International Journal of Computational Intelligence and Applications
Vol. 14, No. 4 (2015) 1550025 (35 pages)

DOI: 10.1142/S146902681550025X

Imperial College Press
www.icpress.co.uk

Survey of Uses of Evolutionary Computation Algorithms and Swarm Intelligence for Network Intrusion Detection

Saber Elsayed*, Ruhul Sarker† and Daryl Essam‡
School of Engineering and Information Technology
University of New South Wales at Canberra, Canberra, Australia
**s.elsayed@adfa.edu.au*
†*r.sarker@adfa.edu.au*
‡*d.essam@adfa.edu.au*

Received 14 August 2014
Revised 7 July 2015
Accepted 26 October 2015
Published 18 December 2015

Many infrastructures, such as those of finance and banking, transportation, military and telecommunications, are highly dependent on the Internet. However, as the Internet's underlying structural protocols and governance can be disturbed by intruders, for its smooth operation, it is important to minimize such disturbances. Of the available techniques for achieving this, computational intelligence methodologies, such as evolutionary algorithms and swarm intelligence approaches, are popular and have been successfully applied to detect intrusions. In this paper, we present an overview of these techniques and related literature on intrusion detection, analyze their research contributions, compare their approaches and discuss new research directions which will provide useful insights for intrusion detection researchers and practitioners.

Keywords: Cybersecurity; intrusion detection; evolutionary algorithms; swarm intelligence.

1. Introduction

The field of computer security covers many areas, such as cryptography, authentication, intrusion detection (ID), forensics and auditing, and risk and vulnerability assessments.[1] It can be described as protecting computing systems, i.e., the Internet, against confidentiality, integrity and availability threats.[2] Although the Internet provides many pros, such as making our world seem smaller, it has numerous cons, such as being subject to fraud attacks and security threats, which have become easier to execute and are of many kind. Therefore, ID systems (IDSs) have attracted the attention of many researchers.[3,4] An IDS aims to classify the illegitimate and malicious activities in a computer system that compromises the confidentiality, integrity and availability of information resources. However, ID is still a challenging problem

*Corresponding author.

because of the proliferation of diverse computer networks with increased connectivity, which makes it easy for intruders to avoid identification.

Many approaches for designing an IDS, such as expert systems, have been presented. Expert systems permit the incorporation of an extensive amount of human experience into a computer application that then utilizes that knowledge to identify activities that match the defined characteristics of an attack.[5] However, they have some disadvantages[5]: (i) they require frequent updates to remain current; (ii) rule-based systems suffer from an inability to detect attack scenarios that may occur over an extended period of time; and (iii) they lack flexibility in a rule-to-audit record representation. Popular alternatives for ID are computational intelligence (CI) approaches that include artificial neural networks (ANNs), evolutionary algorithms (EAs), swarm intelligence (SI) and fuzzy systems (FS).

An ANN can be viewed as having circuits of highly interconnected units with modifiable interconnection weights which transform a set of inputs to a set of desired outputs, with the result determined by the characteristics of the elements and weights associated with the interconnections among them.[5,6] Many studies have employed NNs for IDS and they possess the advantages of being able to: (i) be flexible; (ii) analyze data from a network even if it is incomplete or distorted; (iii) conduct data analysis in a nonlinear fashion; (iv) undertake intrusion responses before permanent damage occurs due to their processing speeds; and (v) "learn" the characteristics of misuse attacks and identify instances that are unlike any previously observed by the network. However, their cons are that[3,5]: (i) their training routines require a very large amount of data to ensure the results are statistically accurate; and, thus, (ii) they are time consuming; and (iii) over-training is a problem for some classes of machine learning.

EAs are population-based search methods that employ some form of selection to bias the search toward good solutions,[7] with several used to solve different optimization problems. They are simple in concept, do not require specific mathematical properties to be satisfied, are robust to dynamic changes, can handle evaluating solutions in parallel, have the capability for self-organization and have broad practical applications.[8,9] However, they still have some drawbacks, such as sometimes being unable to converge to the global optimal.[10]

A swarm is a population of interacting elements that is able to optimize some global objective through a collaborative search of a space.[11] SI is a field that focuses on the collective behaviors that result from local interactions of individuals with each other and their environment, such as flocks of birds and herds of land animals. Of the SI algorithms, ant colony optimization (ACO) and particle swarm optimization (PSO) are the most popular. Using SI for ID has also been paid a great deal of attention during the last few decades.

As, in practice, it is common to deal with vague and imprecise information and many types of intrusions cannot be crisply defined, FSs have been used for ID. A fuzzy set is a class of objects characterized by a membership function in which each object is assigned to a grade of membership, ranging between zero and one.[12] Also, as

the degree of alarm that can happen with intrusions is often fuzzy,[13] in fuzzy misuse detection, fuzzy models, such as fuzzy rules or classifiers, are used,[13–15] and in fuzzy anomaly detection systems, fuzzy normal behavior profiles are built.[16]

There are a number of manuscripts which elaborate on the use of machine learning techniques for ID. However, some were published quite some time ago[17–21] while others focus on only SI.[22] From an evolutionary computational point of view, some did not place much emphasis on EA operators (representation, crossover and mutation) and their effects on the performances of EAs.[23] Therefore, the aim of this review is three-fold: (i) to present an overview of EAs which can help new researchers to understand this field; (ii) to survey the research contributions of investigations into the use of EAs and SI for building IDSs; and (iii) to demonstrate the challenges the current research faces and discuss new research directions.

The rest of this paper is organized as follows: Section 2 presents an overview of IDS. EAs and SI are discussed in Sec. 3. The benchmark datasets used to evaluate IDSs are presented in Sec. 4. Section 5 discusses the EAs proposed in the literature for ID, with Sec. 6 presenting those based on ACO and PSO. Finally, a discussion and conclusions are provided in Secs. 7 and 8, respectively.

2. Overview of IDSs

An intrusion can be defined as any set of actions that attempts to compromise the integrity, confidentiality or availability of a resource.[24] The field of ID dates back to the 1980s, when Anderson[25] investigated the importance of audit trails for detecting misuse. The many IDSs proposed since then can be divided into four groups: (i) misuse versus anomaly; (ii) network-based versus host-based; (iii) real-time versus offline; and (iv) passive versus active response IDs.

A misuse-based IDS compares monitored data with attacking patterns/signatures which have predefined descriptions of intrusive behavior (i.e., learned from the training data). If the monitored data matches the pattern of some known attack data, it is considered intrusive.[26] As these systems usually provide low alarm rates which network administrators can use to take preventive actions, their ability to detect new types of attacks is low.[3] In contrast, an anomaly-based IDS compares the features of a system with normal patterns. If the data deviates from normal behavior, this model classifies it as an attack. One advantage of this type of IDS is that it is able to detect unknown or new attacks, but it has a high false alarm rate (as normal events are predicted to be attacks).

A network-based IDS (NIDS) considers network traffic the main source of audit data[1] but end-to-end encryption can avoid it.[3] A host-based IDS monitors the files and processing activities related to a software environment associated with a specific host through four different methods[27]: file system monitoring (checking the integrity of files and directories), log file analysis (analyzing log files for patterns indicating suspicious activity), connection analysis (monitoring connection attempts to and from a host) and kernel-based ID (detecting malicious activity on a kernel level).

Real-time systems aim to detect intrusions while network traffic is in progress. Although it is of benefit to raise an alarm once an intrusion is found, such detections are challenging, especially for anomaly-based systems that often use computationally expensive techniques.[1] In contrast, offline systems analyze the traces and logs previously collected which aid in the forensic testing domain, with these types of detections beneficial when understanding intruders' behaviors is required.

In a passive alerting-based IDS, alarms are generated when attacks are detected without any action taken. In contrast, in an active response-based IDS, an action is taken for every connection that carries an attack.[3]

3. Evolutionary Computation Algorithms and Swarm Intelligence

CI is "*the study of adaptive mechanisms to enable intelligent behavior in a complex and changing environment*"[28] and includes ANNs, EAs, SI, AIS and FSs. As CI with another process, such as logic, deductive reasoning, expert systems, case-based reasoning or symbolic machine learning systems, forms part of the field of artificial intelligence (AI), it can be seen as a sub-branch of AI.[28] As previously stated, in this paper, we concentrate on only EA and SI techniques.

Due to the number of pages limitation, all details about EAs and SI approaches are found in the supplementary material shown in Appendix A.

4. Datasets

Two well-known datasets have been introduced to evaluate the performance of an IDS: (i) DARPA[29] and (ii) KDD99 Cup.[30] Also, some researchers have tried to generate their own datasets to analyze their algorithms, such as Refs. 31–33. Due to the number of pages limitation, the detailed descriptions of well-known datasets are given in the supplementary material shown in Appendix A.

5. Evolutionary Algorithms for Intrusion Detection

In the following subsections, the use of EAs for ID is provided. Generally speaking, IDS-based EAs fall into four categories: (i) evolving rules, (ii) evolving fuzzy rules, (iii) finding the optimal set of clusters, and (iv) hybrid approaches which combine more than one CI technique.

5.1. *GAs*

5.1.1. *GA for evolving rules (signatures)*

Ludovic[34] proposed a GA for simplified security audit trails analysis (GASSATA). In it, a chromosome was represented as a binary vector in which each element indicated the presence of a particular attack. The fitness function was based on the risk associated with the attacks involved, while a quadratic penalty function for

mismatched details was employed. Based on the results obtained, the algorithm performed well but it did not identify the reason for an attack. In addition, it was not able to ascertain the multiple realizations of a particular attack. This work was then extended[35] to maintain a balance between detecting all possible attacks and reducing the number of false alarms, by introducing the modified fitness function. Based on the results obtained, there were no false positives and the number of false negatives was reduced. The fitness function was then modified, as in Ref. 36, to address the problem of false positives. The system was used for misuse detection by using binary encoding to represent each gene and employing tournament selection. The crossover operator was used to exchange sub-parts of two chromosomes, while in the mutation operator, the allele values were randomly changed based on a predefined probability.

Neri[37,38] used the REGAL[39] methodology, which is based on a distributed GA, as a learning system to detect intrusions. REGAL combines the theory of niches and species of biological evolution together with parallel processing. Different crossover operator types, such as two-point, uniform, generalizing and specializing,[40] were exploited. Unfortunately, no details about the fitness function used to measure the quality of each individual were provided. The system was tested on the DAPRA dataset and its performances were high, above average, average and low for detecting Probe, DOS, U2R and U2L attacks, respectively.

Balajinath and Raghavan[31] proposed a GA-based ID (GBID) for anomaly detection. In it, a GA was used to learn user behaviors in a computing system, where various user commands, i.e., 4353, were mapped into codes to represent a chromosome in the population while the fitness function of each individual was calculated as: $f = 1 - |\sigma - \varphi|$, where σ is the user entropy of a predicted behavior-gene and φ the average user entropy of m-corrected detected behavior-genes. A user behavior was characterized by three components: (i) a match index, that is, the ratio of the number of correctly predicted commands to the length of the command sample; (ii) an entropy index which was a measure of the user behavior dynamics in the command sample; and (iii) a newness index, that is, the ratio of the number of commands which did not happen in the history to the cardinality of the command sample. 25 users whose command history size was greater than 200 commands were considered for the experiments. Based on the results obtained, GBID was able to obtain a detection accuracy rate of 96.8% and a false alarm rate of 3.2%.

Chittur[41] introduced an approach for malicious ID using a GA to generate an empirical model of malicious computer behavior based on a training date. In it, the fitness of an individual was dependent on how many attacks were correctly detected and how many normal-use connections were classified. The algorithm was tested on the KDD99 Cup dataset and was able to detect intrusion with an accuracy level of 97.8% and a false rate of 0.6877%.

In Ref. 42, a classifier-based decision support component for an IDS to monitor the activities of Unix machines from packet to user-level, which determined the correlation among the observed parameters during intrusive activities, was proposed. In it, a classifier system received a message/input from the environment which it

placed in a message board and then searched the classifier list to find a rule that matched it. However, if no rule matched it, a GA was used to evolve new classifiers. The fitness function of each rule was based on how well the rule reflected the domain knowledge, its level of generality and the diversity in the best rule set, with a sequential niching technique used to maintain diversity in the generating rules. The performance of the proposed algorithm was above average.

