI techniques have been applied in RT environments to provide RT systems with intelligent methods to solve complex problems. More precisely, there are various proposals to adapt AI techniques to RT requirements; the most promising algorithms within this field are Anytime algorithms [6] and approximate processing [7]. One line of research in RT Artificial Intelligence (RTAI) is related to large applications or hybrid system architectures that embody RT concerns in many components [7], such as Guardian [8], Phoenix [9], or SA-CIRCA [10]. In the context of the aforementioned research in the area of RTAI and the well-known area of Multi-Agent Systems (MAS), a Real-Time Agent (RTA) can be defined as an agent with temporal constraints in at least one of its responsibilities [11]. So, an agent assigned to RT environments must accomplish its goals, responsibilities and tasks with the additional difficulty of temporal constraints. Such agents may have temporal bounded interactions, a modification that will affect all communication processes in the MAS where the RTA is located.

Accordingly, the paper presents RT-MOVICAB-IDS (MObile VIsualisation Connectionist Agent-Based IDS) [12-14]: a novel Hybrid Artificial Intelligent System (HAIS) IDS with a temporal-bounded intrusion detection mechanism. This system monitors network activity to identify intrusive events by combining different AI paradigms to visualise network traffic for ID at packet level [15]. It is based on a dynamic MAS, which integrates an unsupervised neural projection model and the Case-Based Reasoning (CBR) paradigm [16], through the use of deliberative agents that are capable of learning and evolving with the environment. It is worth highlighting that temporal restrictions are imposed on this IDS in order to perform real/execution time processing and assure system response predictability. To do so, some of the deliberative agents within RT-MOVICAB-IDS have been designed according to these temporal restrictions.

The rest of the paper is organized as follows. Section 2 contains a brief review of the state of the art of IDSs, Real Time and AI areas and methodologies applied in this interdisciplinary study. Section 3 provides an overview of the novel proposed Hybrid-IDS, in which each step forming this system is described in detail. Additionally, sample visualizations of real-traffic data are shown in Section 4 to illustrate the output of the system. Some experimental results on CPU utilization and Average Execution Time are also presented in this section to show the main outcomes of the proposed system. Finally, the conclusions and future work are discussed in Section 5.

2 Previous Work

This section presents the state of the art of several AI areas and methodologies applied to the novel Hybrid IDS presented in this research. There are three main research areas related to this interdisciplinary study: visualization tools based on unsupervised learning, intrusion detection systems and RT agents. These areas are explained and reviewed in the following subsections.

2.1 Visualization Tools Based on Unsupervised Learning

Projection methods project high-dimensional data points onto lower dimensions in order to identify "interesting" directions in terms of any specific index or projection. Such indices or projections are, for example, based on the identification of directions that account for the largest variance of a dataset, such as Principal Component Analysis (PCA) [17-19], or the identification of higher-order statistics such as the skew or kurtosis index, which is the case of Exploratory Projection Pursuit (EPP) [20]. Having identified the interesting projections, the data is then projected onto a lower dimensional subspace plotted in two or three dimensions, which makes it possible to examine its structure with the naked eye. The remaining dimensions are discarded as they mainly relate to a very small percentage of the information or the dataset structure. In that way, the structure identified through a multivariable dataset may be visually analysed with greater ease.

The combination of this type of technique together with the use of scatter plot matrixes constitutes a very useful visualization tool to investigate the intrinsic structure of multidimensional datasets, allowing experts to study the relations between different components, factors or projections, depending on the technique that is used.

From a purely "projection-of-packets" standpoint, some dimensionality reduction techniques - e.g. PCA- have previously been proposed for visualising network data through scatter plots [21-25].

2.2 Intrusion Detection Systems

Intrusion Detection (ID) has been approached from several different points of view up to now; many different intelligent and Soft Computing techniques (such as Genetic Programming [21], Data Mining [22, 23], Fuzzy Logic [24, 25], or Neural Networks [26-28] among others) together with statistical [29] and signature verification [30] techniques have mainly been applied to perform a 2-class classification (normal/anomalous or intrusive/non-intrusive). Most of these systems can generate different alarms when an anomalous situation is detected, but they cannot provide a general overview of what is happening inside a computer network.

In contrast, a great variety of visualization-based approaches to Intrusion Detection have also been proposed. In this case, the ID task is enabled by providing a visual depiction of the network or the traffic. Thus, the identification of attacks must be performed through visual features because no alarms are triggered. Visualization tools rely on the human ability to recognize different features and detect anomalies through graphical devices [31]. Apart from enabling the detection of anomalies, one of the main advantages of this approach is that it can provide a general snapshot of network traffic. As this study focuses on the visualization of network traffic data rather than network structure or topology, only previous work on network data visualization is considered in this section.

Network data are summarized in previous work by:

- **IP addresses**: that is the case of the Galaxy View of NVisionIP [32]. In [33], Border Gateway Protocol data are visualized by a diagram based on IP addresses. IP segments are used in NIVA [34] to locate and colour the data. The Time-based Network Traffic Visualizer [35] combines a matrix display of host IP address and packet timestamps. An IP address-based matrix is also proposed in [36] to detect the propagation of the Welchia and Sasser. D worms.

- **Port numbers**: the main visualization proposed in [37] is based on port and time information. Stacked histograms of aggregate port activity are proposed in [38]. By using port numbers and IP addresses, the system proposed in [38] is able to see the penetration and subsequent activity of the Sasser worm.
- **Different measurements of network traffic**: the Multi Router Traffic Grapher [39] shows the incoming/outgoing traffic in Bits per Second while IDGraphs [40] uses the number of unsuccessful connections.
- **Alarm data**: generated by different IDSs, such as Snort [41] or StealthWatch IDS [42].
- **Others**: additional kinds of data can be also processed by different visualization tools, such as VIAssist [43] or IDtk [44] that are applied to raw TCP packet data or alerts generated by IDS tools.

In contrast to other security tools, IDSs need to be monitored [45]. So, an IDS can be useless if nobody is looking at its outputs. In keeping with this idea, the proposed IDS combines several features extracted from packet headers to depict each simple packet by using neural unsupervised methods based on Exploratory Projection Pursuit (EPP) [20, 46]. It provides the network administrator with a snapshot of network traffic, protocol interactions, and traffic volume generally in order to identify anomalous situations. To do so, an unsupervised neural model (see section 3.4) is applied.

Most of the solutions described in this section use a glyph metaphor [34, 44, 47] to encode information by changing different features (colour, size, opacity, etc.) in addition to the spatial coordinates, while others use traditional representation techniques such as histograms [38, 48], histographs [49] or other graphs [50, 51]. The novel IDS proposed in this research employs the glyph metaphor as well, using different colours and shapes in addition to the spatial coordinates to provide information on the protocol of each packet.

Multi-Agent Systems have been previously applied to the ID problem [52-54]. CIDS (Cougaar-based IDS) [52] provides a hierarchical security agent framework, where a security node consists of four different agents (manager agent, monitor agent, decision agent and action agent) developed over the Cougaar framework [55]. Some works [56, 57] have been carried out using the mobile-agent approach. APHIDS [56] implements the distributed search and analysis tasks with mobile agents equipped with scripting capability to automate evidence gathering.

Considering all this previous work on agent-based ID, the main novelty of RT-MOVICAB-IDS is the inclusion of deliberative (CBR-BDI) agents in a specific IDS for packet ID through visualization based on neural models. Additionally, the IDS proposed in this study incorporates temporal restrictions to state a predictable response time.

2.3 Agents and Real-Time Systems

Real-Time Systems are computer systems in which the correctness of the system behaviour depends not only on the logical results of the computations, but also on the physical instant at which these results are produced. That is the case of security systems where it is important that the detection of the problem will be on time in order to make corrective actions at the right time. The main reason is that less than fifteen minutes are required by some distributed and coordinated attacks to stop a large area of the Internet from functioning [2]. Furthermore, an automated response will be almost useless if triggered after a host is infected [5].

However, classical RT systems are typically rigid and deterministic systems. Classical techniques used in such systems are insufficient if we manage dynamic environments where goals require complex deliberative processes. Conceived to overcome the shortcomings of classic RT systems, RTAI studies how to adapt artificial intelligence techniques to domains where a RT response is required. Anytime algorithms [6] and approximate processing [7] are some examples of the adaptation of AI techniques to RT domains. Another example of AI in the RT domain is proposed by Garvey *et al.* in [58], where a design-to-time scheduling algorithm for incremental decision-making is presented. This algorithm is extended in [59] to develop a more general model that can take any scheduling criteria into account, such as time, cost, and quality and can use uncertainty as part of the decision-making process. An example of the use of the design-to-criteria model is the DECAF architecture [60], which incorporates scheduling algorithms based on this model. One line of research in RTAI has been to build large applications or architectures that embody RT concerns in many components [7], such as Guardian [8], Phoenix [9] and SA-CIRCA (Self-Adaptive Cooperative Intelligent Real-Time Control Architecture), proposed by Musliner *et al.* [10].