Pillai *et al.*[43] proposed a network IDS (NIDS) based on a GA in which a dataset that included the necessary information for generating rules to train the GA was initiated. Once the GA was trained, more network connections were added to the dataset. The rules were in an "*if-then*" format. In their algorithm, after initializing a population of individuals, the GA was used to match the rules with any anomalous connections. Each rule had to carry values for the intrusions it detected and a value for a false alarm it produced. The fitness function used in the algorithm was: $f = \frac{a \times B}{A - b}$, where "$a$" is the value the specific rule carries for the number of correctly detected intrusions, "b" the value the specific rule carries for the number of false alarms, "A" the correctly detected intrusions from all the rules and "B" the total number of normal connections in the dataset. It must be noted that no computational results were presented in the paper to justify the algorithm's performance.

In Ref. 44, a GA was used to automatically construct the rules of ID. Individuals were coded using an extended binary alphabet scheme, such that each gene had a value of 0 or 1 or #. The roulette wheel selection mechanism, a two-point crossover and a bit-flip mutation were employed. The fitness value of each individual was based on obtaining high detection and a low false alarm rates, such that $f = (\frac{\alpha}{A} \times 100\% - \frac{\beta}{B} \times 100\%)$, as described earlier. The algorithm was tested on the DARPA dataset and the results showed that the proposed algorithm was able to obtain high fitness values of 0.9997, and a low false alarm rate of 0.0305%.

Gong *et al.*[45] introduced a GA-based approach, in which the GA was used to derive a set of classification rules from audit data that were then used to detect intrusions. Seven features were selected from the network audit data to compose a rule, where six features were connected using the logical AND operation to compose the "condition" part of a rule, while the other was used to define the attack, with a rule example.

The quality of each rule was based on the support-confidence framework, in which if a rule is represented as ***if*** A ***then*** B, its fitness was determined as: $f = w_1 \times support + w_2 \times confidence$, where $support = \frac{|A \, and \, B|}{N}$, $confidence = \frac{|A \, and \, B|}{|A|}$, N is the total number of network connections in the audit data, $|A|$ the number of network connections matching condition A and $|A \, and \, B|$ the number of network connections matching the rule if A then B. The weights (w_1 and w_2) were used to control the balance between the two terms and had the default values of $w_1 = 0.2$ and $w_2 = 0.8$. The algorithm was tested on the DARPA dataset for detecting only two types of attacks. Although, based on the results obtained, the algorithm performed well on the training data, its performance on the testing data was not sufficiently good.

In fact, the work presented in Ref. 45 was an extension of that presented in Ref. 46 with a few modifications, in which, in Ref. 46, the outcome of a rule was not represented in the chromosome. In addition, the fitness function used was based on whether a field of the connection matched the pre-classified dataset, and if so, multiplied the weight of that field. It is worth mentioning that, in Ref. 46, a niching mechanism, the hamming distance, was used to find multiple rules as a small set of rules would not be sufficient to cover all of the possible network connections. The algorithm was trained and tested on the DAPRA dataset but no details of its performance were discussed, and furthermore, the effects of the GA parameters were not analyzed.

In Ref. 47, a GA-based approach for detecting network misuse attacks, especially those of Smurf and WarezMaster (bugs associated with a FTP server), was introduced. The algorithm's performance was low, with a DR of 59%, while its false alarm was 0.1%. However, the differences between this algorithm[47] and that in Ref. 45 were not clear, and the crossover and mutation types and their effects were not discussed.

Xia *et al.*[48] introduced a hybrid method based on information theory and a GA to detect network attacks in which the former was used to extract the most important features that could be used to efficiently detect a network attack. In that study, the 41 features were reduced to four and a linear structure rule was employed to classify the network's behavior into normal and abnormal. The fitness function of a chromosome was the ratio between the correct detections and sample size. The algorithm was tested on the KDD99 Cup benchmark dataset and obtained DRs of 98.34%, 99.33%, 63.34%, 5.86% and 93.95% for Normal, DOS, U2R, R2L and Probe attacks, respectively, with false positive and negative rates of 1.658% and 0.7529%, respectively.

In Banković *et al.*,[49] the principal component analysis (PCA) technique was employed to define the features that participate most frequently in the machinery of an attack. The fitness function aimed to minimize the number of features while maintaining a high rate of detecting intrusions. Two fitness functions were tested on the proposed algorithm. The algorithm was tested on the KDD99 Cup dataset, with PCA able to extract three of the 41 features. Considering the results obtained, using the first fitness function provided about a 96% average detection rate, while using the second led to a 87.5% maximum detection rate.

In Abdullah *et al.*,[50] the fitness function was $\frac{\alpha}{A} - \frac{\beta}{B}$. A feature reduction mechanism was also used. The algorithm was trained and tested on the KDD99 Cup dataset, obtaining a detection rate of 99.87% and false positive rate of 0.003%.

Hoque *et al.*[51] proposed a GA framework of two steps: a pre-calculation and detection scheme in which a population of individuals was created for test data and proceeding through the evolution process and, consequently, the type of test data was predicted. To measure the fitness of a chromosome, the standard deviation equation with distance was employed. Based on the results obtained, the algorithm showed good performance in detecting DOS intrusions, with an accuracy rate of 99.4%, but was poor in terms of detecting all other types of intrusions.

Gómez *et al.*[52,53] introduced a Pareto-based multi-objective EA within the detection engine of Snort[54] (MOEA-Snort). The rules/signatures were optimized based on two objectives: (i) minimizing the number of false positives (FP) (nonantagonistic traffic considered an attack); and (ii) false negatives (FN) (antagonistic traffic not detected), and a population of solutions was randomly initialized. At each generation, the individuals were evaluated according to two optimization modes (single aggregate objective function ($f = \alpha \cdot FN + \beta \cdot FP$, where $\alpha + \beta = 1, \alpha = [0, 1]$ and $\beta = [0, 1]$) or Pareto-optimization, with the best solutions continually stored. Once a stopping criterion was met, the best solution was obtained by the single aggregate mode and all nondominated solutions returned. The algorithm was tested on the DAPRA data, which proved that using multi-objective optimization could improve results.

5.1.2. *GA for evolving fuzzy rules*

Gomez and Dasgupta[55] used a GA to generate fuzzy rules for detecting anomalies and some specific intrusions. In it, each chromosome was encoded as an expression tree. The fuzzy confusion matrix was used to calculate the fitness of each chromosome. The computational results showed that the algorithm was able to obtain a detection accuracy rate of 98.95% and a false alarm rate of 7.0%.

In Ref. 56, a framework based on fuzzy genetic learning for ID, in which a GA was used to evolve fuzzy *if-then* rules, was introduced. In this algorithm, after an initial population of random rules was initialized, each fuzzy rule (R_j) was assigned to a class and a membership grade (μ_{R_j}). The uniform crossover and random mutation methods were considered for the GA. The algorithm was tested on the KDD99 Cup dataset and was able to obtain a 99.08% accuracy rate, with a false alarm rate of 3.85%.

As an extension of their previous work presented in Ref. 56, Abadeh *et al.*[57] introduced a parallel genetic local search algorithm (PAGELS) to generate fuzzy rules for misuse detection. The system used the Michigan's approach, in which each individual represented the fuzzy rule "*if conditions then a prediction*", and the population was divided into subpopulations, with each assigned to a distinct processor. The number of subpopulations was equal to the number of classes in the classification problem. For each subpopulation, all individuals represented rules of the same class in the "prediction" part, while the local search was conducted only for individuals with a fitness higher than a threshold. The fitness of each chromosome was similar to that presented in Ref. 56. Based on the results obtained, PAGELS was able to detect Normal, U2R, R2L and DOS attacks with accuracy rates of 97.6%, 84.74%, 92.4% and 98.76%, respectively, and a false alarm rate of 0.29%.

The authors then extended their work by providing a comparison among three genetic fuzzy systems based on the following three techniques[58]: (i) Michigan: A single fuzzy if-then rule coded as an individual. (ii) Pittsburgh: A set of fuzzy if-then rules was coded as an individual. (iii) Iterative rule learning (IRL): At each iteration,

the fuzzy rule that best classified the current distribution was added to a fuzzy rule base, and four objectives were used to measure the quality of each rule.[59] All three fuzzy systems were tested on the DARPA dataset. The detection rate of the Pittsburgh-based system was the best (99.53%), while those of the Michigan- and IRL-based systems were 88.13% and 93.2%, respectively. On the other hand, the false alarm rate of the Pittsburgh-based system was high (1.94%), while those of the Michigan- and IRL-based systems were 0.11% and 0.18%, respectively.

Tsang *et al.*[60,61] proposed a fuzzy rule-based system, based on an agent-based technique for anomaly ID, which could also act as a genetic feature selection wrapper to search for an optimal feature subset for dimensionality reduction. In it, every fuzzy-set agent used a fuzzy-sets distribution strategy based on a hierarchical GA (HGA) to initialize the fuzzy-sets information from which an interpretable fuzzy rule base was generated. A nondominated sorting GA (NSGA-II) was then used to evolve the fuzzy rules, each of which was encoded as a string of length D, with the following three criteria used to measure the fitness of each chromosome: (i) accuracy in terms of the classification rate; (ii) number of fuzzy rules; and (iii) total length of fuzzy rules. The experimental results obtained from the KDD99 Cup benchmark data showed that this proposed approach was able to obtain a high detection accuracy rate for intrusion attacks and a low false alarm rate for normal network traffic with a minimized number of features.

Motivated by the work presented in Ref. 59, in Ref. 62, an IDS based on classification and association rules mining for predicting different behaviors in networked computers was proposed. For each class label, a GA with uniform crossover and biased mutation operators was run and, after the evolutionary process, the best individual (rule) was inserted in the fuzzy rule base. The algorithm was tested on the KDD99 Cup dataset and obtained accuracy detection rates of 95.8%, 54.10%, 97.40%, 10.9% and 6.9% for Normal, Probe, DOS, U2R and R2L classes, respectively.

Dasgupta and González[63] proposed an algorithm, inspired by the negative selection mechanism of the immune system that can detect foreign patterns for anomaly detection, with a GA used to evolve "good" rules to cover the nonselfspace and a niching technique employed to maintain diversity. The fitness value of a rule (R) was based on two factors: (i) the number of elements in the training set (S) belonging to the subspace represented by the rule, and (ii) the volume of the subspace represented by the rule, $\underline{x}_j$ and $\overline{x}_j$ are the lower and upper values, respectively. The algorithm was tested on the DARPA dataset and was able to obtain an accuracy rate of 87.5% and maximum false alarm rate of 1%. This work was then extended in Ref. 64, in which a deterministic crowding technique was used. Also, a minor modification was made to the fitness function, such that $f = \Theta \cdot \mathrm{vol}(R) + (1 - \Theta) \cdot n_e(R)$. The algorithm was tested on the two well-known datasets: (i) KDD99 Cup, for which the accuracy rate was 98.30% and FPR 2.0%; and (ii) DARPA, for which the DR was 98.33%. The abovementioned work was then extended in Ref. 65 by using a different structured representation method to codify the rules which sped up the GA's

convergence. The results obtained from testing the algorithm on the DARPA and KDD99 Cup datasets showed respective DRs of 94.63% and 98.22%.

Fries[66] introduced a fuzzy-genetic mechanism to ID with its algorithm consisting of two phases: (i) First, a GA was used to establish an optimal subset of particular communication features in which each chromosome was represented as a set of features, each of which was 0 or 1. The GA was then evolved to find the optimal set of features to be used to train the rule set. The fitness function of each chromosome was the weighted sum of the accuracy of the hypothesized subset (the ratio of correct identifications to total number of communications in the test set) and the number of features used in the subset. Based on the results obtained, the 41 features were reduced to only 8; and then (ii) a GA was used to optimize a set of fuzzy rules. The algorithm was tested on the KDD99 Cup dataset, and was able to obtain a 99.6% accuracy rate and false positive alarm rate of 0.2%.

5.1.3. *GA with fuzzy clustering*

Although machine learning techniques have the positive capability to automatically retrain detection models on different input data, labeled data are not readily available in most cases. Motivated by this fact, Liu *et al.*[67] introduced an ID based on a genetic clustering (IDBGC) algorithm which involved two stages: (i) establishing a set of original clusters using the nearest neighbor method to group very similar instances into a cluster and by altering noisy data objects based on some similarity or dissimilarity metrics; and (ii) combining the original clusters using a GA. The fitness function was based on the degrees of nearness and separation among the clusters and the single-point crossover was employed, while in the mutation operator, each bit changed from 0 to 1 based on a predefined probability. The algorithm was tested on the KDD99 Cup dataset, and based on the results obtained, was able to detect DOS, U2R, R2L and Probe attacks with accuracy rates of 56%, 78%, 66% and 44%, respectively, and a false alarm rate of 0.4%.