In research relating to RTAI Systems, a Real-Time Agent (RTA) can be defined as an agent with temporal restrictions [11]. Temporal correctness has to be taken into account for this kind of agent, which is expressed by means of a set of temporal restrictions that are imposed by the environment, compliance with which must be ensured by the RTA. Over recent years, some examples of the application of agent technology in RT domains have been studied. DiPippo *et al.* [61, 62] presented a Real-Time Multi-Agent system (RT-MAS) based on RT-Corba [63]. The operation of the system is based on CORBA, but here the client and server have RT features. However, this approach has some problems. On one hand, the time needed by one of the offered services, the Scheduling Service, is unknown and, on the other hand, the communication process is temporal unbounded, and therefore, unpredictable.

The *ObjectAgent* Architecture is another example of RTAs. This architecture, developed by Princeton Satellites in 2001 [64], is used to control little mono-functional satellite systems. These satellites work together as a unique satellite with multiple functions. Each mini-satellite is identified by an agent with its temporal restrictions. This architecture supports RT communication, while the net topology is known and predictable. Unfortunately, this assumption is only true for very specific networks (CAN networks, inter-satellite laser links, etc.). Thus, if this platform were extrapolated to common network media (Ethernet, serial, wifi, etc), this feature would be lost.

Another example of RTA is presented by Prouskas and Pitt in [65]. They define time-aware agents as agents capable of operating in two temporal dimensions: agent-agent and human-agent, seamlessly combining the predictability and reliability of small-scale RT exchanges with the fuzzy temporal requirements of large-scale human interactions. Time-aware agent systems deal with an amalgam of hard, soft, human and non-RT interactions, reason the temporal constraints placed on the system by each type of interaction, make transformations between themselves and co-ordinate (schedule) activities seamlessly irrespective of their constituent constraints.

In addition, the ARTIS agent architecture, specially designed to develop RT Systems was also developed [66]. An ARTIS agent is an agent able to operate in distributed RT domains. The ARTIS architecture is an extension of the blackboard model [67], which has been adapted to work in hard RT environments. This architecture includes the use of well-known RTAIS techniques in an approach that is guaranteed to react with the environment in a dynamic and flexible way. The main problems of this proposal are the lack of complex reasoning capabilities in ARTIS agents and the complexity of the design and implementation processes, which makes practical use of this proposal very difficult.

The research projects that are above reviewed show the feasibility of using agent technology within RT domains so as to provide more dynamicity, distribution, flexibility and greater deliberative capabilities than previous approaches. With this in mind, the use of RTA technology appears appropriate for the development of the security systems presented in this paper.

The issue of RT has been considered earlier in the field of ID, mainly because the underlying techniques can reduce processing time so as to enable a faster response time. That is the case of the MAID formulation [68], in which an algorithm for the RT updating of the reference model was designed. Singular Value Decomposition is proposed in [69] as a pre-processing step to reduce the dimensionality of the data for fast ID. RT ID is approached in [70] by mathematically proving that the number of computational steps is reduced by applying Fast Time Delay Neural Networks instead of applying other conventional time delay neural networks. Fast response is also pursued in [71] by speeding up data analysis through Non-negative Matrix Factorization. Apart from the ID engine, [72] proposed a load-balancing device to perform RT ID in high-speed networks. None of these previous studies focus on MAS-based IDSs. As a result, response times of deliberative agents are not taken into account when pursuing a fast and timely IDS responses, which is one of the main novelties, along with predictability, of the present research.

3 RT-MOVICAB-IDS

RT-MOVICAB-IDS (Real-Time MObile VIsualisation Connectionist Agent-Based IDS) focuses on network-based Intrusion Detection (ID) from the visualisation and hybrid AI standpoints. It combines different AI paradigms to visualise network traffic for ID at packet level. As a result of depicting each simple packet and preserving the temporal context, RT-MOVICAB-IDS is able to provide security personnel with a synthetic, intuitive snapshot of network traffic and protocol interactions. This visualisation interface supports the straightforward detection of anomalous situations and their identification (as shown in Section 3.5). Additionally, it can help to ascertain the internal structure and behaviour of the traffic data, thereby improving supervision of network activity.

Different tasks are required to perform traffic monitoring and ID, such as those proposed for traffic management [73], (collecting data, processing collected data, and deploying mechanisms). For the data collecting task, a 4-stage framework [74] is adapted to RT-MOVICAB-IDS in the following way:

1. **Data capture:** as network-based ID is pursued, the continual data flow of network traffic must be managed. This data flow contains information on all the packets travelling along the network to be monitored. Only a reduced portion of this data is captured at this time for further process. Accordingly, few fields from the packet headers (timestamp, source and destination ports, size and protocol) are selected to generate the datasets.
2. **Data selection**: Network IDSs have to deal with the practical problem of high volumes of quite diverse data [75]. To manage high diversity of data, RT-MOVICAB-IDS splits the traffic into different groups, taking into account the protocol (UDP, TCP, ICMP, and so on) over IP, as there are differences between the headers of these protocols. Once the captured data is classified by the protocol, it can be processed in different ways.
3. **Segmentation**: The two first stages do not deal with the problem of continuity in network traffic data. The neural model to be later applied cannot process data "on the fly". To overcome this shortcoming, the segmentation task is in charge of creating temporarily limited datasets from the continuous network data flow. To do so, RT-MOVICAB-IDS splits the pre-processed data stream into simple (containing all the packets whose timestamp is between the segment initial and final time limit) and accumulated segments (consisting of the addition of several consecutive simple segments).
4. **Data pre-processing**: Finally, the different datasets must be pre-processed before presenting them to the neural model in subsequent stages. At this stage, categorical features are converted into numerical ones. This happens with the protocol information; each packet is assigned a previously defined value according to the protocol to which it belongs.

Once the data-collecting task is performed and the data is ready, the ID process of RT-MOVICAB-IDS performs two further tasks:

- **Data analysis**: a neural model called Cooperative Maximum Likelihood Hebbian Learning (CMLHL) [76] is applied to analyse the data. Some other unsupervised models have also been applied to perform this task for comparison purposes.
- **Visualisation**: the projections of simple and accumulated segments are presented to the network administrator for scrutiny and monitoring. One interesting feature of the proposed IDS is its mobility; this visualisation task may be performed on a different device other than the one used for the previous tasks. To improve the accessibility of the system, results may be visualised on a mobile device (such as phones or blackberries), enabling informed decisions to be taken anywhere and at any time.

In summary, the RT-MOVICAB-IDS task organisation comprises the six tasks described above. The following AI paradigms are combined within RT-MOVICAB-IDS to perform these tasks:

- **Multi-agent system**: some of the components are wrapped as deliberative agents, capable of learning and evolving with the environment [77]. As described below, they integrate AI techniques and models, becoming intelligent agents [78].

- **Case-based reasoning**: some of the agents contained in the MAS are known as CBR-BDI agents [79] because they integrate the BDI (Beliefs, Desires and Intentions) [80] model and the CBR (Case-Based Reasoning) [16] paradigm.
- **Artificial neural networks**: the connectionist approach fits the ID challenge mainly because it allows a system to learn, in an empirical way, the input-output relationship between traffic data and its subsequent interpretation [28]. Some of the previously described CBR-BDI agents incorporate an unsupervised neural model to generate projections of network traffic.

To assess RT-MOVICAB-IDS in the fulfilment of the ID tasks, a novel testing method based on mutations was developed [14]. The main idea behind this technique is to confront RT-MOVICAB-IDS (and some other visualization-based IDSs) with previously unseen attacks. These novel situations simulate the new attacks (known as "0-day" attacks) that a computer system may face for the first time. Additionally, the neural model supporting the visualization capabilities of RT-MOVICAB-IDS is tested by comparing its projections to those generated by some other neural projection models.

3.1 Multiagent System

An extended version of the Gaia methodology [81, 82] was applied to design the RT-MOVICAB-IDS MAS according to the previously introduced tasks. The following roles were identified after the Architectural Design Stage of the methodology:

- **SNIFFER**: this role involves continuously capturing the traffic data flowing across a network segment. At the same time, when there is enough captured data, this data is split and its readiness is communicated to other roles.
- **PREPROCESSOR**: this role preprocesses the captured data. After that, an analysis for this new piece of data is requested.
- **ANALYZER**: this role negotiates for data analysis. Once an analysis is assigned to this role, it analyzes the new preprocessed data.
- **CONFIGURATIONMANAGER**: this role involves managing the configuration of several parameters (related to the splitting, pre-processing and the analysis of traffic data) and making such information available to some other roles.
- **COORDINATOR**: this organizational role involves coordinating some of the other roles and balancing the workload among them.
- **VISUALIZER**: this role is responsible for updating the visualization when new information (analyzed data or system information) is generated.

The following protocols were also defined after this stage: *AnalysisAborted, AnalysisCompleted, ChangeSplitConfig, ChangePreprocessConfig, ChangeAnalysisConfig, ManageSplitError, NegotiateAnalysis, PreprocessAborted, PreprocessedDataReady, RequestAnalysisConfig, RequestAnalyzedData, RequestPreprocessConfig, RequestPreprocessedData, RequestSplitConfig, RequestSplitData, RequestVisualization, SplitAborted, SplitDataReady, UpdateAnalysisConfig, UpdatePreprocessConfig, UpdateSplitConfig, UpdateSystemInfo.*

It may be concluded from the Detailed Design Stage that there is a one-to-one correspondence between roles and agent classes in the system. As a result, RT-MOVICAB-IDS incorporates six agents, as shown in Fig. 1.