In Ref. 68, the proposed algorithm consisted of two stages: the first built up clustering sets using a similarity rule; and the second used a GA to optimize these sets to distinguish between normal and intrusive actions. The objective function was based on how many attacks were correctly detected and how many normal-use connections were classified as attacks. The algorithm was tested on the KDD99 Cup dataset and achieved an overall accuracy level of 95% and low false alarm rate of 0.32%.

Leon *et al.*[69] used the unsupervised niche clustering (UNC) with fuzzy sets theory for anomaly detection. This was a genetic niching technique for clustering, which was able to automatically detect the number of clusters and generate a fuzzy membership function for each cluster, which defined the normalcy level of a data sample. In this UNC, each individual represented a candidate cluster, as determined by its center, along with a robust measure of its scale (σ^2). The fitness value of the ith chromosome was defined as the density of a hypothetical cluster at that location: $f = \frac{\sum_{j=1}^{N} w_{ij}}{\sigma_i^2}$,

where $w_{ij} = -\frac{d_{ij}^2}{2\sigma_i^2}$ is a weight that measures a typical data point (x_j) in the ith cluster, d_{ij}^2 the distance from x_j to the cluster center (c_i) and N the number of data points. The algorithm was tested on the KDD99 Cup dataset and was able to achieve an overall accuracy level of 99.20% and false alarm rate of 2.2%.

5.1.4. *Hybrid GA*

Han and Cho[70] proposed an evolutionary NN (ENN) framework for anomaly detection, in which an EA was used to determine the internal weights, topologies and numbers of hidden nodes of NNs, with a matrix-based encoding scheme, rank-based selection and single-point crossover considered. In the mutation operator, two nodes of a NN were randomly selected, and if there was no connection between them, a connection was established using random weights; otherwise, the connection link and weights were removed. The proposed methodology was able to obtain better results than multi-layer NN and Elman network, as well as reducing the computational time required to build normal-behavior models.

An integration of neuro-fuzzy networks, the fuzzy inference approach and a GA for detecting and classifying intrusions in a computer network, in which some parallel neuro-fuzzy classifiers were deployed to generate an initial classification, was proposed in Ref. 15. The fuzzy inference system was used to generate the final decision as to whether the current activity was legitimate or malicious, while the GA was used to optimize the membership function of the fuzzy decision-making module. In this algorithm, two fitness functions were employed; the first was based on the CPE with equal misclassifications costs; and the second on the cost per example. According to the results obtained, the algorithm performed well.

In Ref. 71, a hybrid EA and NN algorithm was proposed for ID. In it, a GA was used to select an appropriate feature subset, optimize the number of hidden neurons, determine a number of training epochs and choose a basis function type for a radial basis function (RBF) network. In addition, pre-processing algorithms were applied to extract statistical information and only TCP and IP headers were used for feature extraction. The fitness value of each individual was determined using the classification error (percentage of wrongly classified TCP connections) of the corresponding RBF network. Based on the results, the algorithm was able to attain an average accuracy rate of 99.4%.

Kim *et al.*[72] proposed a hybrid methodology, involving a GA and SVM, for anomaly detection in which the aim of using the GA was to improve the performance of the SVM by obtaining the optimal set of features, as well as optimal parameters, for a kernel function. In the GA, a chromosome was decoded into a set of features and parameters for a kernel function, which was then used by the SVM classifier. The evolution process was continued to determine the optimal detection model based on the fitness values obtained from the SVM classifier. The approach was tested on the KDD99 Cup dataset and was able to achieve a 99% detection rate.

Stein *et al.*[73] proposed a hybrid algorithm that combined DT and GA for ID, in which the GA was used to select a subset of input features for decision tree classifiers to randomly generate an initial population. Each individual consisted of 41 features, with each gene binary encoded, and for it, the C4.5 approach was used to build a decision tree. Subsequently, DT was tested on validation datasets to generate the overall classification error rate which represented the fitness value of that individual. After calculating the fitness values of all individuals, the GA started to evolve the population in order to generate a new one using a two-point crossover and bit-level mutation. The algorithm was tested on the KDD99 Cup dataset and the results compared with those using only DTs. Although it demonstrated minor improvements in detecting DOS, Probe and R2L attacks, DTs were better at detecting a U2R attack.

Kuang *et al.*[74] introduced a multi-layer SVM classifier for ID. In it, kernel principal component analysis (KPCA) was used to reduce the number of features, while a GA was deployed to optimize the punishment factor, kernel parameters and the tube size of SVM. The algorithm used binary encoding, and a mean absolute percentage error was used as a fitness function. On the KDD99 Cup dataset, the algorithm performed better than other algorithms, as its DR was between 94.226% and 96.377%, while the FR was between 0.975% and 1.025%.

5.2. *GP*

5.2.1. *GP for evolving rules*

In Lu and Traore,[75] the support-confidence framework was used to calculate the fitness function. Symbolic expressions were used to generate the initial population and a single-point crossover was employed. In addition, a population diversity mechanism, inspired by token competition, was used. The algorithm was tested on the DARPA benchmark set, and the results showed that it obtained a 5.04% FPR, 5.23% FNR and 57.14% rate for detecting unknown attacks. Moreover, on the testing dataset, it obtained a detection rate close to 100% with a low false negative rate, i.e., between 1.4% and 1.8%. However, when the false positive rate was close to 0%, the detection rate was about 40%.

Song *et al.*[76,77] introduced an anomaly IDS based on a linear GP. The algorithm used the dynamic subset selection (DSS) technique to provide a hierarchy of subset selections, compatible with the organization of memory hierarchies widely employed in computer architectures. An individual was described in terms of a (uniform) randomly selected number of pages. Three variation operators, with each applied based on a corresponding probability, and the three fitness functions of (i) equal class, (ii) variable weights and (iii) hierarchical were employed, along with a single-point crossover. Two types of mutation operator were considered: (i) ex-OR which exchanged an instruction with a new one but did not show any benefits; and (ii) swap whereby two instructions with a uniform probability were first identified in the same

individual and then interchanged. The algorithm was tested on the KDD99 Cup dataset and it was noted from the results that it obtained a $\approx 90\%$ detection rate.

Abraham *et al.*[78] compared the performances of three GP techniques (LGP, multi-expression programming (MEP), and gene expression programming (GEP)) for detecting known types of attacks. Based on the results obtained, the MEP and GEP accuracy rates were more than 95%, with the former achieving 99.75% for three attack classes. However, no details were provided regarding the fitness function used.

Folino *et al.*[79] presented a GP ensemble for a distributed IDS (GEdIDS). The algorithm ran on a distributed hybrid multi-island model-based environment in which each island contained a cellular GP, the aim of which was to generate a decision-tree predictor. Also, every GP cooperated with its neighboring nodes, and the fitness function used was the number of training examples correctly classified. The GEdIDS was tested on the KDD99 Cup dataset and was able to reach a 90.5812% detection rate, with a false positive rate of 0.5648%. Later, Folino *et al.*[80] proposed another GEdIDS in which data was distributed across several sites and a combination of different classifiers cooperated to provide the required information. Each node in the network was represented as an island, where a learner was employed and which, periodically, imported the remote classifiers from the other islands and combined them with its own classifier. The fitness function used was the number of training examples classified in the correct class. Experiments performed on the KDD99 Cup dataset showed that the GEdIDS was able to attain accuracy rates of 99.44%, 81.51%, 97.03%, 9.47% and 7.36% for detecting Normal, Probe, DOS, U2R and R2L attacks, respectively, with a false alarm rate of 0.5225%.

In Ref. 81, the suitability of an LGP algorithm for modeling an IDS was investigated and tested on the DARPA dataset, and was compared with DT and SVM approaches. Based on the results obtained, it was superior to both the other algorithms in terms of detection accuracies, except for the U2R class for which the DT method was considered the best. Unfortunately, no details of the evaluations of individuals were discussed.

In Ref. 82, a page-based LGP for detecting Data Link layer attacks on a WiFi network was proposed. Motivated by the work presented in Ref. 83, the homologous crossover, *instruction-wide mutation* and *field-specific* mutation operators were used. Two fitness functions were employed: the first was called a *switching fitness function* which penalized the GP based on whether there was a false positive or false negative result; and the second was $f = 0.5 \cdot ((1 - DR) + FPR)$. The results showed that the algorithm was able to achieve a detection rate greater than 90% with a FPR less than 1%.

Hansen *et al.*[84] investigated the potential of GP as an adaptive method for responding to the complexities of cyberterrorist ID, by increasing the detection rate, while at the same time, reducing the false positive detection rate. A machine-code linear structure was used to encode the individuals and the homologous crossover employed. The algorithm was tested on the DAPRA and the KDD99 Cup datasets and, based on the results obtained, the detection rates for Normal, Probe, DOS, U2R and R2L attacks were 98.9%, 99.9%, 99.8%, 99.9% and 100%, respectively.

5.2.2. *GP for evolving fuzzy rules*

Mabu *et al.*[4] proposed a fuzzy class-association rule mining method, based on GNP, for detecting misuses and anomalies using a single-point crossover. Regarding the mutation operator, for an original individual, a new mutated offspring was generated as follows. Each node branch was selected to be reconnected to/or changed with another one based on a predefined probability, with the fitness function of each individual calculated as: $f = \sum_{R \in RS} w_1 \cdot Fit + w_2 \cdot \alpha_{new}(R)$, where $Fit = \frac{NT_c}{NT} - \frac{Nn_i}{Nn}$ is the fitness of the extracted rule (R), NT_c the number of connections correctly detected by rule $R, \psi NT$ the number of connections in the training data, Nn_i the number of normal connections incorrectly detected by rule $R, \psi Nn$ the number of normal connections in the training data, RS the set of suffixes of the association rules extracted by the individual, where $\alpha_{new}(R) = \alpha_{new}$ if R is new or 1 otherwise, and w_1 and w_2 the control parameters. The experimental results obtained using the KDD99 Cup and DARPA datasets showed that the proposed method provided competitively high detection rates compared with other machine-learning techniques and GNP with crisp data mining, with the algorithm able to obtain a detection rate of 98.7% for misuse, and the FPR and FNR were 0.53% and 3.75%, respectively. For anomaly detection, DR, FPR and FNR were 94.4%, 7.2% and 5.0%, respectively.

5.2.3. *Hybrid GP*

Crosbie *et al.*[85] proposed an IDS in which multiple agents ran in parallel, with each trying to detect anomalous intrusions, but independently of each other. Furthermore, GP was used to evolve the agents and the fitness function was $f = (100 - \delta) - penalty$, where the absolute difference (δ) between the agent's reported suspicion and scenario's outcome was computed as: $\delta = |outcome - suspicion|$, while the penalty value was computed based on how the scenario was ranked, i.e., $penalty = \delta \times \frac{ranking}{100}$.

Based on the results obtained, with three agents used to detect three activities (connections to privileged ports, logins and then long pauses, then logins, and logins and ftp with long pauses), the results were not sufficiently good. For the three activities, the first agent's detection accuracies were 83%, 31% and 73%, respectively, the second's 100%, 26% and 47%, respectively, and the third's 98%, 0.0% and 26%, respectively.

A flexible neural tree (FNT) for selecting input variables and detecting network intrusions was introduced in Ref. 86. The hierarchical structure was evolved using GP instead of the probabilistic incremental program EA used in their earlier work,[87] while the parameters (weights and flexible activation function parameters) were optimized by a memetic algorithm (i.e., a local search with a GA was used) and a single-point crossover used. Five different mutation operators were employed to generate offspring from the parents: (i) changing one terminal node; (ii) changing all terminal nodes; (iii) growing; (iv) pruning; (v) pruning redundant terminals. The algorithm was tested on the DARPA dataset and showed good performance in comparison with NN and decision tree techniques.

Faraoun and Boukelif[88] proposed a dynamic GP-based classifier for ID which genetically co-evolved a population of nonlinear transformations on the input data to be classified and mapped them to a new space with a reduced dimension. The fitness value of each individual was based on it being inversely proportional to the computed number of common points between transformed sets ($T(D^i_{Train})$). A tree representation, two-point crossover operator and roulette wheel were employed. For the mutation used, a mutation point, which could be either a leaf node or sub-tree, was randomly selected and then randomly replaced. The algorithm was trained and tested on the KDD99 Cup dataset and could detect Normal, Probe, DOS, U2R and R2L with rates of 92.54%, 78.77%, 99.97%, 26.7% and 99.74%, respectively.