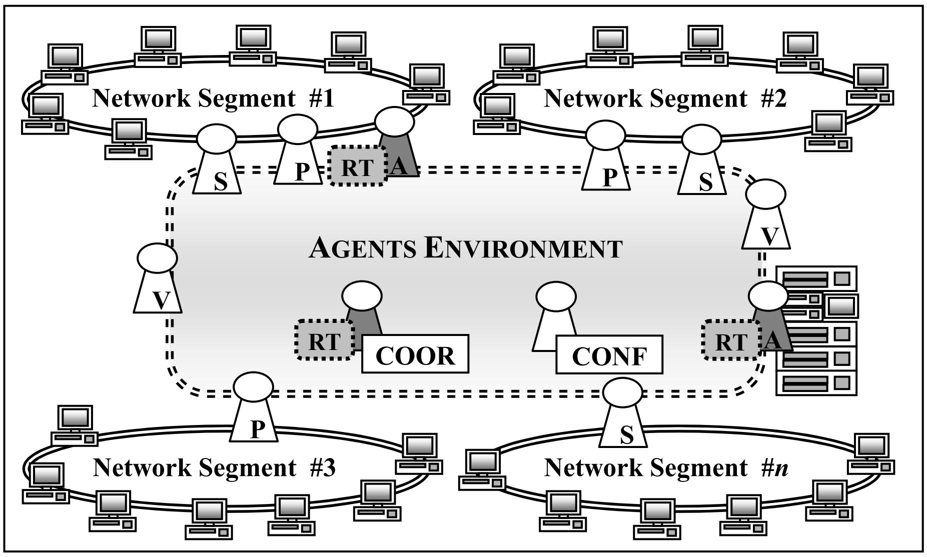


Fig. 1. RT-MOVICAB-IDS architecture.

The agents included in RT-MOVICAB-IDS can therefore be defined as:

- **Sniffer** (S in Fig. 1): this reactive agent is in charge of capturing traffic data. The continuous traffic flow is captured and split into segments in order to send them through the network for further processing. Then, the readiness of the data for pre-processing 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*).
- **Preprocessor** (P in Fig. 1): after splitting traffic data, the generated segments are pre-processed prior to their analysis. Once the data has been pre-processed, an analysis for this new piece of data is requested.
- **Analyzer** (A in Fig. 1): this is a CBR-BDI agent. It has a connectionist model (CMLHL [76]) embedded in the adaptation stage of its CBR system that helps to analyze the pre-processed traffic data. This agent generates a solution (or achieves its goals) by retrieving a case and analyzing the new one using a CMLHL network. This RT agent is comprehensively described in section 3.4.
- **ConfigurationManager** (CONF in Fig. 1): the configuration information (such as packets to capture, segment length, features to extract,...) is managed by this agent, which is in charge of providing this information to the Sniffer, Pre-processor, and Analyzer agents. A reactive architecture was chosen as this agent requires no training and all of its decisions are based on local information.
- **Coordinator** (COOR in Fig. 1): there can be several Analyzer agents (from 1 to *m*) but only one Coordinator. The latter is in charge of distributing the analyses among the former. In order to improve the efficiency and perform RT processing, the pre-processed data must be dynamically and optimally assigned. This assignment is performed taking into account both the

capabilities of the machines where the Analyzer agents are located and the analytical demands (amount and volume of data to be analysed).

- **Visualizer** (V in Fig. 1): 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, mobile visualization interface. To improve the accessibility of the system, the administrator may visualize the results on a mobile device, enabling informed decisions to be taken anywhere and at any time.

3.2 Addressing Real-Time

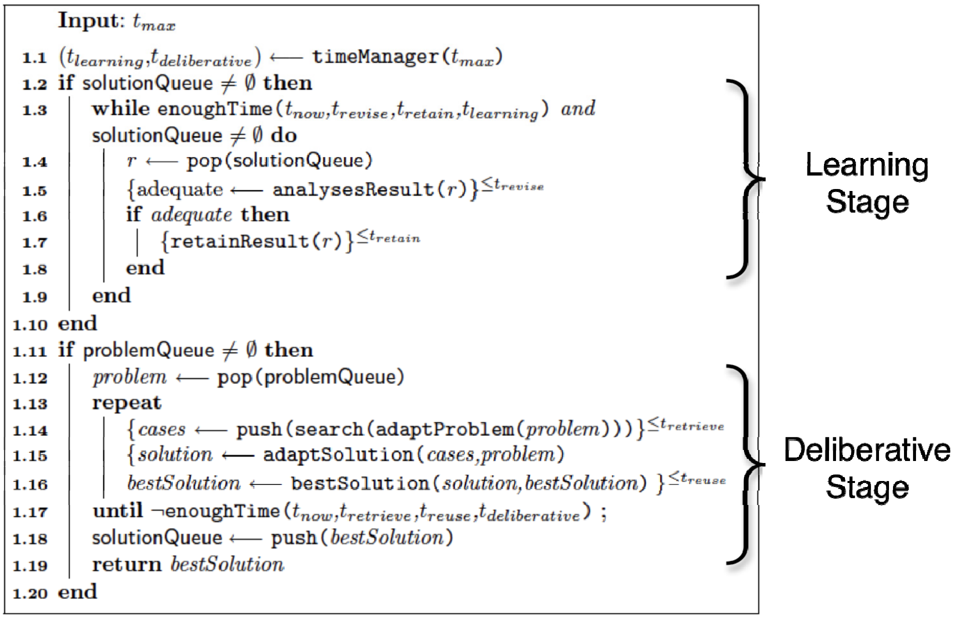
The agents in charge of the analysis and the coordination of the whole process must be temporally bounded, in order for RT-MOVICAB-IDS to complete its analysis within a maximum time. It is therefore necessary to adapt the AI techniques they employ in order to guarantee RT constraints. For this reason, RTAs, which provide the necessary control mechanisms to carry out this task, are used to complete the analysis and coordination on time. When a new segment is ready for analysis, the Coordinator agent has a limited amount of time to assign the pending analysis to the available Analyzer agents, which in turn, must provide an answer as soon as possible before a given deadline.

To apply the CBR paradigm [16] as a reasoning mechanism in RTAs, it is necessary to adapt the techniques to be executed so that they satisfy RT requirements. In RT environments, the CBR stages (Retrieve, Revise, Reuse and Retain) must be temporally bounded to ensure that the solutions are produced on time; giving the system a temporal bounded deliberative case-based behaviour. Thus, a Temporal Bounded CBR (TB-CBR) mechanism [83] is suitable as the basis of the deliberative reasoning of RTAs.

The proposed TB-CBR algorithm is a modification of the classical CBR cycle to be applied in domains with RT constraints. Algorithm 1 shows a pseudo-code of this approach. First, the four phases of the CBR cycle are grouped in two stages defined as:

- **Learning stage**, which consists of the revise and retain phases.
- **Deliberative stage**, which includes the retrieve and reuse phases.

Each phase will schedule its execution time. Therefore, the designer can choose to either assign more time to the deliberative stage, or keep some time for the learning stage (and thus the designed agents will be more sensitive to updates). These CBR stages must be designed as an anytime algorithm, where the process is iterative and each iteration is time-bounded and may improve the final response.



```
Input: $t_{max}$
$(t_{learning}, t_{deliberative}) \longleftarrow$ timeManager($t_{max}$)
if solutionQueue $\neq \emptyset$ then
    while enoughTime($t_{now}$,$t_{revise}$,$t_{retain}$,$t_{learning}$) and
    solutionQueue $\neq \emptyset$ do
        $r \longleftarrow$ pop(solutionQueue)
        {adequate $\longleftarrow$ analysesResult($r$)}$^{\leq t_{revise}}$
        if adequate then
            {retainResult($r$)}$^{\leq t_{retain}}$
        end
    end
end
if problemQueue $\neq \emptyset$ then
    problem $\longleftarrow$ pop(problemQueue)
    repeat
        {cases $\longleftarrow$ push(search(adaptProblem(problem)))}$^{\leq t_{retrieve}}$
        {solution $\longleftarrow$ adaptSolution(cases,problem)
        bestSolution $\longleftarrow$ bestSolution(solution,bestSolution) }$^{\leq t_{reuse}}$
    until $\neg$enoughTime($t_{now}$,$t_{retrieve}$,$t_{reuse}$,$t_{deliberative}$) ;
    solutionQueue $\longleftarrow$ push(bestSolution)
    return bestSolution
end
```

Algorithm 1. Time-Bounded CBR algorithm.

The TB-CBR cycle starts at the learning stage, which entails checking whether previous cases are awaiting revision and could be stored in the case-base. The plans provided at the end of the deliberative stage are stored in a solution list while feedback on their utility is received. This list is accessed when each new TB-CBR cycle begins. If there is sufficient time, the learning stage is implemented for cases where solution feedback has recently been received. If the list is empty, this process is omitted.

The next stage to be implemented is the deliberative stage. The retrieval algorithm is used to search the case-base and chose a case that is similar to the current case (i.e. the one that characterizes the problem to be solved). Each time a similar case is found, it is sent to the reuse phase where it is transformed into a suitable plan for the current problem by using a reuse algorithm. Therefore, at the end of each iteration of the deliberative stage, the TB-CBR method is able to provide a solution to the problem at hand, which may be improved in subsequent iterations if there is any time remaining at the deliberative stage.

The temporal cost of executing the cognitive task is greater than or equal to the sum of the execution times of the learning and deliberative stages (as shown in equation 1):

$$\begin{aligned} t_{cognitiveTask} &\geq t_{learning} + t_{deliberative} \\ t_{learning} &\geq (t_{revise} + t_{retain}) * n \\ t_{deliberative} &\geq (t_{retrieve} + t_{reuse}) * m \end{aligned} \quad \textbf{(1)}$$

where $t_{cognitiveTask}$ is the maximum time available for the agent to provide a response; $t_{learning}$ and $t_{deliberative}$ are respectively the total execution times of the learning and the deliberative stages; t_x is the execution time of phase x; and n and m are the number of iterations of the learning and deliberative stages, respectively.

The RTA can launch the TB-CBR algorithm when needed and if there is enough time to execute it. The maximum time available to complete the execution cycle (t_{max}, where $t_{max} \geq t_{cognitiveTask}$) must be stated. t_{max} must be split into the learning and the deliberative stages to guarantee the execution of each stage. The *timeManager* function is in charge of completing this task. Through this function, the designer specifies how the agent acts in the environment. The designer can assign more time to the learning stage, if an agent with a greater learning capacity is required. Otherwise, the function can allocate more time to the deliberation stage. Regardless of the type of agent, the *timeManager* function should allow enough time for each deliberative stage to ensure that at least one answer will be given by the stated deadline. Naturally, the greater the time allocated to the deliberative stages, the better the response, using an anytime algorithm that enables RTAs to refine the result of each iteration. The anytime behaviour of the TB-CBR mechanism is achieved through the use of two loop control sequences. The loop condition is built by using the *enoughTime* function, which determines if a new iteration can be performed according to the total time allocated to each stage of the TB-CBR.

The first phase of the algorithm executes the learning stage if the agent has the solutions from previous executions stored in the *solutionQueue*. The solutions are stored just after the end of the deliberative stage. The deliberative stage is only launched if there is a problem in the *problemQueue* that the agent cannot solve. This configuration allows the agent to launch the TB-CBR so that it only learns (no solution is needed and the agent has enough time to reason previous decisions), only deliberates (there are no previous solutions to consider and there is a new problem to solve) or so that it performs both functions.

For RT-MOVICAB-IDS to perform RT ID, a temporal constraint on the process (starting with a new generated segment and ending with the Analyzer agent generating the projection) is essential to ensure prompt execution. To perform this temporal control, all the steps in the process must be known and temporal bounded. Additionally, the system has to be deterministic. The deliberative agents within RT-MOVICAB-IDS agent environment (Coordinator and Analyzer agents) take advantage of the TB-CBR method to assign the pending analysis and complete the analysis in order to guarantee these conditions. The most relevant issues of these two agents are described in the following two sections.

3.3 Time-Bounded Coordinator Agent

The RT-MOVICAB-IDS Coordinator agent, in charge of assigning the pending analyses to the available Analyzer agents, is defined as a Case-Based Planning (CBP-BDI) agent [84]. CBP [85] attempts to solve new planning problems by reusing past successful plans [86]. The Coordinator agent plans to allocate an analysis to one of the available Analyzer agents based on the following criteria:

- **Location:** analyzer agents located in the network segment where the Visualizer or Pre-processor agents are placed would be prioritised.
- **Available resources:** the computer resources where each Analyzer agent is running and their rate of use all have to be taken into account in such a way that the workload of the computers is measured.
- **Analysis demands:** the amount and volume of data to be analysed are key issues to be considered.
- **Analyser agent behaviour:** these agents behave in a "learning" or "exploitation" mode. Learning behaviour causes an Analyzer agent to spend more time over an analysis than exploitation behaviour does.

As a computer network is an unstable environment, the availability of the Analyzer agents changes dynamically. Network links may stop working from time to time, so the Coordinator agent must be able to re-assign the analyses previously sent to the Analyzer agents located in the network segment that may be down at any time. These issues are included in the representation of cases, as indicated in Table 1.

Class	Feature	Type	Description
P	#packets	Integer	Total number of packets contained in the dataset to be analysed.
P	Analyzers / location	Array	An array (of variable length depending on the number of available Analyzer agents) indicating the network segment where the Analyzer agent is located.
P	Analyzers / features	Array	An array (of variable length depending on the number of available Analyzer agents) containing information about the resources, their availability, and pending tasks.
P	Analyzers / failures	Array	An array (of variable length depending on the number of available Analyzer agents) containing information about the number of times each Analyzer agent has stopped working in the recent past (execution failures).
S	Analyzers / plans	Array	An array (of variable length depending on the number of available Analyzer agents) containing the analyses assigned to each Analyzer agent.

Table 1. Coordinator agent-representation of case features. Classes: P (problem description attribute) and S (solution description attribute).

The Coordinator agent must provide a distribution of the analysis between the different Analyzer agents. In order to complete the analysis within the maximum predefined time, the Coordinator agent must apply CBR to generate the plan that best distributes the analysis and its allocation on time. Therefore, it is necessary to employ a temporal-bounded CBP, which is able to ensure compliance with the deadlines. Then, the Coordinator agent is modelled using a TB-CBP, which is a simple adaptation of the previously presented TB-CBR approach. On this occasion, the case-base stores previously executed and validated plans.

The four phases of the TB-CBP cycle of the Coordinator agent are re-defined to comply with the temporal constraints. As a solution must be provided within a preset time, the retrieval and reuse stages are initially performed. When a solution for the new problem is obtained, if no analysis is pending, the Coordinator agent executes the revise and retain stages. Consequently, the four phases are defined as follows:

- **(Plan) Retrieve**: when a new pre-processed dataset is ready, an analysis is requested from the Coordinator agent. The most similar plan is obtained by associative retrieval, taking into account the case/plan description shown in Table 2. As the time required to extract a case is predictable, this RTA knows how long it takes to arrive at the first solution. If there is some extra time before the deadline, the Coordinator agent will attempt to improve this first solution within the available time by allocating additional time to search for alternative plans. Once the time is finished, the best plan is used as a retrieved plan.
- **Reuse**: the retrieved plan is adapted to the new planning problem. The only restriction is that the analyses running at that time (the results of which have not yet been reported) cannot be reassigned. The others (pending) can be reassigned in order to optimize overall performance. The Coordinator agent knows when the adaptation of the cases to the new planning problem will finish. In this phase, as the Coordinator agent calculates when the Analyzer agents will complete their assigned tasks, it also knows that it can continue building the new plan, because the Analyzer agents will still be executing pending analyses when this phase is completed. Thus, the new assignment of an analysis to an Analyzer agent depends on its workload at that particular time.
- **Revise**: the plan revision consists of a two-fold analysis. On the one hand, planning failures are identified by finding under-exploited resources. As an example, the following hypothetical situation is identified as a planning failure: one of the Analyzer agents is not performing any

- **Retain**: when a plan is adopted, the Coordinator agent stores a new case containing the dataset-descriptor and the solution (see Table 1).

3.4 Time-bounded Analyzer Agent

The Analyzer agent is a temporal-bounded, hybrid deliberative agent. It employs the CMLHL neural model to analyse pre-processed traffic data. In other words, this neural projection model is applied to reduce the dimensionality of the captured segments and generate subsequent visualizations of them.

The CMLHL model is based on Maximum Likelihood Hebbian Learning (MLHL) [87]. Considering an N-dimensional input vector (x), and an M-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:

$$y_i = \sum_{j=1}^{\mathbf{N}} W_{ij} x_j, \forall i . \qquad \textbf{(2)}$$

2. Lateral activation passing:

$$y_i(t+1) = [y_i(t) + \tau(b - Ay)]^+ . \qquad \textbf{(3)}$$

3. Feedback step:

$$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i, \forall j . \qquad \textbf{(4)}$$

4. Weight change:

$$\Delta W_{ij} = \eta . y_i . sign(e_j) | e_j |^{p-1} . \qquad \textbf{(5)}$$

Where: η is the learning rate, τ is the "strength" of the lateral connections, b the bias parameter, p a parameter related to the energy function [76, 87] and A a symmetric matrix used to modify the response to the data [76]. The effect of this matrix is based on the relation between the distances separating the output neurons.

This agent also incorporates an intelligent paradigm (TB-CBR) to tune the parameters of the neural model introduced above. This agent generates a solution (or achieves its goals) by retrieving a previous case and analysing the new one through the CMLHL architecture. Cases are defined by several features, as can be seen in Table 2.

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 on 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.
S	Weights	Matrix	A matrix containing the synaptic weights calculated by the CMLHL model after training.

Table 2. Analyzer agent - representation of case features. Classes: P (problem description attribute) and S (solution description attribute).