5.3. *EP*

In fact, very little work using EP for IDS has been presented in the literature.

Anchor *et al.*[57] introduced a multi-objective EP for detecting attacks against computer networks. In their algorithm, existing knowledge about an attack was injected into a computer defense immune system (CDIS) to develop antibodies which detected that attack plus generalized versions of it. Knowledge of the attack, especially the relationships between packets in it, was used to build up and generalize an attack signature, with this generalized pattern then passed to EP to generate antibodies. The fitness value of an individual was dependent on two issues: (i) a correctly detected attack; and (ii) a false detection. Two types of multi-objective approaches, lexicographic and Pareto-based, were implemented, and the algorithm was only tested to examine its efficiency, rather than the solution's effectiveness. It was found that the methodology based on the lexicographic approach was faster than the Pareto-based approach.

5.4. *EDA*

Chen *et al.*[89] evaluated the performance of an EDA for training a three-layer feed-forward NN classifier to detect intrusions in a network in which the weights, bias and flexible activation function parameters were optimized by the EDA. In it, a floating-point coding scheme was adopted to represent individuals in the initial population. The algorithm was tested on the DARPA dataset and was able to detect Normal, Probe, DOS, U2R and R2L attacks with accuracy rates of 97.58%, 95.57%, 97.76%, 99.90% and 98.90%, respectively. In addition, it was found to be better than two other algorithms, namely, using PSO to evolve a NN (PSO-NN) and decision trees, for detecting all attacks, except U2R, for which PSO-NN was superior to both the other algorithms.

5.5. *DE*

In Ref. 90, DE was used to train a NN for IDS. To add to this, the PCA technique was employed to reduce the number of features. The algorithm was tested on the

KDD99 cup dataset and showed better performance against other NN approaches. However, no details about the fitness functions were discussed.

Elsayed *et al.*[91] evaluated the performance of DE on anomaly detection. The authors proposed a new fitness function, which was based on the detection rate and partial matching. The algorithm showed superior performance to other state-of-the-art algorithms, in which it was able to reach a DR of 100%, while the FPR was 0.489%.

6. Swarm Intelligence for Intrusion Detection

Here, a review of the use of PSO and ACO for ID is provided.

6.1. *PSO*

6.1.1. *Rules-based PSO*

In Ref. 92, PSO was used to extract classification rules. Each particle represented an "*if-then*" rule which was then updated using PSO processes until a stopping criterion was met. The paper introduced a representation scheme called "indexical coding", and when tested on the KDD99 Cup dataset, the algorithm showed feasibility.

Motivated by the fact that most IDSs had high numbers of false alarms, Chang and Wei-ping[93] introduced a PSO-based algorithm to extract a better rules set with a lower false alarm rate for detecting attacks. In the encoding step, symbolic attributes, such as "protocol type", were mapped to integer values. The fitness function considered was $f = \frac{a}{b}$, where a represented the number of correctly detected attacks and b the number covered by this individual in normal connections. The proposed algorithm was trained and tested on the KDD99 Cup dataset, and the best correct classification rate obtained was 92%, while the false alarm rate was 2.84%, which was still a little high.

6.1.2. *Fuzzy Rules-based PSO*

In Ref. 94, PSO was used as a local optimizer to improve the performance of a fuzzy genetics-based machine-learning method. In it, an initial population of fuzzy "if–then" rules was generated, all individuals were then evolved using GA operations, and subsequently PSO used to improve the performance of each individual. The fitness value of the fuzzy *if–then* rule was calculated as the number of training patterns correctly classified by a rule (R_i). The algorithm was tested on the DARPA benchmark dataset, and its detection rates for Normal, U2R, R2L, DOS and Probe attacks were 90.8%, 49.1%, 79.4%, 97.5% and 81.2%, respectively.

In Ref. 95, a hybrid approach involving the K-means, Fuzzy K-means and Swarm K-means approaches was introduced. The algorithm consisted of two phases: (i) training in which particles were evolved until a predefined stopping condition, with each particle assigned a fuzzy membership function indicating the degree of dependency of the ith particle on the Cth cluster, and the fitness function calculated by a

fuzzy inference engine; and (ii) testing in which the Euclidian distance between cluster centroids of the best-evolved particle and input data was calculated and an anomaly notified when that distance was beyond a threshold. The algorithm was tested on the KDD99 Cup dataset and its DR was 95.876% and FPR rate was 2.125%.

6.1.3. *Hybrid PSO*

Hybridizing PSO with other machine-learning techniques, such as NN and SVM, has been presented in the literature.

In Ref. 96, a hybrid approach involving an algorithm-based wavelet NN (WNN), quantum-behaved particle swarm optimization (QPSO) and conjugate gradient algorithm (CG) was introduced for anomaly detection in which QPSO and the CG were used to determine the structure and parameters of the WNN, respectively. The fitness function of each particle was calculated as the error value after training the corresponding network by inputing the training samples, with the gradient descent algorithm then used to optimize some particles. This algorithm was tested on the KDD99 Cup dataset which showed that its detection rate was 96.01% and its false alarm rate was high (4.89%). This work was then extended in Refs. 97 and 98. In Ref. 97, the authors used the hybridization of QPSO and GD to train a RBF NN (RBFNN) for anomaly detection. This algorithm was tested on the KDD99 Cup dataset and obtained a 96.77% accuracy rate and false alarm rate of 8.01%, while in Ref. 98, the authors used QPSO to train a wavelet fuzzy NN (WFNN), with the algorithm obtaining a 95.585% detection rate.

Following the same idea presented in Ref. 89, Michailidis *et al.*[99] proposed a hybrid NN and PSO framework for ID. In it, an initial population of particles corresponding to the weights of a NN was randomly generated. The weights were passed to the NN and each particle's fitness value was calculated, based on the mean squared error (MSE). Then, the local best for each particle and global best particle were recorded, and subsequently, the velocity of each particle was updated, a process which was continued until a predefined stopping criterion was met. The framework was tested on the KDD99 Cup dataset, and based on the results obtained, the algorithm was able to detect Normal, Probe, DOS, U2R and R2L attacks with accuracy rates of 96.88%, 92.2%, 97.74%, 52.86% and 8.3%, respectively, with an average false alarm rate of 0.61%.

In Ref. 100, PSO was used to evolve RBFNNs. The algorithm began by generating a population of particles, each of which consisted of the center of a RBF, and the variance between its function and weight. The fitness function used to measure the quality of each particle was based on its classification accuracy, such that $f = \frac{TC}{TC+FC}$, where TC and FC were correct and false classifications, respectively. Experiments were conducted on the KDD99 Cup dataset, with results showing accuracy rates for detecting Probe, DOS, U2R and R2L attacks of 88.86%, 92.57%, 91.14% and 94.29%, respectively.

Srinoy[101] used PSO to implement a feature selection, with SVMs used as an evaluator of the PSO fitness function (the classification accuracy for the dataset). The algorithm was tested on the KDD99 Cup dataset and was able to detect Normal, Probe, DOS, U2R and R2L with accuracy rates of 99.84%, 99.95%, 99.32%, 68.57% and 90.25%, respectively.

In Ref. 102, a SVM and PSO were combined to detect intrusions, with PSO used to evolve two parameters of the SVM. An affinity function was employed to measure the quality of each particle, such that $f = \frac{1}{e} + \frac{\lambda \cdot l}{\sigma}$, where e is the MSE, λ is a trade-off coefficient between the two items, l is the size of the training dataset and σ is the number of support vectors. Also, a new mutation based on AIS was introduced to help to avoid the algorithm becoming stuck in local solutions. It was tested on the KDD99 Cup dataset and the MSE values obtained were higher than those from a RBFNN, but unfortunately, no details of the accuracy rates were discussed in the paper. Similarly, PSO was used to find two SVM parameters in Ref. 103, and considering the DARPA dataset, the algorithm obtained an average accuracy rate of 97.2612%.

Also, in Ref. 104, PSO was employed to determine a SVM's parameters. To do this, a binary PSO was used to obtain the optimum feature subset for anomaly detection. In this proposal, every particle represented a solution (a selected subset of features and parameter values) and all were then used to build SVM classifier models, with each particle's fitness function based on the detection accuracy. While the results from testing on the DARPA dataset showed that the algorithm's accuracy was 99.8438%, but the false alarm rate was not provided.

Liu *et al.*[105] used rough set theory (RST) to extract features from the dataset, and a PSO to optimize the SVM parameters. k-fold cross-validation of was used to calculate the fitness function of each particle, for which the training dataset was divided into k equal parts, with the average classification accuracy of k times representing the accuracy of the cross-validation (the quality of a particle). The results obtained from testing the algorithm on the DARPA dataset showed that its accuracy rate was $\approx 86.25\%$.

Xiao *et al.*[106] proposed a K-means algorithm based on PSO (PSO-KM), with a Bayesian classifier trained to extract features from the dataset. In it, the position of each particle consisted of k cluster centroids with D dimensions. After the centroids were confirmed, the clustering abided by the nearest rule. The fitness function of each particle was calculated as: $f = \frac{1}{1+\sum_{j=1}^{k}\sum_{x_i \in C_j} d(x_i, C_j)}$, where $d(x_i, C_j)$ is the Euclidean distance between a point (x_i) and a class center (C_j). The algorithm was tested on the KDD99 Cup dataset and the detection accuracy rates for Probe, DOS and U2R ranged from 78% to 94%, but that for R2L was only 22%.

Hu *et al.*[107] proposed a distributed ID framework, in which a local parameterized detection model was constructed in each node using an online Adaboost-based algorithm. A global detection model was then constructed in each node by combining the local parametric models using a small number of samples in the node. This combination was achieved using a hybrid between PSO and SVM. The algorithm

was tested on the KDD99 Cup dataset and obtained 99.99% detection rate on 6 distributed nodes, while the FR was 1.35%.

Similar to the idea presented in their earlier described previous work,[74] Kuang *et al.*[108] introduced a multi-layer SVM classifier for ID, and instead of using GA to optimize SVM parameters, they used chaotic PSO. The algorithm used binary encoding and a mean absolute percentage error was used as a fitness function. The algorithm was tested on the KDD99 Cup dataset, and obtained a DR between 95.036% and 96.698%, while the FR was between 0.964% and 1.015%.

6.2. *ACO*

6.2.1. *Clustering-based ACO*

The ACO algorithm presented in Ref. 109 showed its benefits for clustering small and low-dimensional data. As a further improvement, to deal with highly degraded, or when clustering large and complex data, an ant colony clustering model (ACCM) was introduced in Refs. 110–112 for anomaly ID, in which the meta-heuristic employed global optimization principles in the context of SI. In addition, the PCA technique was used to extract important features from the dataset. The algorithm was tested on the KDD99 Cup dataset and was able to reach a 92.23% detection rate, with a false alarm rate of 1.53%. Regrettably, no information about the fitness function or encoding scheme was provided in the manuscript.

Ramos and Abraham[2] proposed a self-organized ant colony based IDSs (ANTIDSs) to detect intrusions. ANTIDS aimed at finding the optimal clusters of each group of events. To add to this, it employed a mechanism to reduce the number of features which was based on the DT. The algorithm showed competitive performance in comparison with DT, SVM and LGP.

Feng *et al.*[113] introduced an unsupervised anomaly approach based on ACO and Bayesian classification, in which the ACO was used to find the clusters. At each generation, each ant randomly chose an object, and then picked it up or moved it, or dropped it down based on predefined probability, which considered the similarity of the object within the local region. At the end, the best set of clusters was generated. Experiments were conducted on the KDD99 Cup dataset and the algorithm was able reach a 93.03% detection rate, while the false alarm was 8.24%. This work was then extended in Ref. 114, in which ACO and self-organizing map (SOM) was used to find clusters, a feature reduction mechanism was also deployed. The results showed that the algorithm was able to obtain a high detection rate, but the false alarm rate was not mentioned, nor the fitness function used in the algorithm. As another extension,[115,116] both ACO and dynamic self-organizing maps (DSOM) were used to produce clusters for anomaly detection. DSOM is an unsupervised NN, which was initialized with four nodes and grows nodes to represent the input data. A swarm similarity function was used as a fitness function of each ant. Experimental results conduced on the KDD99 Cup dataset showed that the algorithm was superior to the SVM and NNs techniques, but the false alarm rate was not discussed.