The Analyzer agent incorporates two different behaviours, namely "learning" and "exploitation". As previously described, this agent initially incorporates new knowledge (modelled as sets of problem/solution) into the case base during the set-up stage. This learning behaviour is characterized by the TB-CBR stages described below, through which the agent stores the results on previous similar datasets to generate the parameter values of a new problem in the future. Once the case base is wide enough (according to different criteria), the exploitation behaviour is started. From then on, the revise and retain stages of the cycle are no longer performed as there is a wide range of previous cases already stored in the case base. When a new analysis request arrives, the Analyzer agent retrieves the most similar case stored in the case base. Then, the weights contained in that solution are reused to project the new data. To reduce the execution time, the neural network is not trained again and as a result, the other parameters of the neural model are not reused.

The Analyzer is clearly the most resource-consuming class of RT-MOVICAB-IDS agents as it trains the neural model during the learning behaviour. The amount of computational resources needed to analyze the data coming from different network segments is extremely high. To respond to this demand, Analyzer agents can be located in high-performance computing clusters or in less powerful machines whose computing resources are under-used. In this way, RT-MOVICAB-IDS can be adapted to the available resources for ID. Additionally, time-bounding these agents will cause a reduction of the response time (especially in the worst case) while reducing the amount of considered solutions. As a consequence, less training of the neural model during the learning behaviour will ensure that the Analyzer Agent is capable of quickly obtaining a result in a deterministic way.

To do so, the Analyzer agent implements a temporal bounded behaviour in all of its phases. As a consequence, the result of the training will be improved when extra time is available to complete this phase.

The different stages of the TB-CBR applied by the Analyzer agent of RT-MOVICAB-IDS can be defined as follows:

- **Retrieve and Reuse phases**: when a new analysis is requested, the Analyzer agent tries to find the most similar case to the new one in the case base and it is reused to obtain a solution (the values of the parameters used to train the CMLHL model). Theses phases are implemented by means of the *anytime* algorithm. This algorithm extracts a solution in a known amount of time, smaller than the one available to complete these phases. In the reuse phase, a set of trainings for the CMLHL neural model are defined by combining the different parameter values recovered from the cases in the case base. As the number of iterations of each one is known, so too is the training time. As a result, the Analyzer agent can predict how many neural network models could be built in the available time.
- **Revise and Retain phases**: the revise and retain phases depend on human experience which means that strict temporal control is not applicable to these decisions. For this reason, these phases are completed offline. Once the human expert performs a visual analysis of the segment, one of the projections is selected and the related parameters are stored in the case-base for future executions. The time required by a human expert to perform this action is variable and indeterminate. As a consequence, these phases lie outside the RT decision algorithm used by the Analyzer agents.

4 Experiments and Results

RT-MOVICAB-IDS has been tested on a real-life network by generating several segments. Experiments on each segment have been carried out. For the sake of brevity, this section shows only some of the results obtained through these experiments.

Due to its vulnerabilities [14], anomalous situations concerning the Simple Network Management Protocol (SNMP) are targeted in the experimental setting of this study. SNMP is oriented to manage nodes in the Internet community [88]; it is used to control routers, bridges, and some other network elements, reading and writing a wide variety of information (such as operating system, version, routing tables, default TTL and so on) on these devices. All this information is stored in the Management Information Base (MIB), so it can be defined in broad terms as the database used by SNMP to store information about the elements that it controls. In addition to the SNMP packets (both "normal" and "anomalous"), the segments contain traffic related to other protocols, considered as "normal" traffic.

4.1 SNMP Anomalous Situations

This experimental study is focused on the identification of SNMP-related attacks. Thus, three main anomalous situations are distributed throughout the different segments in this study, namely: scans, SNMP community searches and MIB information transfers. These situations (described in the following paragraphs) can be very risky on their own and all together (a network scan followed by an SNMP community search and ending with an MIB information transfer) constitute a complete SNMP attack in which an intruder obtains SNMP managed information without possessing any previous knowledge about the network under attack.

These three anomalous situations can be defined as:

- Scans. A port scan may be defined as series of messages sent to different port numbers of a host to gain information on its activity status. These messages could be sent by an external agent to find out more about the network services a host is providing. On the contrary, in a network scan the same port is the target for a number of hosts (usually all the hosts in an IP address range). A port scan provides information on where to probe for weaknesses, for which reason scanning generally precedes any further intrusive activity. In this experimental study, the datasets contain network scans aimed at port numbers 1,434 (registered port assigned to Microsoft-SQL-Monitor, the target of the W32.SQLExp.Worm) and 65,788 (as an example of a dynamic or private port).

- SNMP Community Search. The unencrypted "community string" can be seen as the SNMP password for versions 1 and 2. An SNMP community search is characterized by the intruder sending SNMP queries to the same port number of different hosts trying to guess the SNMP community string by means of different strategies (brute force, dictionary, etc.) [89]. Once the community string has been obtained, all the information stored in the MIB is available for the intruder.
- MIB Information Transfer. This situation is a transfer of some (or all the) information contained in the SNMP MIB, generally through the *get* (or *get-bulk*) command. This kind of transfer is potentially a dangerous situation. However, the "normal" behaviour of a network may include queries to the MIB. This is a situation in which visualization-based IDSs are especially useful; these situations are visualized in a "special" way by the IDS, but it is the network administrator's responsibility to decide whether or not it is a "normal" (i.e. a previously scheduled) MIB transfer.

4.2 RT-MOVICAB-IDS Projections

In this section, some snapshots are shown. Each one of them depicts all the packets contained in the dataset whose projection is shown. RT-MOVICAB-IDS plots the packets in different colours and shapes taking into account the protocol information, leading to an intuitive visualization. In these snapshots, and in general for projection models, the axes forming the projections are combinations of the features contained in the original datasets, as shown in Fig. 2. The horizontal and vertical axes of the projections are not associated with a unique original feature.

Fig. 2 shows the projection of a simple segment containing no anomalous situations. This is the way that RT-MOVICAB-IDS depicts "normal" traffic: parallel straight lines. After analysing each packet that is depicted, it was noticed that a certain ordering related to the input variables is preserved in this and other projections. The original features (timestamp, source port, destination port, protocol and size) of the packets are preserved as indicated in Fig. 2. Any sign of non-parallel evolution or high packet concentration is viewed as an anomaly. It can be seen how in this figure all the packets (related to "normal" traffic) evolve in parallel "lines". For some protocols, we cannot define a proper "line" because there are not enough packets. We can draw a line crossing all these packets (from the same protocol) in the plot. This line will be then parallel to the others.

Additionally, Fig. 2 allows us to identify a disruption in protocol traffic. As can be seen in this figure, the normal traffic related to a certain protocol (Group 1 in Fig. 2) is interrupted at a certain point. Thus, the network administrator should realise that this protocol stopped working for a while. This would require an in-depth investigation to ascertain the reasons for such an interruption, as it might not be related to an intrusion.

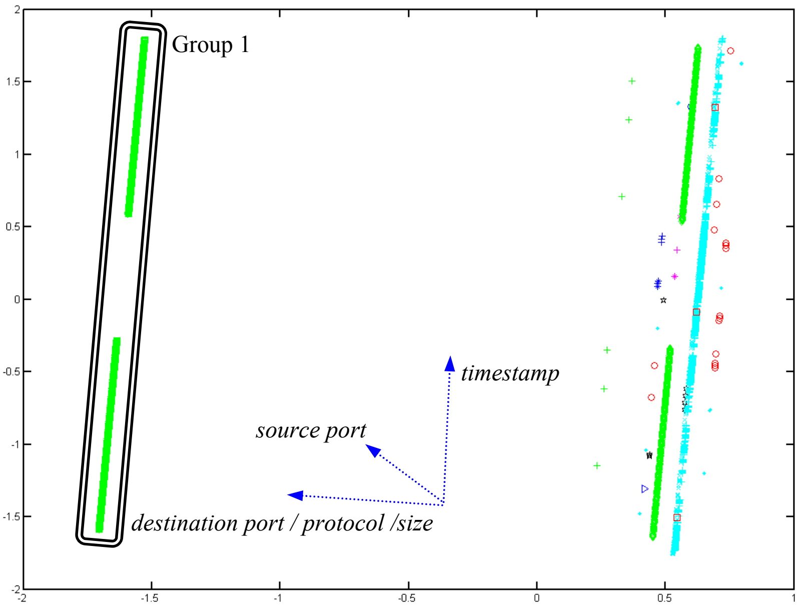


Fig. 2. RT-MOVICAB-IDS sample visualization of “normal” traffic.

RT-MOVICAB-IDS visualisation of a simple segment containing two network scans is shown in Fig. 3. As in the previous segment, most of the traffic (identified as "normal") evolves in parallel straight lines. Additionally, there are two "groups" (Groups 1 and 2) of packets that are not depicted in the same way. Looking at the source data, it was checked that all these packets (visualised in a non-parallel line to normal traffic) formed part of the network scans contained in this segment. Packets contained in Group 1 were related to a network scan aimed at port number 1434, while packets contained in Group 2 made up the scan aimed at port number 65788.

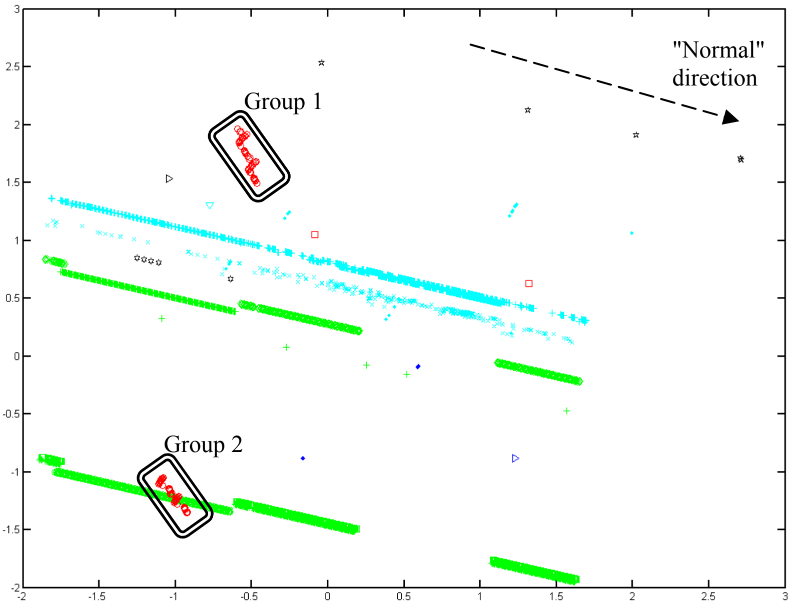


Fig. 3. RT-MOVICAB-IDS sample visualization of a simple segment containing two network scans.

Fig. 4 shows the way in which the system visualizes an accumulated segment containing several anomalous situations [12-14]: network scans (Group 1), SNMP community searches (Groups 2, 3, and 4), and MIB information transfers (Groups 5 to 8). These anomalous situations are identified by their non-parallel evolution and their high packet concentrations. Although these anomalous situations are placed in a 106 minute-long accumulated segment containing almost 50,000 packets, they do not slip by unnoticed. This outcome shows the intrinsic robustness of the applied neural model (CMLHL), which is able to respond effectively to a complex dataset.

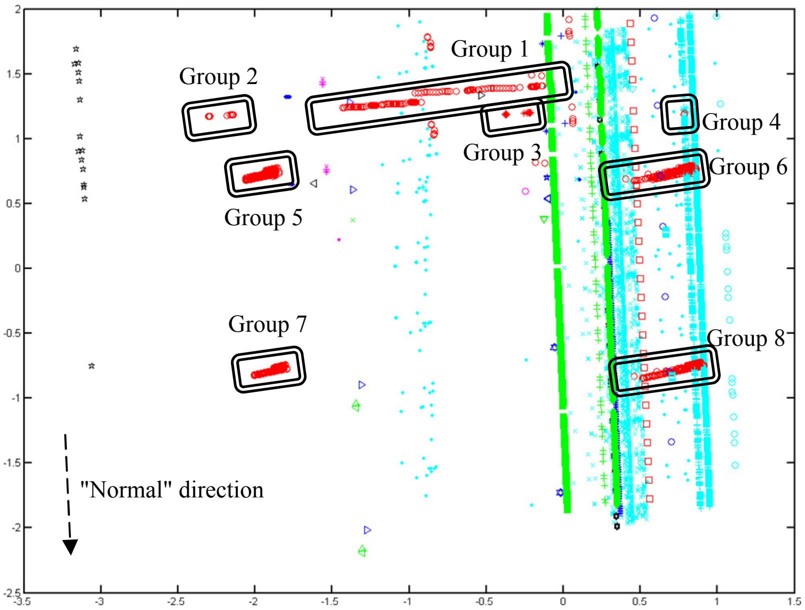


Fig. 4. RT-MOVICAB-IDS sample visualization of an accumulated segment containing several different anomalous situations.

Some other statistical and unsupervised models such as Principal Component Analysis (PCA) [17], Curvilinear Component Analysis [90] or Self-Organizing Maps [91] have been applied to analyze the internal structure of these traffic datasets. Nevertheless, CMLHL [14] was able to identify anomalous situations in the most intuitive way.

4.3 Results for the Coordinator Agent

This section shows the results of time-bounding the Coordinator Agent. The main advantages of using the TB-CBP, as against CBP without temporal constraints, are the maximization of CPU utilization and minimization of the average execution time of the distribution of the analyses to the Analyzer agents. A set of tests was performed to validate this claim, the results of which are shown in Table 3. One hundred tests were completed, each consisting of a set of segments that the Coordinator agent should distribute to available Analyzer agents. The maximum time for planning was two milliseconds and the Coordinator agent therefore had to complete the distribution of pending analysis before that time expired. As the Coordinator agent has been implemented using the TB-CBP model, it is able to complete this task (providing a plan) while meeting the temporal constraints. Additionally, the utilization of the CPU resources is maximized as can be seen in Table 3.

	Average CPU utilization	Average Execution Time
TB-CBP	97 %	1.6 ms
CBP	72 %	3.4 ms

Table 3. TB-CBP vs. CBP as reasoning mechanism of the Coordinator agent.

4.4 Results for the Analyzer Agent

The percentage of completed CMLHL trainings for different response times were evaluated, in order to measure the consequences of applying the TB-CBR model instead of CBR in the formulation of the Analyzer Agent. Table 4 shows the results obtained by considering both simple and accumulated segments. As can be seen, using the new TB-CBR-based approach, the percentage of completed trainings is reduced due to the time limitation. Nevertheless, this reduction of trainings does not strongly worsen the quality of the obtained projections of the pre-processed traffic data, according to the perception of the human experts. It can be said that the main outcome of the new temporal-bounded approach is that it works in a faster and, what is more important, a more predictable way.

	Segment Type	Completed Trainings
CBR	Simple	100 %
	Accumulated	100 %
TB-CBR	Simple	58%
	Accumulated	40%

Table 4. TB-CBR vs. CBR as reasoning mechanism of the Analyzer agents.

5 Conclusions and Future Work

This study has presented RT-MOVICAB-IDS, and IDS incorporating temporal constraints on the deliberative agents that employ a CBR architecture, which enables them to respond to events in real-time. As a consequence, the Coordinator and Analyzer agents will always give a solution within the available time, thereby maximizing CPU utilization.

The deliberative Coordinator agent, working at a high level with Belief-Desire-Intention (BDI) concepts, is temporal-bounded by redefining the four stages of its CBP cycle employing the TB-CBP model. This means that the deliberative process of the Coordinator agent is predictable, so that the Coordinator agent knows how much time is available to provide a solution. The Coordinator agent can obtain the best solution in the time allotted for this purpose. Moreover, the deliberative process time is reduced and the CPU utilization by the Coordinator agent increases.

A key step of the ID process is the assignement of each pending analysis to available Analyzer agents, which is performed by the Coordinator agent. Accordingly, temporal constraints are incorporated in the Coordinator agent without affecting its deliberative capabilities.

The Analyzer agent also incorporates time restrictions. In this case, the TB-CBR included in this kind of agents allows them to provide a faster response. The main drawback is that the number of trainings in the learning phases of this agent is reduced. However, the visualisation does not significantly deteriorate, so the visual analysis performance is similar but the visualization is obtained much earlier. The Analyzer agent is predictable which allows a temporal bounded analysis of the pre-processed traffic data.

With these RT features, the tasks of RT-MOVICAB-IDS that employ intelligent techniques are converted into predictable tasks, and therefore, the global response of the system can be assured within a maximum amount of time. Moreover, the use of an anytime approximation, as it has been described in previous sections, allows an improved response quality if more time is available in order to obtain the expected solution. As a consequence, the end result is a trade-off between fast detection of intrusions and their accurate identification.

From a general perspective, it can be concluded that the proposed RT-MOVICAB-DIS formulation enables a predictable and intuitive visualization of network traffic, including normal and anomalous situations. As a result, security personnel employing this tool will be able to monitor the traffic of a given network.

Future work will be based on the application of different neural models in order to obtain better visualization results within the temporal constraints of RT-MOVICAB-IDS. Additionally, some other MAS issues, such as failure and attack tolerance, will be considered.

Acknowledgments. This research is funded through the Junta of Castilla and León (BU006A08); the Spanish Ministry of Science and Innovation (TIN2010-21272-C02-01), Education and Innovation (CIT-020000-2008-2 and CIT-020000-2009-12); the Spanish government (TIN2009-13839-C03-01), FEDER and CONSOLIDER-INGENIO (2010 CSD2007-00022) and the Generalitat Valenciana (PROMETEO/2008/051). The authors would also like to thank the vehicle interior manufacturer, Grupo Antolin Ingenieria S.A. for supporting the project through the MAGNO2008 - 1028.- CENIT Project funded by the Spanish Ministry of Science and Innovation.

References

[1] J.M. Myerson, Identifying Enterprise Network Vulnerabilities, International Journal of Network Management, 12 (2002) 135-144.

[2] D. Moore, V. Paxson, S. Savage, C. Shannon, S. Staniford, N. Weaver, The Spread of the Sapphire/Slammer Worm, in: Technical Report, International Computer Science Institute - University of California at Berkeley, 2003.

[3] H. Kopetz, Real-time Systems: Design Principles for Distributed Embedded Applications, Kluwer Academic Publishers, 1997.

[4] W. Wang, X. Guan, X. Zhang, Processing of Massive Audit Data Streams for Real-time Anomaly Intrusion Detection, Computer Communications, 31 (2008) 58-72.