Motivated by the ant-miner algorithm presented in Ref. 117, Soroush *et al.*[118] proposed a boosting ACO algorithm for ID, in which ACO was used to extract a set of classification rules (*if-then*) from a network dataset which was then used to detect normal and abnormal behaviors. The DARPA data was used to evaluate the performance of the algorithm. Based on the results obtained, although the detection rate was high, the false alarm rate was 2.7%.

Also based on Ref. 117, He *et al.*[119] introduced a multiple ACO (MACO) to extract a set of rules for ID. In MACO, several ant colonies were employed (each focused on a child problem) to find solutions for a problem simultaneously. Each colony of ants deposited a different type of pheromone and each ant was only attracted by the pheromone deposited by other ants in the same colony. The quality of each rule was determined by the accuracies of positive and negative instances. Using the KDD99 Cup dataset, MACO's detection accuracy was good.

6.2.2. *Hybrid ACO*

Hai-Hua *et al.*[120] proposed an IDS-based ACO, in which ACO was used to extract features from the data and the SVM was used for training. In it, features were represented as graph nodes, with the edges between them denoting the choice of the next feature. Ants then traversed through the graph to add nodes until the stopping criterion was satisfied, with the Fisher discrimination rate adopted as the heuristic information for the ants' traversal and SVM as a classifier to evaluate the feature subset generated by it. Different experiments were conducted on the KDD99 Cup dataset to detect each attack type separately. The algorithm was able to detect Probe, DOS, and U2R&R2L attacks with accuracy rates of 99.4%, 95.2% and 98.7%, and false alarm rates of 0.35%, 3.24% and 1.6%, respectively.

Srinoy[121] proposed a framework based on a SVM and fuzzy ACO for anomaly detection, with the SVM employed to create raw clusters which were then refined using an artificial fuzzy ants clustering (AFAC) algorithm to find good partitions of the data. Ants and objects were placed randomly on the board, with each ant moved on the board and possibly picking up or dropping an object based on a probability. Also, the SVM was used to reduce the number of unnecessary features. Although the algorithm was trained and tested on the KDD99 Cup dataset, details of the results were not provided and its encoding scheme and fitness function were not discussed.

Abadeh *et al.*[122–124] employed an ACO-based local search algorithm to improve the quality of a final fuzzy classification system. In it, the authors extended their earlier study previously discussed,[57] the Michigan-based ID algorithm, from a problem with four classes, to a five-class classification problem and used an ACO algorithm as a local search along with a GA. The algorithm was tested on the DARPA dataset and was able to detect Normal, U2R, R2L, DOS and Probe attacks with accuracy rates of 98.5%, 76.3%, 89%, 98% and 82.5%, respectively, with a false alarm rate was of 0.1831%.

In Ref. 125, the self-organized ACO-based algorithm described earlier[114] and SVM were combined for ID. In it, the SVM was used to find support vectors and generate a hyperplane that separated normal and abnormal data. All objects within the neighborhoods of the support vectors were then selected by an ACO clustering phase for the next SVM training phase. To reduce the false negative rate, a network connection was only confirmed as normal data when it was classified as "normal" by both of the classifiers. Unfortunately, no experimental results were presented to analyze the performance of the algorithm.

In Ref. 126, a three-level hybrid classifier was introduced. The algorithm hybridized a tree classifier algorithm, namely C4.5, and ACO. Motivated by the assumption that U2R and R2L attacks had close similarities to normal connections, in the 1st level, DOS and Probe types of attacks were classified from the dataset, while U2R, R2L and Normal connections were grouped as "others" using the C4.5 algorithm. Then, in the 2nd level, by using ACO, the "others" group was split into its corresponding U2R and R2L, and in the 3rd level, normal and abnormal connections were identified. The experimental results from testing on the KDD99 Cup dataset showed that the detection rates for DOS, Probe, U2R, R2L and Normal connections were 99.35%, 99.7%, 73.2%, 71.1% and 95.42%, respectively, while the false negative rate was 3.37% and the false alarm rate 9.1%. Also, it was noted that the ACO's performance for identifying U2R and R2L attacks was not sufficiently good, but unfortunately, no details of the used fitness function or representation scheme were discussed.

In Ref. 127, an ACO-based network attack graph algorithm (AntNAG) aimed at finding the minimum critical set of exploits that must be prevented to guarantee no attack scenario was possible was introduced. In it, after initializing all the parameters and pheromone trails at each generation, each ant incrementally constructed a critical set of exploits. A pheromone trail was assigned to each exploit to describe the desirability of including that exploit in an ant's solution. Finally, a local search was used to improve the overall performance of the algorithm, with the process continuing until a predefined stopping criterion was met. Although the algorithm showed good performance, the network attack graphs were very large and complex.

6.2.3. *Tracing-based ACO*

ACO was also used to trace the origin of an attack. In Refs. 128 and 129, an ID and response executed with agent mobility (IDReAM) was introduced. In it, the IDS was inspired by the behavior of the human immune system, and while the intrusion response system (IRS) was based on ACO, IDAs moved randomly. Once it accessed a machine, it probed the incoming events and subsequently computed the suspicion index (SI). If the SI was greater than a predefined threshold, the IDS launched an alert, and built and deployed a pheromone. The IRS's task was to find and go to the place where the alert was given by the IDAs, which was done by the ACO, in which once a pheromone was found, a tracking state was initiated by following the

pheromone back to its source. IDReAM's reliability for detecting and responding to intrusions was tested (based on resource consumption). From the results, it was found that the services and agents did not consume too much CPU or bandwidth.

Chen *et al.*[130] proposed an ACO trace-back approach for determining a DOS attack's original-source IP address. In it, ants were located on a victim node and initial values for the pheromones were set on each router. Each ant at each victim used the topological information to find all the neighboring routers, with the flow information and pheromone trails of neighboring nodes used to compute the movement probability. Then, the next router to move was chosen based on the previously calculated probability. All the ants did the same and then a pheromone trail's intensity was re-calculated. Preliminary experiments were conducted on a simulated network with NetFlow-enable, and although the results showed that the proposed algorithm was able to detect the DOS attack path, this was not the case in real environments.

7. Discussion

Over the last few decades, researchers have shown a great deal of interest in developing IDSs, with their algorithms based on expert systems, NNs, EAs, SI and FSs. Of them, those based on EAs and SI have shown good performances and can be categorized into four groups: (i) evolving classification rules; (ii) evolving fuzzy classification rules; (iii) finding the optimal set of clusters; and (iv) hybrid approaches, i.e., using EAs to find the optimal set of parameters of a NN and SVM.

Generally speaking, the main operators for EAs are encoding, selection, crossover and mutation, which play pivotal roles in the success of any EA. According to the literature, binary encoding has been the most common choice, while tournament, roulette wheel and rank selections have been used in different studies. Of the types of crossover operators, one-point, two-point and homologous have been the most widely used. It is worth mentioning that using the homologous crossover in LGP has shown the best performance to date,[84] while the bit-flip and random mutation operators have often been employed. In addition, evaluating how good each individual in a population is based on a proper fitness function is critical. Table 1 shows the fitness functions used in the literature in IDS-based EAs and SI, the details of which were described earlier, but it should also be noted that many research studies discussed in this paper did not describe the ones they used.

With the aim of maintaining diversity in the search space, niching techniques have been used with many EAs and SI. Of them, fitness sharing (reduces the fitness values of individuals with highly similar members), crowding (uses the most similar member for replacement), hamming distance (measures the similarity between two individuals) and the Euclidean distance approaches have commonly been used. However, they add complexity to any algorithm and there are some parameters which might affect performance.

Table 1. Common fitness functions used in literature.

34	$f = \alpha + \sum_{j=1}^{D} R_j x_j - \beta T_\varepsilon^2$
35	$f = N_\varepsilon - T'$
36	$\frac{\sum_{j=1}^{N_\varepsilon} (AE{\cdot}I)_j - \max(0, \sum_{j=1}^{N_\varepsilon} (AE{\cdot}I)_j - O_j)}{\sum_{j=1}^{N_\varepsilon} (AE{\cdot}I)_j}$
31	$f = 1 - \lvert\sigma - \varphi\rvert$
41, 44, 49, 50	$f = \frac{\alpha}{A} - \frac{\beta}{B}$
43	$f = \frac{a \times B}{A - b}$
45, 47, 49	$f = w_1 \times support + w_2 \times confidence$
46	$f = 1 - penalty$
48, 70, 79, 92, 94, 104	$f = \frac{a}{b}$
55	$f = w_1 \times S + w_2 \times C + w_3 \times L$
56, 57	$f = \sum_{p \in Class_{C_j}} PPF_p^{R_j}$
60, 61	(1) accuracy in terms of classification rate; (2) number of fuzzy rules; and (3) total length of fuzzy rules
62	$f = \begin{cases} 0, & \text{if } f_2 > k_{\max} \\ f_1 \times \left(1 - \left(\frac{f_2}{k_{\max}}\right)\right), & otherwise \end{cases}$
63	$f = \Theta \cdot \mathrm{vol}(R) + (1 - \Theta) \cdot n_e(R)$
66	Fitness function based on degrees of nearness and separation among clusters
69	$f = \frac{\sum_{j=1}^{N} w_{ij}}{\sigma_i^2}$
15	Cost per example ($CPE = \frac{\sum_{i=1}^{m} \sum_{j=1}^{m} CM(i,j) \cdot C(i,j)}{N}$)
33, 71–73, 96, 99, 105	Classification error of corresponding RBF network, SVM or DT
75	$f = \begin{cases} support, & if\ support < \varsigma \\ w_1 \times support + w_2 \times confidence_{norm}, & otherwise \end{cases}$
76, 77	(1) equal class fitness function; (2) variable weights fitness function; and (3) hierarchical fitness function
82	$f = 0.5 \cdot ((1 - DR) + FPR)$
4	$f = \sum_{R \in RS} w_1 \cdot Fit + w_2 \cdot \alpha_{new}(R)$
85	$f = (100 - \delta) - penalty$
100, 101	$f = \frac{TC}{TC + FC}$
102	$f = \frac{1}{e} + \frac{\lambda l}{\sigma}$
106	$f = \frac{1}{1 + \sum_{j=1}^{k} \sum_{x_i \in C_j} d(x_i, C_j)}$
113	Swarm similarity function was used for each ant

Evolving (fuzzy) classification rules using ACO has rarely been used in the literature, whereas many clustering-based ACOs for IDSs have been proposed. Similarly, utilizing ACO to obtain a response after detecting an intrusion has been paid a great deal of attention. Furthermore, using ACO with other machine-learning techniques has been employed to enhance the overall performance of algorithms.

As PSO can become stuck in local solutions, only a few research studies have used it to evolve (fuzzy) signatures, but researchers have placed more emphasis on hybridizing it with other techniques, i.e., using it to find the best set of parameters of a NN and SVM, or using it as a local search.

Although good results have been achieved by current EA and SI approaches, there are still avenues for further investigation, as presented below:

- Most of the work presented in the literature using EAs was based on GAs and GPs. However, as there are other EAs, such as DE and CMA-ES, which have shown better performances in solving many optimization problems, placing emphasis on them can be a future research direction.
- During the last decade, new advances have emerged in the field of EAs, such as self-adaptively configuring the best EA among EAs[131] or the most suitable operator in a set of operators.[132] Adapting such algorithms for ID may offer advantages for obtaining high detection rates. Although using an ensemble of machine-learning techniques has been proposed in Ref. 133, the abovementioned ideas are different and have demonstrated their superiority over many other techniques.
- A great deal of interest has been directed towards the multi-objective field over the last few decades. However, a limited number of publications has proposed solving the ID problem as a multi-objective problem, as most have converted it to a single-objective problem. From our point of view, the ID problem can be formulated as a multi-objective problem, which has not been done before, and then be solved using enhanced many or multi-objective techniques.
- As in large-scale optimization problems, it is beneficial to use decomposition techniques testing the effects of using them instead of feature selection mechanisms may be valuable.
- As previously mentioned, little interest has been shown in using PSO to evolve classification rules due to its drawback of becoming stuck in local solutions. However, as recent work[134] has shown that this can be tackled by adopting a self-adaptive mix of PSO, this could be beneficial for ID.
- Developing an IDS that can detect new attacks with a zero FPR is still, hopefully, a possibility.
- Most algorithms proposed in the literature had difficulty in obtaining high detection rates for U2R, R2L and Probe attacks. This was because the distribution in a test dataset was different from that in a training dataset or some attacks in the training data were rare, as mentioned in Sec. 3. However, designing accurate techniques is necessary.
- It is still essential to develop IDSs which are capable of identifying unknown intrusions (see Ref. 66).
- Due to the complexity, heterogeneity and dynamics of cloud computing systems using EAs and SI to detecting intrusions in them is still an interesting research direction.
- As mobile malware is adapting and evolving faster than security tools can learn to detect and evade its threats, using EAs and SI to detect it is a challenge.
- As virtual currencies have become popular and successful, using EAs and SI to protect against cybercriminals is beneficial.

- There is an absence of appropriate metrics and assessment methodologies, as well as a general framework, for evaluating and comparing alternative IDS techniques.
- Although mobile ad hoc networks have become attractive, using EAs and SI to reduce security risks has not been fully explored.

To sum up, a summary table about the performance of selected algorithms, considered in this paper, is provided in Appendix B. Also, the top 13 best algorithms are also indicated. Among these 13 algorithms, we found that there were seven GAs, one GP algorithm, one DE algorithm, two PSOs and two ACO based algorithms. Generally speaking, a very limited work on using DE for ID has been done; however, DE outperformed other EAs and SI approaches. This gives a new direction on improving/putting more emphasis on using DE when designing IDSs.

8. Conclusions and Future Directions

It is a fact that there have been many incidents of cyber-attacks over the last few years which have motivated researchers to develop IDSs using CI techniques, such as ANNs, FSs, evolutionary computation methods, AISs and SI.

Of these approaches, EA- (GAs, GP, EP and EDAs) and SI- (ACO and PSO) based algorithms were reviewed in this paper. Although their performances were encouraging, each had its own pros and cons, and moreover, a limited number of EAs has been developed for IDS. Current challenges in this field as well as new research directions were also discussed.

Acknowledgment

This work was supported in part by the Australian Centre for Cyber Security Research Funding Program, under Grant No. PS38135.

Appendix A. Supplementary Materials

Supplementary materials can be downloaded from https://sites.google.com/site/saberelsayed3/IJCIA-Supp.Material.pdf?attredirects=o&d=1.

Appendix B. A Summary of Selected EAs and SI based Approaches for Intrusion Detection. The "+" Sign Means Algorithm is among the Top 13 Performing Techniques Considered in This Paper

Refs.	Dataset	Algo.	Results
37, 38	DAPRA	GA	DRs were high, above average, average and low for detecting Probe, DOS, U2R and U2L attacks
41	KDD99	GA	DR = 97.8%; FR = 0.6877%
42	real-time data	GA	The performance was above average
44+	DARPA	GA	The best DR was 99.97%; best FR = 0.0305%
45	DARPA	GA	Its performance on the testing data was not sufficiently good
47	KDD99	GA	DR = 59%; FR = 0.1%.
48	KDD9	GA	DRs were 98.34%, 99.33%, 63.34%, 5.86% and 93.95% for Normal, DOS, U2R, R2L and Probe attacks, respectively. FPR and FNR were 1.658% and 0.7529%, respectively
49	KDD99	GA	DR = 96%
	KDD99	GA	DRs for Probe, DOS, U2R and R2L attacks were 99.38%, 75.36%, 96.77%, 21.52%, respectively, while FR = 2.80%
50+	KDD99	GA	DR = 99.87%; FPR = 0.003%.
51	KDD99	GA	DR for DOS = 99.4%, but was poor for all other types
55	DAPRA	GA	DR = 98.95%; FR of 7.0%.
56	KDD99	GA	DR = 99.08%; FR = 3.85%.
57	DAPRA & KDD99	GA	DR for normal, U2R, R2L and DOS were 97.6%, 84.74%, 92.4% and 98.76%, respectively; FR = 0.29%.
60, 61	KDD99	GA	DR = 99.24% and FPR = 1.1%
62	KDD99	GA	DRs were 95.8%, 54.10%, 97.40%, 10.9% and 6.9% for Normal, Probe, DOS, U2R and R2L classes, respectively
63+	KDD99 & DARPA	GA	For KDD99 DR = 98.30% and FPR 2.0%; For DARPA, DR = 98.33%.
66+	KDD99	GA	DR = 99.6%, while 0FPR = 0.2%.
67	KDD99	GA	DRs for DOS, U2R, R2L and Probe = 56%, 78%, 66% and 44%, respectively, and FR = 0.4%.
68	KDD99	GA	DR = 95% and low FR = 0.32%.
69+	KDD99	GA	DR = 99.20%, FR = 2.2%.
71+	DARPA	GA	DR = 99.4%, no details of the false alarm rate were provided
33	KDD99	GA	DR = 95%, no details of the false alarm rate were provided
72+	KDD99	GA	99%, no details of the false alarm rate were provided
74	KDD99	GA	$94.226\% \leq DR \leq 96.377\%$; $0.975\% \leq FR \leq 1.025\%$
75	DARPA	GP	FPR was 5.04%, FNR 5.23% and 57.14% rate for detecting unknown attacks
76, 77	KDD99	GP	DR ≈ 90%, no details of the false alarm rate were provided
78	KDD99	GP	DR > 95%, no details of the false alarm rate were provided
79	KDD99	GP	DR = 90.5812%; FR = 0.5648%.
80	KDD99	GP	DRs were 99.44%, 81.51%, 97.03%, 9.47% and 7.36% for Normal, Probe, DOS, U2R and R2L attacks, respectively, with FR = 0.5225%
82	Wi-Fi network	GP	DR = 90% with a FPR less than 1%.
84+	KDD99	GP	DRs for Normal, Probe, DOS, U2R and R2L attacks were 98.9%, 99.9%, 99.8%, 99.9% and 100%

Appendix B. (*Continued*)

Refs.	Dataset	Algo.	Results
4	KDD99 & DARPA	GP	DR, FPR and FNR were 94.4%, 7.2% and 5.0%, respectively
85, 88	KDD99	GP	DRs for normal, Probe, DOS, U2R and R2L attacks were 92.54%, 78.77%, 99.97%, 26.7% and 99.74%, respectively.
89	DARPA	EP	DRs for normal, Probe, DOS, U2R and R2L are 97.58%, 95.57%, 97.76%, 99.90% and 98.90%, respectively
91+	KDD99	DE	DR = 100, FPR = 0.489%
93	KDD99	PSO	DR = 92%, FR = 2.84%
94	DARPA	PSO	DR for Normal, U2R, R2L, DOS and Probe attacks were 90.8%, 49.1%, 79.4%, 97.5% and 81.2%, respectively.
95	KDD99	PSO	DR was 95.876% and FPR rate 2.125%
96	KDD99	PSO	DR = 96.01%, and FPR = 4.89%
97	KDD99	PSO	DR = 96.77% and FR = 8.01%
98	KDD99	PSO	DR = 95.585%, no details of the false alarm rate were provided
99	KDD99	PSO	DRs for normal, Probe, DOS, U2R and R2L attacks were 96.88%, 92.2%, 97.74%, 52.86% and 8.3%, respectively; FR = 0.61%.
100	KDD99	PSO	DRs for Probe, DOS, U2R and R2L attacks = 88.86%, 92.57%, 91.14% and 94.29%, respectively
101	KDD99	PSO	DRs for Normal, Probe, DOS, U2R and R2L attacks = 99.84%, 99.95%, 99.32%, 68.57% and 90.25%, respectively
102, 103	DARPA	PSO	97.2612%, no details of the false alarm rate were provided
104+	DARPA	PSO	99.8438%, no details of the false alarm rate were provided
105	DARPA	PSO	DR = 86.25%
106	KDD99	PSO	DRs for Probe, DOS and U2R ranged from 78% to 94% but that for R2L was only 22%
107+	KDD99	PSO	DR = 99.99% on 6 distributed nodes; FR = 1.35%.
108	KDD99	PSO	$95.036\% \leq DR \leq 96.698\%$, $0.964\% \leq FR \leq 1.015\%$
109	KDD99	ACO	DR = 92.23%; FR = 1.53%
113	KDD99	ACO	DR = 93.03%; FR = 8.24%
118+	DARPA	ACO	DR = 99.5%; FR = 2.7%
119+	KDD99	ACO	DR = 98.42%, FR = 0.14%
120	KDD99	ACO	DRs for Probe, DOS, and U2R & R2L were 99.4%, 95.2% and 98.7%, and FRs = 0.35%, 3.24% and 1.6%, respectively
122–124	DARPA	ACO	DRs for normal, U2R, R2L, DOS and Probe attacks were 98.5%, 76.3%, 89%, 98% and 82.5%, respectively; while FR was 0.1831%
126	KDD99	ACO	DRs for DOS, Probe, U2R, R2L and normal attacks were 99.35%, 99.7%, 73.2%, 71.1% and 95.42%, respectively; while the FR was 3.37%

References

1. K. Shafi, An online and adaptive signature-based approach for intrusion detection using learning classifier systems, Ph.D. dissertation (University of New South Wales Canberra, 2008).
2. V. Ramos and A. Abraham, ANTIDS: Self organized ant-based clustering model for intrusion detection system, in *Soft Computing as Transdisciplinary Science and*

Technology, eds. A. Abraham, Y. Dote, T. Furuhashi, M. Köppen, A. Ohuchi and Y. Ohsawa (Springer, Berlin, 2005), pp. 977–986.
3. D. Dasgupta, Computational intelligence in cyber security, in *Proc. 2006 IEEE Int. Conf. IEEE Computational Intelligence for Homeland Security and Personal Safety*, 2006, pp. 2–3.
4. S. Mabu, C. Chen, N. Lu, K. Shimada and K. Hirasawa, An intrusion-detection model based on fuzzy class-association-rule mining using genetic network programming, *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.* **41** (2011) 130–139.
5. J. Cannady, Artificial neural networks for misuse detection, in *National Information Systems Security Conf.*, 1998, pp. 368–381.
6. P. Baldi and K. Hornik, Neural networks and principal component analysis: Learning from examples without local minima, *Neural Netw.* **2** (1989) 53–58.
7. D. Whitley, S. Rana, J. Dzubera and K. E. Mathias, Evaluating evolutionary algorithms, *Artif. Intell.* **85** (1996) 245–276.
8. L. Fogel, J. Owens and M. Walsh, *Artificial Intelligence Through Simulated Evolution* (John Wiley & Sons, New York, 1966).
9. S. M. E. Elsayed, *Evolutionary Approach for Constrained Optimization* (Australian Defence Force Academy, 2012).
10. K. Deb, Multi-objective genetic algorithms: Problem difficulties and construction of test problems, *Evol. Comput.* **7** (1999) 205–230.
11. J. F. Kennedy, J. Kennedy and R. C. Eberhart, *Swarm Intelligence* (Morgan Kaufmann, 2001).
12. L. A. Zadeh, Fuzzy sets, *Inf. Control* **8** (1965) 338–353.
13. J. E. Dickerson, J. Juslin, O. Koukousoula and J. A. Dickerson, Fuzzy intrusion detection, in *Joint 9th IFSA World Congr. 20th NAFIPS Int. Conf.*, Vol. 1503, 2001, pp. 1506–1510.
14. J. E. Dickerson and J. A. Dickerson, Fuzzy network profiling for intrusion detection, in *19th Int. Conf. North American Fuzzy Information Processing Society*, 2000, pp. 301–306.
15. A. N. Toosi and M. Kahani, A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers, *Comput. Commun.* **30** (2007) 2201–2212.
16. G. Florez, S. M. Bridges and R. B. Vaughn, An improved algorithm for fuzzy data mining for intrusion detection, in *Annual Meeting of the North American Fuzzy Information Processing Society*, 2002, pp. 457–462.
17. C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin and W.-Y. Lin, Intrusion detection by machine learning: A review, *Expert Syst. Appl.* **36** (2009) 11994–12000.
18. C. Langin and S. Rahimi, Soft computing in intrusion detection: The state of the art, *J. Ambient Intell. Human. Comput.* **1** (2010) 133–145.
19. P. Kabiri and A. A. Ghorbani, Research on intrusion detection and response: A survey, *Int. J. Netw. Security* **1** (2005) 84–102.
20. A. Lazarevic, V. Kumar and J. Srivastava, Intrusion detection: A survey, in *Managing Cyber Threats*, eds. V. Kumar, J. Srivastava and A. Lazarevic (Springer, US, 2005), pp. 19–78.
21. F. Sabahi and A. Movaghar, Intrusion detection: A survey, in *The 3rd Int. Conf. Systems and Networks Communications*, 2008, pp. 23–26.
22. C. Kolias, G. Kambourakis and M. Maragoudakis, Swarm intelligence in intrusion detection: A survey, *Comput. Security* **30** (2011) 625–642.
23. S. X. Wu and W. Banzhaf, The use of computational intelligence in intrusion detection systems: A review, *Appl. Soft Comput.* **10** (2010) 1–35.