[5] S.E. Schechter, J. Jung, A.W. Berger, Fast Detection of Scanning Worm Infections, in: 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004), Springer Berlin / Heidelberg, 2004, pp. 59-81.

[6] T. Dean, M. Boddy, An Analysis of Time-dependent Planning, in: 7th National Conference on Artificial Intelligence, 1988, pp. 49–54.

[7] A. Garvey, V. Lesser, A Survey of Research in Deliberative Real-time Artificial Intelligence, Real-Time Systems, 6 (1994) 317-347.

[8] B. Hayes-Roth, R. Washington, D. Ash, A. Collinot, A. Vina, A. Seiver, Guardian: A Prototype Intensive-care Monitoring Agent, Artificial Intelligence in Medicine, 4 (1992) 165-185.

[9] A.E. Howe, D.M. Hart, P.R. Cohen, Addressing Real-time Constraints in the Design of Autonomous Agents, Real-Time Systems, 2 (1990) 81-97.

[10] D.J. Musliner, E.H. Durfee, K.G. Shin, CIRCA: A Cooperative Intelligent Real-time Control Architecture, IEEE Transactions on Systems, Man, and Cybernetics, 23 (1993) 1561 - 1574.

[11] V. Julian, V. Botti, Developing Real-time Multi-agent Systems, Integrated Computer-Aided Engineering, 11 (2004) 135-149.

[12] Á. Herrero, E. Corchado, J.M. Sáiz, MOVICAB-IDS: Visual Analysis of Network Traffic Data Streams for Intrusion Detection, in: E. Corchado, Yin, H., Botti, V., Fyfe, C. (Ed.) 7th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2006), Springer, Heidelberg, 2006, pp. 1424-1433.

[13] Á. Herrero, E. Corchado, Mining Network Traffic Data for Attacks through MOVICAB-IDS, in: Foundations of Computational Intelligence, Springer, 2009, pp. 377-394.

[14] E. Corchado, Á. Herrero, Neural Visualization of Network Traffic Data for Intrusion Detection, Applied Soft Computing, ("Accepted - In press") (2010).

[15] F.-Y. Leu, C.-T. Yang, F.-C. Jiang, Improving Reliability of a Heterogeneous Grid-based Intrusion Detection Platform using Levels of Redundancies, Future Generation Computer Systems, 26 554-568.

[16] A. Aamodt, E. Plaza, Case-Based Reasoning - Foundational Issues, Methodological Variations, and System Approaches, AI Communications, 7 (1994) 39-59.

[17] H. Hotelling, Analysis of a Complex of Statistical Variables into Principal Components, Journal of Education Psychology, 24 (1933) 417-444.

[18] K. Pearson, On Lines and Planes of Closest Fit to Systems of Points in Space, Philosophical Magazine, 2 (1901) 559-572.

[19] E. Oja, Neural Networks, Principal Components, and Subspaces, International Journal of Neural Systems, 1 (1989) 61-68.

[20] J.H. Friedman, J.W. Tukey, A Projection Pursuit Algorithm for Exploratory Data-Analysis, IEEE Transactions on Computers, 23 (1974) 881-890.

[21] A. Abraham, C. Grosan, C. Martin-Vide, Evolutionary Design of Intrusion Detection Programs, International Journal of Network Security, 4 (2007) 328-339.

[22] K. Julisch, Data Mining for Intrusion Detection: A Critical Review, in: D. Barbará, S. Jajodia (Eds.) Applications of Data Mining in Computer Security, Kluwer Academic Publishers, 2002, pp. 33-62.

[23] T. Chih-Fong, H. Yu-Feng, L. Chia-Ying, L. Wei-Yang, Intrusion Detection by Machine Learning: A Review, Expert Systems with Applications, 36 (2009) 11994-12000.

[24] A. Tajbakhsh, M. Rahmati, A. Mirzaei, Intrusion Detection using Fuzzy Association Rules, Applied Soft Computing, 9 (2009) 462-469.

[25] A. Abraham, R. Jain, J. Thomas, S.Y. Han, D-SCIDS: Distributed Soft Computing Intrusion Detection System, Journal of Network and Computer Applications, 30 (2007) 81-98.

[26] S. Zanero, S. Savaresi, Unsupervised Learning Techniques for an Intrusion Detection System, in: ACM Symposium on Applied Computing, 2004, pp. 412-419.

[27] C. Zhang, J. Jiang, M. Kamel, Intrusion Detection using Hierarchical Neural Networks, Pattern Recognition Letters, 26 (2005) 779-791.

[28] Á. Herrero, E. Corchado, P. Gastaldo, R. Zunino, Neural Projection Techniques for the Visual Inspection of Network Traffic, Neurocomputing, 72 (2009) 3649-3658.

[29] D.J. Marchette, Computer Intrusion Detection and Network Monitoring: A Statistical Viewpoint, Springer-Verlag New York, Inc., 2001.

[30] M. Roesch, Snort–Lightweight Intrusion Detection for Networks, in: 13th Systems Administration Conference (LISA '99), 1999, pp. 229-238.

[31] C. Ahlberg, B. Shneiderman, Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays, in: Readings in Information Visualization: using Vision to Think, Morgan Kaufmann Publishers Inc., 1999, pp. 244-250.

[32] K. Lakkaraju, W. Yurcik, A.J. Lee, NVisionIP: Netflow Visualizations of System State for Security Situational Awareness, in: 2004 ACM Workshop on Visualization and Data Mining for Computer Security, ACM, Washington DC, USA, 2004, pp. 65-72.

[33] S.T. Teoh, K.L. Ma, S.F. Wu, X. Zhao, Case Study: Interactive Visualization for Internet Security, in: IEEE Conference on Visualization (Vis 2002), IEEE Computer Society, Boston, Massachusetts, 2002.

[34] K. Nyarko, T. Capers, C. Scott, K.A. Ladeji-Osias, Network Intrusion Visualization with NIVA, an Intrusion Detection Visual Analyzer with Haptic Integration, in: T. Capers (Ed.) 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (HAPTICS 2002), 2002, pp. 277-284.

[35] J.R. Goodall, W.G. Lutters, P. Rheingans, A. Komlodi, Preserving the Big Picture: Visual Network Traffic Analysis with TNV, in: IEEE Workshop on Visualization for Computer Security (VizSEC 05), IEEE Computer Society, 2005, pp. 47-54.

[36] H. Koike, K. Ohno, K. Koizumi, Visualizing Cyber Attacks Using IP Matrix, in: IEEE Workshop on Visualization for Computer Security (VizSEC 05), IEEE Computer Society, 2005, pp. 91-98.

[37] C. Muelder, K.L. Ma, T. Bartoletti, Interactive Visualization for Network and Port Scan Detection, in: D. Zamboni, C. Kruegel (Eds.) Nineth International Symposium on Recent Advances in Intrusion Detection (RAID 2006), Springer, Heidelberg, 2006, pp. 265-283.

[38] A. Kulsoom, C. Lee, G. Conti, J.A. Copeland, Visualizing Network Data for Intrusion Detection, in: Sixth Annual IEEE Information Assurance Workshop - Systems, Man and Cybernetics (SMC) , 2005, 2005, pp. 100-108.

[39] MRTG: The Multi Router Traffic Grapher, http://www.mrtg.org. Last access: 18/10/2010

[40] P. Ren, Y. Gao, Z.C. Li, Y. Chen, B. Watson, IDGraphs: Intrusion Detection and Analysis Using Stream Compositing, IEEE Computer Graphics and Applications, 26 (2006) 28-39.

[41] H. Koike, K. Ohno, SnortView: Visualization System of Snort Logs, in: 2004 ACM Workshop on Visualization and Data Mining for Computer Security, ACM Press, Washington DC, USA, 2004, pp. 143-147.

[42] K. Abdullah, C.P. Lee, G. Conti, J.A. Copeland, J. Stasko, IDS RainStorm: Visualizing IDS Alarms, in: IEEE Workshop on Visualization for Computer Security (VizSEC 05), IEEE Computer Society, 2005, pp. 1-10.

[43] A.D. D'Amico, J.R. Goodall, D.R. Tesone, J.K. Kopylec, Visual Discovery in Computer Network Defense, IEEE Computer Graphics and Applications, 27 (2007) 20-27.

[44] A. Komlodi, P. Rheingans, A. Utkarsha, J.R. Goodall, J. Amit, A User-Centered Look at Glyph-Based Security Visualization, in: IEEE Workshop on Visualization for Computer Security (VizSEC 05), IEEE Computer Society, 2005, pp. 21-28.

[45] A. Chuvakin, Monitoring IDS, Information Security Journal: A Global Perspective, 12 (2004) 12 - 16.

[46] P. Diaconis, D. Freedman, Asymptotics of Graphical Projection Pursuit, The Annals of Statistics, 12 (1984) 793-815.

[47] R.F. Erbacher, Visual Traffic Monitoring and Evaluation, in: Conference on Internet Performance and Control of Network Systems II, 2001, pp. 153–160.

[48] F. Mansmann, D.A. Keim, S.C. North, B. Rexroad, D. Sheleheda, Visual Analysis of Network Traffic for Resource Planning, Interactive Monitoring, and Interpretation of Security Threats, IEEE Transactions on Visualization and Computer Graphics, 13 (2007) 1105-1112.