24. R. Heady, G. Luger, A. Maccabe and M. Servilla, The architecture of a network-level intrusion detection system, Department of Computer Science, College of Engineering, University of New Mexico, 1990.
25. J. P. Anderson, Computer security threat monitoring and surveillance, in Technical report, James P. Anderson Company, Fort Washington, Pennsylvania, 1980.
26. Z.-S. Pan, S.-C. Chen, G.-B. Hu and D.-Q. Zhang, Hybrid neural network and C4. 5 for misuse detection, in *IEEE Int. Conf. Machine Learning and Cybernetics*, 2003, pp. 2463–2467.
27. P. de Boer and M. Pels, Host-based intrusion detection systems, Informatics Institute, Amsterdam University, 2005.
28. D. Poole, A. Mackworth and R. Goebel, *Computational Intelligence* (Oxford University Press, Oxford, 1998).
29. The DARPA-Lincoln Dataset, 1998.
30. S. Hettich and S. D. Bay, The Third International Knowledge Discovery and Data Mining Tools Competition, 1999.
31. B. Balajinath and S. Raghavan, Intrusion detection through learning behavior model, *Comput. Commun.* **24** (2001) 1202–1212.
32. K. Shafi and H. A. Abbass, Evaluation of an adaptive genetic-based signature extraction system for network intrusion detection, *Pattern Anal. Appl.* **16** (2013) 549–566.
33. L. Wang, G. Yu, G. Wang and D. Wang, Method of evolutionary neural network-based intrusion detection, in *IEEE Int. Conf. Info-tech and Info-net*, Beijing, 2001, pp. 13–18.
34. M. Ludovic, GASSATA, a genetic algorithm as an alternative tool for security audit trails analysis, in *The First International Workshop on the Recent Advances in Intrusion Detection*, Louvain-la-Neuve, Belgium, 1998, pp. 1–11.
35. P. A. Diaz-gomez, Improved off-line intrusion detection using a genetic algorithm, in *The Seventh Int. Conf. Enterprise Information Systems*, Citeseer, 2005, pp. 66–73.
36. P. A. Diaz-Gomez and D. F. Hougen, A genetic algorithm approach for doing misuse detection in audit trail files, in *15th Int. Conf. Computing*, 2006, pp. 329–338.
37. F. Neri, Mining TCP/IP Traffic for Network Intrusion Detection by Using a Distributed Genetic Algorithm, in *Machine Learning: ECML*, 2000, eds. R. López de Mántaras and E. Plaza (Springer, Berlin, 2000), pp. 313–322.
38. M. Mischiatti and F. Neri, Applying local search and genetic evolution in concept learning systems to detect intrusion in computer networks, in *The 11th European Conf. Machine Learning (ECML '00)*, 2000.
39. N. Filippo, Exploring the power of genetic search in learning symbolic classifiers, *IEEE Trans. Pattern Anal. Mach. Intell.* **18** (1996) 1135–1141.
40. A. Giordana and F. Neri, Search-intensive concept induction, *Evol. Comput.* **3** (1995) 375–416.
41. A. Chittur, Model generation for an intrusion detection system using genetic algorithms, High School Honors Thesis, Ossining High School. In cooperation with Columbia University (2001).
42. D. Dasgupta and F. A. Gonzalez, An intelligent decision support system for intrusion detection and response, in *Information Assurance in Computer Networks* (Springer, 2001), pp. 1–14.
43. M. Pillai, J. H. Eloff and H. Venter, An approach to implement a network intrusion detection system using genetic algorithms, in *Proc. 2004 Annual Research Conf. South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries*, South African Institute for Computer Scientists and Information Technologists, 2004, pp. 221–221.

44. J. Guan, D.-X. Liu and B.-G. Cui, An induction learning approach for building intrusion detection models using genetic algorithms, in *The Fifth World Congr. Intelligent Control and Automation*, Vol. 4335, 2004, pp. 4339–4342.
45. R. H. Gong, M. Zulkernine and P. Abolmaesumi, A software implementation of a genetic algorithm based approach to network intrusion detection, in *IEEE Sixth Int. Conf. Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2005 and First ACIS International Workshop on Self-Assembling Wireless Networks. SNPD/SAWN 2005, pp. 246–253.
46. W. Li, Using genetic algorithm for network intrusion detection, in *Proc. United States Department of Energy Cyber Security Group* (2004), pp. 1–8.
47. S. Selvakani and R. Rajesh, Genetic algorithm for framing rules for intrusion detection, *IJCSNS* **7** (2007) 285–290.
48. T. Xia, G. Qu, S. Hariri and M. Yousif, An efficient network intrusion detection method based on information theory and genetic algorithm, in *24th IEEE Int. Performance, Computing, and Communications Conf.*, IPCCC, 2005, pp. 11–17.
49. Z. Banković, D. Stepanović, S. Bojanić and O. Nieto-Taladriz, Improving network security using genetic algorithm approach, *Comput. Electr. Eng.* **33** (2007) 438–451.
50. B. Abdullah, I. Abd-Alghafar, G. I. Salama and A. Abd-Alhafez, Performance evaluation of a genetic algorithm based approach to network intrusion detection system, in *13th Int. Conf. Aerospace Sciences and Aviation Technology*, Military Technical College, Kobry Elkobbah, Cairo, Egypt, 2009.
51. M. S. Hoque, M. Mukit, M. Bikas and A. Naser, An implementation of intrusion detection system using genetic algorithm, *Int. J. Netw. Security Appl.* **4** (2012) 109–120.
52. J. Gómez, C. Gil, R. Baños, A. L. Márquez, F. G. Montoya and M. G. Montoya, A Pareto-based multi-objective evolutionary algorithm for automatic rule generation in network intrusion detection systems, *Soft Comput.* **17** (2013) 255–263.
53. J. Gómez, C. Gil, R. Baños, A. L. Márquez, F. G. Montoya and M. G. Montoya, A multi-objective evolutionary algorithm for network intrusion detection systems, in *Advances in Computational Intelligence*, eds. J. Cabestany, I. Rojas and G. Joya (Springer, Berlin, 2011), pp. 73–80.
54. B. Caswell and J. Beale, Snort 2.1 intrusion detection, Syngress, 2004.
55. J. Gomez and D. Dasgupta, Evolving fuzzy classifiers for intrusion detection, in *Proc. 2002 IEEE Workshop on Information Assurance* (IEEE Computer Press, New York, 2002), pp. 321–323.
56. M. S. Abadeh, J. Habibi and C. Lucas, Intrusion detection using a fuzzy genetics-based learning algorithm, *J. Netw. Comput. Appl.* **30** (2007) 414–428.
57. M. Saniee Abadeh, J. Habibi, Z. Barzegar and M. Sergi, A parallel genetic local search algorithm for intrusion detection in computer networks, *Eng. Appl. Artif. Intell.* **20** (2007) 1058–1069.
58. M. S. Abadeh, H. Mohamadi and J. Habibi, Design and analysis of genetic fuzzy systems for intrusion detection in computer networks, *Expert Syst. Appl.* **38** (2011) 7067–7075.
59. F. Hoffmann, Combining boosting and evolutionary algorithms for learning of fuzzy classification rules, *Fuzzy Sets Syst.* **141** (2004) 47–58.
60. C.-H. Tsang, S. Kwong and H. Wang, Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection, *Pattern Recogn.* **40** (2007) 2373–2391.
61. C.-H. Tsang, S. Kwong and H. Wang, Anomaly intrusion detection using multi-objective genetic fuzzy system and agent-based evolutionary computation framework, in *The Fifth IEEE Int. Conf. Data Mining*, 2005, pp. 789–792.

62. T. Özyer, R. Alhajj and K. Barker, Intrusion detection by integrating boosting genetic fuzzy classifier and data mining criteria for rule pre-screening, *J. Netw. Comput. Appl.* **30** (2007) 99–113.
63. D. Dasgupta and F. González, An immunity-based technique to characterize intrusions in computer networks, *IEEE Trans. Evol. Comput.* **6** (2002) 281–291.
64. J. Gomez, F. Gonzalez and D. Dasgupta, An immuno-fuzzy approach to anomaly detection, in *The 12th IEEE Int. Conf. Fuzzy Systems, FUZZ '03*, 2003, Vol. 1212, pp. 1219–1224.
65. F. Gonzalez, J. Gomez, K. Madhavi and D. Dasgupta, An evolutionary approach to generate fuzzy anomaly (attack) signatures, in *Information Assurance Workshop*, 2003, IEEE Systems, Man and Cybernetics Society, 2003, pp. 251–259.
66. T. P. Fries, A fuzzy-genetic approach to network intrusion detection, in *Proc. 10th Annual Conf. Companion on Genetic and Evolutionary Computation*, ACM, Atlanta, GA, USA, 2008, pp. 2141–2146.
67. Y. Liu, K. Chen, X. Liao and W. Zhang, A genetic clustering method for intrusion detection, *Pattern Recogn.* **37** (2004) 927–942.
68. J.-L. Zhao, J.-F. Zhao and J.-J. Li, Intrusion detection based on clustering genetic algorithm, in *Proc. 2005 Int. Conf. Machine Learning and Cybernetics*, 2005, pp. 3911–3914.
69. E. Leon, O. Nasraoui and J. Gomez, Anomaly detection based on unsupervised niche clustering with application to network intrusion detection, in *IEEE Congr. Evolutionary Computation*, 2004, pp. 502–508.
70. S.-J. Han and S.-B. Cho, Evolutionary neural networks for anomaly detection based on the behavior of a program, *IEEE Trans. Syst. Man Cybern. B, Cybernetics* **36** (2005) 559–570.
71. A. Hofmann and B. Sick, Evolutionary optimization of radial basis function networks for intrusion detection, in *IEEE Proc. Int. Joint Conf. Neural Networks*, 2003, pp. 415–420.
72. D. S. Kim, H.-N. Nguyen and J. S. Park, Genetic algorithm to improve SVM based network intrusion detection system, in *The 19th Int. Conf. Advanced Information Networking and Applications*, 2005, pp. 155–158.
73. G. Stein, B. Chen, A. S. Wu and K. A. Hua, Decision tree classifier for network intrusion detection with GA-based feature selection, in *The 43rd Annual Southeast Regional Conf.*, Vol. 2, ACM, Kennesaw, Georgia, 2005, pp. 136–141.
74. F. Kuang, W. Xu and S. Zhang, A novel hybrid KPCA and SVM with GA model for intrusion detection, *Appl. Soft Comput.* **18** (2014) 178–184.
75. W. Lu and I. Traore, Detecting new forms of network intrusion using genetic programming, *Comput. Intell.* **20** (2004) 475–494.
76. S. Dong, M. I. Heywood and A. N. Zincir-Heywood, Training genetic programming on half a million patterns: An example from anomaly detection, *IEEE Trans. Evol. Comput.* **9** (2005) 225–239.
77. D. Song, M. Heywood and A. N. Zincir-Heywood, A linear genetic programming approach to intrusion detection, in Genetic and Evolutionary Computation — GECCO 2003, eds. E. Cantú-Paz, J. Foster, K. Deb, L. Davis, R. Roy, U.-M. O'Reilly, H.-G. Beyer, R. Standish, G. Kendall, S. Wilson, M. Harman, J. Wegener, D. Dasgupta, M. Potter, A. Schultz, K. Dowsland, N. Jonoska and J. Miller (Springer, Berlin, 2003), pp. 2325–2336.
78. A. Abraham, C. Grosan and C. Martin-Vide, Evolutionary design of intrusion detection programs, *Int. J. Netw. Security* **4** (2007) 328–339.
79. G. Folino, C. Pizzuti and G. Spezzano, GP ensemble for distributed intrusion detection systems, in *Pattern Recognition and Data Mining* (Springer, 2005), pp. 54–62.