[49] P. Ren, Y. Gao, Z.C. Li, Y. Chen, B. Watson, IDGraphs: Intrusion Detection and Analysis Using Histographs, in: IEEE Workshop on Visualization for Computer Security (VizSEC 05), IEEE Computer Society, 2005, pp. 39-46.

[50] R.A. Becker, S.G. Eick, A.R. Wilks, Visualizing Network Data, IEEE Transactions on Visualization and Computer Graphics, 1 (1995) 16-28.

[51] A. Oline, D. Reiners, Exploring Three-Dimensional Visualization for Intrusion Detection, in: IEEE Workshop on Visualization for Computer Security (VizSEC 05), IEEE Computer Society, 2005, pp. 113-120.

[52] D. Dasgupta, F. Gonzalez, K. Yallapu, J. Gomez, R. Yarramsettii, CIDS: An Agent-based Intrusion Detection System, Computers & Security, 24 (2005) 387-398.

[53] I.M. Hegazy, T. Al-Arif, Z.T. Fayed, H.M. Faheem, A Multi-agent Based System for Intrusion Detection, IEEE Potentials, 22 (2003) 28-31.

[54] E.H. Spafford, D. Zamboni, Intrusion Detection Using Autonomous Agents, Computer Networks: The International Journal of Computer and Telecommunications Networking, 34 (2000) 547-570.

[55] Cougaar: Cognitive Agent Architecture, http://cougaar.org/. Last access: 18/10/2010

[56] K. Deeter, K. Singh, S. Wilson, L. Filipozzi, S. Vuong, APHIDS: A Mobile Agent-Based Programmable Hybrid Intrusion Detection System, in: First International Workshop on Mobility Aware Technologies and Applications (MATA 2004), Springer, Heidelberg, 2004, pp. 244-253.

[57] G. Kolaczek, A. Pieczynska-Kuchtiak, K. Juszczyszyn, A. Grzech, R.P. Katarzyniak, N.T. Nguyen, A Mobile Agent Approach to Intrusion Detection in Network Systems, in: Knowledge-Based Intelligent Information and Engineering Systems, Springer, Heidelberg, 2005, pp. 514-519.

[58] A.J. Garvey, V.R. Lesser, Design-to-Time Real-Time Scheduling, IEEE Transactions on Systems, Man and Cybernetics, 23 (1993) 1491-1502.

[59] T. Wagner, V. Lesser, Design-to-Criteria Scheduling: Real-Time Agent Control, in: Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems, Springer Verlag, 2001, pp. 128-143.

[60] J.R. Graham, K. Decker, Towards a Distributed, Environment-Centered Agent Framework, in: 6th International Workshop on Intelligent Agents VI, Agent Theories, Architectures, and Languages (ATAL), Springer-Verlag, 2000, pp. 290-304.

[61] L.C. DiPippo, E. Hodys, B. Thuraisingham, Towards a Real-Time Agent Architecture: A Whitepaper, in: Fifth International Workshop on Object-Oriented Real-Time Dependable Systems, 1999, pp. 59-64.

[62] L. Cingiser DiPippo, V. Fay-Wolfe, L. Nair, E. Hodys, O. Uvarov, A Real-Time Multi-Agent System Architecture for e-Commerce Applications, in: 5th International Symposium on Autonomous Decentralized Systems, 2001, pp. 357-364.

[63] Real-time CORBA, http://www.omg.org/technology/documents/formal/real-time_CORBA.htm. Last access: 18/10/2010

[64] D.M. Surka, M.C. Brito, C.G. Harvey, The Real-time ObjectAgent Software Architecture for Distributed Satellite Systems, in: IEEE Aerospace Conference 2001, 2001, pp. 2731-2741.

[65] K. Prouskas, J. Pitt, Towards a Real-time Architecture for Time-aware Agents, in: First International Joint Conference on Autonomous Agents and Multiagent Systems, ACM, Bologna, Italy, 2002, pp. 92-93.

[66] V.J. Botti, C. Carrascosa, V. Julián, J. Soler, Modelling Agents in Hard Real-Time Environments, in: 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World: MultiAgent System Engineering, Springer-Verlag, 1999, pp. 63-76.

[67] P. Nit, Blackboard Systems: the Blackboard Model of Problem-Solving and the Evolution of Blackboard Architecture, AI Magazine, 7 (1986) 38-53.

[68] J. Li, C. Manikopoulos, Early Statistical Anomaly Intrusion Detection of DOS Attacks using MIB Traffic Parameters, in: IEEE Systems, Man and Cybernetics Society Information Assurance Workshop, IEEE, New York, 2003, pp. 53-59.

[69] S. Rawat, A.K. Pujari, V.P. Gulati, On the Use of Singular Value Decomposition for a Fast Intrusion Detection System, in: First International Workshop on Views on Designing Complex Architectures (VODCA 2004) 2006, pp. 215-228.

[70] H.M. El-Bakry, N. Mastorakis, A Real-time Intrusion Detection Algorithm for Network Security, WSEAS Transactions on Communications, 7 (2008) 1222-1234.

[71] X. Guan, W. Wang, X. Zhang, Fast Intrusion Detection based on a Non-negative Matrix Factorization Model, Journal of Network and Computer Applications, 32 (2009) 31-44.

[72] W. Jiang, H. Song, Y. Dai, Real-time Intrusion Detection for High-speed Networks, Computers & Security, 24 (2005) 287-294.

[73] S. Babu, L. Subramanian, J. Widom, A Data Stream Management System for Network Traffic Management, in: Workshop on Network-Related Data Management (NRDM 2001), 2001.

[74] Á. Herrero, E. Corchado, Traffic Data Preparation for a Hybrid Network IDS, in: Third International Workshop on Hybrid Artificial Intelligence Systems (HAIS 2008), Springer, Heidelberg, 2008, pp. 247-256.
[75] H. Dreger, A. Feldmann, V. Paxson, R. Sommer, Operational Experiences with High-Volume Network Intrusion Detection, in: 11th ACM Conference on Computer and Communications Security, ACM Press New York, 2004, pp. 2-11.
[76] E. Corchado, C. Fyfe, Connectionist Techniques for the Identification and Suppression of Interfering Underlying Factors, International Journal of Pattern Recognition and Artificial Intelligence, 17 (2003) 1447-1466.
[77] Á. Herrero, E. Corchado, M.A. Pellicer, A. Abraham, MOVIH-IDS: A Mobile-Visualization Hybrid Intrusion Detection System, Neurocomputing, 72 (2009) 2775-2784.
[78] R.L. King, S. H. Russ, A. B. Lambert, D. S. Reese, An Artificial Immune System Model for Intelligent Agents, Future Generation Computer Systems, 17 (2001) 335-343.
[79] C. Carrascosa, J. Bajo, V. Julián, J.M. Corchado, V. Botti, Hybrid Multi-agent Architecture as a Real-Time Problem-Solving Model, Expert Systems with Applications: An International Journal, 34 (2008) 2-17.
[80] M.E. Bratman, Intentions, Plans and Practical Reason, Harvard University Press, Cambridge, M.A., 1987.
[81] F. Zambonelli, N.R. Jennings, M. Wooldridge, Developing Multiagent Systems: the Gaia Methodology, ACM Transactions on Software Engineering and Methodology, 12 (2003) 317-370.
[82] M. Wooldridge, N.R. Jennings, D. Kinny, The Gaia Methodology for Agent-Oriented Analysis and Design, Autonomous Agents and Multi-Agent Systems, 3 (2000) 285-312.
[83] M. Navarro, S. Heras, V. Julián, Guidelines to Apply CBR in Real-Time Multi-Agent Systems, Journal of Physical Agents, 3 (2009) 39-43.
[84] J. Bajo, J. Corchado, S. Rodríguez, Intelligent Guidance and Suggestions Using Case-Based Planning, in: Case-Based Reasoning Research and Development, Springer, Heidelberg, 2007, pp. 389-403.
[85] K.J. Hammond, Case-based Planning: Viewing Planning as a Memory Task, Academic Press Professional, Inc., 1989.
[86] L. Spalzzi, A Survey on Case-Based Planning, Artificial Intelligence Review, 16 (2001) 3-36.
[87] E. Corchado, D. MacDonald, C. Fyfe, Maximum and Minimum Likelihood Hebbian Learning for Exploratory Projection Pursuit, Data Mining and Knowledge Discovery, 8 (2004) 203-225.
[88] J. Case, M.S. Fedor, M.L. Schoffstall, C. Davin, Simple Network Management Protocol (SNMP), in: IETF RFC 1157, 1990.
[89] S. Northcutt, M. Cooper, K. Fredericks, M. Fearnow, J. Riley, Intrusion Signatures and Analysis, New Riders Publishing Thousand Oaks,, 2001.
[90] P. Demartines, J. Herault, Curvilinear Component Analysis: A Self-Organizing Neural Network for Nonlinear Mapping of Data Sets, IEEE Transactions on Neural Networks, 8 (1997) 148-154.
[91] T. Kohonen, The Self-Organizing Map, Proceedings of the IEEE, 78 (1990) 1464-1480.