80. G. Folino, C. Pizzuti and G. Spezzano, An ensemble-based evolutionary framework for coping with distributed intrusion detection, *Genet. Prog. Evol. Mach.* **11** (2010) 131–146.
81. A. Abraham, Evolutionary computation in intelligent network management, in *Evolutionary Computation in Data Mining* (Springer, 2005), pp. 189–210.
82. P. LaRoche and A. N. Zincir-Heywood, Genetic programming based WiFi data link layer attack detection, in *The 4th Annual Communication Networks and Services*, 2006, pp. 285–292.
83. M. I. Heywood and A. N. Zincir-Heywood, Dynamic page based crossover in linear genetic programming, *IEEE Trans. Syst. Man Cybern. B, Cybern.* **32** (2002) 380–388.
84. J. V. Hansen, P. B. Lowry, R. D. Meservy and D. M. McDonald, Genetic programming for prevention of cyberterrorism through dynamic and evolving intrusion detection, *Decision Support Syst.* **43** (2007) 1362–1374.
85. M. Crosbie and G. Spafford, Applying genetic programming to intrusion detection, in *Working Notes for the AAAI Symp. Genetic Programming* (MIT, Cambridge, MA, USA, 1995), pp. 1–8.
86. Y. Chen, A. Abraham and B. Yang, Feature selection and classification using flexible neural tree, *Neurocomputing* **70** (2006) 305–313.
87. Y. Chen, A. Abraham and B. Yang, Hybrid flexible neural-tree-based intrusion detection systems, *Int. J. Intell. Syst.* **22** (2007) 337–352.
88. A. Boukelif and K. M. Faraoun, Genetic programming approach for multi-category pattern classification applied to network intrusions detection, *Int. J. Comput. Intell. Appl.* **6** (2006) 77–99.
89. Y. Chen, Y. Zhang and A. Abraham, Estimation of distribution algorithm for optimization of neural networks for intrusion detection system, in *Artificial Intelligence and Soft Computing — ICAISC 2006*, eds. L. Rutkowski, R. Tadeusiewicz, L. Zadeh and J. Żurada (Springer, Berlin, 2006), pp. 9–18.
90. Z. Salek, F. M. Madani and R. Azmi, Intrusion detection using neuarl networks trained by differential evaluation algorithm, in *The 10th Int. ISC Conf. Information Security and Cryptology (ISCISC)*, 2013, pp. 1–6.
91. S. Elsayed, R. Sarker and J. Slay, Evaluating the performance of a differential evolution algorithm in anomaly detection, in *IEEE Congr. Evolutionary Computation*, Sendai, Japan, 2015.
92. C. Guolong, C. Qingliang and G. Wenzhong, A PSO-based approach to rule learning in network intrusion detection, in *Fuzzy Information and Engineering*, ed. B.-Y. Cao (Springer, Berlin, 2007), pp. 666–673.
93. C. Zhao and W.-P. Wang, An improved PSO-based rule extraction algorithm for intrusion detection, in *Int. Conf. Computational Intelligence and Natural Computing*, 2009, pp. 56–58.
94. M. S. Abadeh, J. Habibi and S. Aliari, Using a particle swarm optimization approach for evolutionary fuzzy rule learning: A case study of intrusion detection, in *Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU)*, 2006, pp. 2–7.
95. R. Ensafi, S. Dehghanzadeh, R. Mohammad and T. Akbarzadeh, Optimizing fuzzy K-means for network anomaly detection using PSO, in *IEEE/ACS Int. Conf. Computer Systems and Applications (AICCSA)*, 2008, pp. 686–693.
96. M. Ruhui, L. Yuan and L. Xing, Hybrid QPSO based wavelet neural networks for network anomaly detection, in *Second Workshop on Digital Media and its Application in Museum & Heritages*, 2007, pp. 442–447.

97. M. Ruhui, L. Yuan, L. Xing and W. Zhang, Network anomaly detection using RBF neural network with hybrid QPSO, in *IEEE Int. Conf. Networking, Sensing and Control (ICNSC)*, 2008, pp. 1284–1287.
98. R. Ma and Y. Liu, Wavelet Fuzzy neural network based on modified QPSO for network anomaly detection, *Appl. Mech. Mater.* **20–23** (2010) 1378–1384.
99. E. Michailidis, S. K. Katsikas and E. Georgopoulos, Intrusion detection using evolutionary neural networks, in *IEEE Panhellenic Conf. Informatics*, 2008, pp. 8–12.
100. C. Zhifeng and Q. Peide, Application of PSO-RBF neural network in network intrusion detection, in *Third Int. Symp. Intelligent Information Technology Application*, 2009, pp. 362–364.
101. S. Srinoy, Intrusion detection model based on particle swarm optimization and support vector machine, in *IEEE Symp. Computational Intelligence in Security and Defense Applications*, 2007, pp. 186–192.
102. T. W. Jie and L. J. Cheng, Intrusion detection quantitative analysis with support vector regression and particle swarm optimization algorithm, in *Int. Conf. Wireless Networks and Information Systems*, 2009, pp. 133–136.
103. Z. Tie-Jun, L. Yang and L. Jia, Research on intrusion detection of SVM based on PSO, in *Int. Conf. Machine Learning and Cybernetics*, 2009, pp. 1205–1209.
104. J. Wang, X. Hong, R.-R. Ren and T.-H. Li, A real-time intrusion detection system based on PSO-SVM, in *Int. Workshop Information Security and Application*, 2009, pp. 319–321.
105. L. Huaping, J. Yin and L. Sijia, A new intelligent intrusion detection method based on attribute reduction and parameters optimization of SVM, in *The Second Int. Workshop on Education Technology and Computer Science (ETCS)*, 2010, pp. 202–205.
106. X. Lizhong, S. Zhiqing and L. Gang, K-means algorithm based on particle swarm optimization algorithm for anomaly intrusion detection, in *The Sixth World Congr. Intelligent Control and Automation*, 2006, pp. 5854–5858.
107. H. Weiming, G. Jun, W. Yanguo, W. Ou and S. Maybank, Online adaboost-based parameterized methods for dynamic distributed network intrusion detection, *IEEE Trans. Cybern.* **44** (2014) 66–82.
108. F. Kuang, S. Zhang, Z. Jin and W. Xu, A novel SVM by combining kernel principal component analysis and improved chaotic particle swarm optimization for intrusion detection, *Soft Comput.* **19** (2014) 1–13.
109. J. Handl, J. Knowles and M. Dorigo, Strategies for the increased robustness of ant-based clustering, in *Engineering Self-Organising Systems*, eds. G. M. Serugendo, A. Karageorgos, O. Rana and F. Zambonelli (Springer, Berlin, 2004), pp. 90–104.
110. C.-H. Tsang and S. Kwong, Ant colony clustering and feature extraction for anomaly intrusion detection, in *Swarm Intelligence in Data Mining* (Springer, 2006), pp. 101–123.
111. W. Tsang and S. Kwong, Unsupervised anomaly intrusion detection using ant colony clustering model, in *Soft Computing as Transdisciplinary Science and Technology*, eds. A. Abraham, Y. Dote, T. Furuhashi, M. Köppen, A. Ohuchi and Y. Ohsawa (Springer, Berlin, 2005), pp. 223–232.
112. T. Chi-Ho and K. Sam, Multi-agent intrusion detection system in industrial network using ant colony clustering approach and unsupervised feature extraction, *IEEE Int. Conf. Industrial Technology*, 2005, pp. 51–56.
113. Y. Feng, Z.-F. Wu, K.-G. Wu, Z.-Y. Xiong and Y. Zhou, An unsupervised anomaly intrusion detection algorithm based on swarm intelligence, in *The Fourth Int. Conf. Machine Learning and Cybernetics*, 2005, pp. 3965–3969.

114. F. Yong, Z. Jiang, Y. Chun-Xiao and W. Zhong-Fu, Clustering based on self-organizing ant colony networks with application to intrusion detection, in *The Sixth Int. Conf. Intelligent Systems Design and Applications*, 2006, pp. 1077–1080.
115. Y. Feng, J. Zhong, Z.-Y. Xiong, C.-X. Ye and K.-G. Wu, Intrusion detection classifier based on dynamic SOM and swarm intelligence clustering, in *Advances in Cognitive Neurodynamics (ICCN 2007)*, eds. R. Wang, E. Shen and F. Gu (Springer, Netherlands, 2008), pp. 969–974.
116. Y. Feng, J. Zhong, Z.-Y. Xiong, C.-X. Ye and K.-G. Wu, Network anomaly detection based on DSOM and ACO clustering, in *Advances in Neural Networks (ISNN 2007)*, eds. D. Liu, S. Fei, Z. Hou, H. Zhang and C. Sun (Springer, Berlin, 2007), pp. 947–955.
117. R. S. Parpinelli, H. S. Lopes and A. A. Freitas, Data mining with an ant colony optimization algorithm, *IEEE Trans. Evol. Comput.* **6** (2002) 321–332.
118. E. Soroush, M. S. Abadeh and J. Habibi, A boosting ant-colony optimization algorithm for computer intrusion detection, in *The 2006 Int. Symp. Frontiers in Networking with Applications*, 2006, pp. 1–6.
119. J. He and D. Long, An improved ant-based classifier for intrusion detection, in *The Third Int. Conf. Natural Computation*, 2007, pp. 819–823.
120. G. Hai-Hua, Y. Hui-Hua and W. Xing-Yu, Ant colony optimization based network intrusion feature selection and detection, in *Proc. 2005 Int. Conf. Machine Learning and Cybernetics*, 2005, pp. 3871–3875.
121. S. Srinoy, An adaptive IDS model based on swarm intelligence and support vector machine, in *Int. Symp. Communications and Information Technologies*, 2006, pp. 584–589.
122. M. S. Abadeh, J. Habibi and E. Soroush, Induction of fuzzy classification systems via evolutionary ACO-based algorithms, *Computer* **35** (2008) 37–44.
123. M. S. Abadeh, J. Habibi and E. Soroush, Induction of fuzzy classification systems using evolutionary ACO-based algorithms, in *First Asia Int. Conf. Modelling & Simulation, AMS'07*, 2007, pp. 346–351.
124. M. S. Abadeh and J. Habibi, A hybridization of evolutionary fuzzy systems and ant colony optimization for intrusion detection, *ISC Int. J. Inf. Security* **2** (2010) 33–46.
125. Q. Zhang and W. Feng, Network intrusion detection by support vectors and ant colony, in *Proc. 2009 International Workshop on Information Security and Application*, 2009, pp. 639–642.
126. L. P. Rajeswari, A. Kannan and R. Baskaran, An escalated approach to ant colony clustering algorithm for intrusion detection system, in *Distributed Computing and Networking*, eds. S. Rao, M. Chatterjee, P. Jayanti, C. S. Murthy and S. Saha (Springer, Berlin, 2008), pp. 393–400.
127. M. Abadi and S. Jalili, An ant colony optimization algorithm for network vulnerability analysis, *Iranian J. Electr. Electron. Eng.* **2** (2006) 106–120.
128. N. Foukia, IDReAM: Intrusion detection and response executed with agent mobility architecture and implementation, in *Proc. Fourth Int. Joint Conf. Autonomous Agents and Multiagent Systems, ACM*, The Netherlands, 2005, pp. 264–270.
129. N. Foukia, IDReAM: Intrusion detection and response executed with agent mobility, in *Engineering Self-Organising Systems*, eds. S. Brueckner, G. M. Serugendo, A. Karageorgos and R. Nagpal (Springer, Berlin, 2005), pp. 227–239.
130. C.-M. Chen, B. Jeng, C. Yang and G. Lai, Tracing denial of service origin: Ant colony approach, in *Applications of Evolutionary Computing*, eds. F. Rothlauf, J. Branke, S. Cagnoni, E. Costa, C. Cotta, R. Drechsler, E. Lutton, P. Machado, J. Moore, J. Romero, G. Smith, G. Squillero and H. Takagi (Springer, Berlin, 2006), pp. 286–295.

131. S. M. Elsayed, R. A. Sarker and D. L. Essam, Adaptive configuration of evolutionary algorithms for constrained optimization, *Appl. Math. Comput.* **222** (2013) 680–711.
132. S. M. Elsayed, R. A. Sarker and D. L. Essam, Multi-operator based evolutionary algorithms for solving constrained optimization problems, *Comput. Oper. Res.* **38** (2011) 1877–1896.
133. S. Mukkamala, A. H. Sung and A. Abraham, Intrusion detection using an ensemble of intelligent paradigms, *J. Netw. Comput. Appl.* **28** (2005) 167–182.
134. S. M. Elsayed, R. A. Sarker and E. Mezura-Montes, Self-adaptive mix of particle swarm methodologies for constrained optimization, *Inf. Sci.* **27** (2014) 216–233.