Iraqi Journal of Statistical Sciences (25) 2013
The 6th Scientific Conference of the College of Computer Sciences & Mathematics
pp [194-209]

Intrusion Detection and Classification Using Ant Colony Optimization Algorithm

Dr. Mafaz Mohsin Khalil Al-Anezi* Omar Nazar Bader**
Zainab Mohammad Abdullah** Ayad Imad Atallah**

Abstract

Studies of ant colonies have contributed in abundance to the set of intelligent algorithms. The modeling of pheromone depositing by ants in their search for the shortest paths to food sources resulted in the development of shortest path optimization algorithms. Ant colony optimization (ACO) algorithms have been successfully applied to combinatorial optimization tasks especially to data mining classification problem.

Internet and local networks have become everywhere. So organizations are increasingly employing various systems that monitor IT security breaches because intrusion events are growing day by day.

Ant-based algorithms or ant colony optimization (ACO) algorithms can be applied to the data mining field to extract rule-based classifiers and have been applied successfully to combinatorial optimization problems. More recently, researches applied ACO to data mining classification problems, where they introduced a classification algorithm called Ant-Miner algorithm. The Ant-Miner algorithm is based on the behavior of ants in searching of food. The aim of this paper is to use an Ant Colony-based classification system (Ant_Miner algorithm) to extract a set of rules for detection and classification, and it obtained a hopeful classification accuracy.

كشف التطفل وتصنيفه باستخدام خوارزمية مستعمرة النمل المثلى

المستخلص

ساهمت الدراسات لمستعمرات النمل في كثير من الخوارزميات الذكائية. ففقد أنتج نموذج المواد العطرية المترسبة من النمل أثناء بحثها عن أقصر طريق لمصدر الطعام عدة خوارزميات أمثلية لإيجاد الطريق الأقصر.

*Lecturer / Dept. Computer Sciences / Computer Sciences and Mathematics College / Mosul University

** researcher / Dept. Computer Sciences / Computer Sciences and Mathematics College / Mosul University

Received Date 9/10/2013 **Accept Date 21/11/2013**

ونجحت خوارزميات مستعمرات النمل المثلى في تطبيقها على مسائل الامثلية وخاصة في مسائل التصنيف وتنقيب البيانات.

أصبح الانترنت والشبكات المحلية في كل مكان، هذا الذي اجبر المنظمات على توظيف عدة أنظمة لتراقب أمنية تقنية المعلومات من حوادث التطفل المتزايدة يوما بعد يوم.

بإمكان الخوارزميات المعتمدة على النمل أو خوارزميات مستعمرات النمل المثلى ACO أن تطبق في حقل تنقيب البيانات لاستخراج قواعد التصنيف وطبقت بنجاح على مسائل الامثلية الإحصائية. مؤخرا طبق الباحثون خوارزميات مستعمرات النمل في مسائل تصنيف تنقيب البيانات بتقديم خوارزمية تصنيف تدعى Ant-Miner. تعتمد خوارزمية Ant-Miner على سلوك النمل في البحث عن الطعام. الهدف من هذا البحث هو استخدام نظام تصنيف يعتمد على مجتمعات النمل (خوارزمية Ant-Miner) لاستخراج مجموعة من قواعد الكشف والتصنيف، وقد حقق دقة تصنيف واعدة.

1. INTRODUCTION

In computer science and operations research, the ant colony optimization algorithm (ACO) is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs [Wikipedia, 2013]. This algorithm is a member of the ant colony algorithms family, in swarm intelligence methods, and it constitutes some metaheuristic optimizations. Initially proposed by Marco Dorigo in 1992 in his PhD thesis, the first algorithm was aiming to search for an optimal path in a graph, based on the behavior of ants seeking a path between their colony and a source of food. The original idea has since diversified to solve a wider class of numerical problems, and as a result, several problems have emerged, drawing on various aspects of the behavior of ants [Wikipedia, 2013].

The essential trait of ACO algorithms is the combination of a priori information about the structure of a promising solution with a posteriori information about the structure of previously obtained good solutions [Maniezzo, 2003].

Meta heuristic algorithms are algorithms which, in order to escape from local optima, drive some basic heuristic: either a constructive heuristic starting from a null solution and adding elements to build a good complete one, or a local search heuristic starting from a complete solution and iteratively modifying some of its elements in order to achieve a better one. The meta heuristic part permits the low level heuristic to obtain solutions better than those it could have achieved alone, even if iterated. Usually, the controlling mechanism is achieved either by constraining or by randomizing the set of local neighbor

solutions to consider in local search (as is the case of simulated annealing or tabu search), or by combining elements taken by different solutions (as is the case of evolution strategies and genetic or bionomic algorithms) [Maniezzo, 2003].

Classification is a data mining technique studied by statisticians and machine leaning researchers for a very long period of time. It is a process in which classes are predefined and a classifier is built (learnt) which allocates data samples to classes. Prediction rules are often represented in the IF THEN form, and it can be described as: IF<conditions> THEN <class>. In each rule, the <condition> part (the antecedent) usually contains a logic combination of predictor attributes in the form: term1 AND term2 AND…AND term¡. And each term is a triple <attribute, operation, value>, such as <Color = Red>. The consequent part <class> shows the predicted class whose cases satisfy the <condition>. Using these rules, people can make prediction of a certain phenomenon or gain insights from large amount of data [Seidlová, 2005].

This paper, studied ant colony algorithms inspired by the behavior of ants during searching/finding paths from the nest to food sources. Social insects like ants, bees and termites work by themselves in their simple tasks, independently of others members of the colony. However, when they act as a community, they are able to solve complex problems emerging in their daily lives, by means of mutual cooperation. This emergent behavior of a group of social insects is known as “swarm intelligence”. Ants are able to find the shortest path between a food source and the nest without the aid of visual information, and also to adapt to a changing environment. It was found that the way ants communicate with each other is based on pheromone trails. While ants move, they drop a certain amount of pheromone on the floor, leaving behind a trail of this substance that can be followed by other ants. The more ants follow a pheromone trail, the more attractive the trail becomes to be followed in the near future [Seidlová, 2005]. The basic idea is illustrated in figure 1.

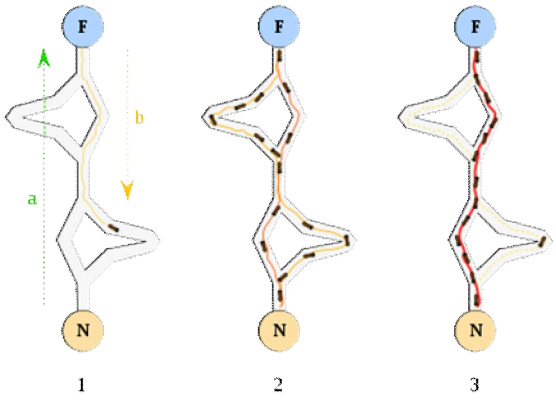

Figure 1: Ants follow a path between Nest and Food [9].

Shortening in the natural world, ants (initially) wander randomly, and upon finding food return to their colony while laying down pheromone trails. If other ants find such a path, they are likely not to keep travelling at random, but to instead follow the trail, returning and reinforcing it if they eventually find food [Wikipedia, 2013].

Over time, however, the pheromone trail starts to evaporate, thus reducing its attractive strength. The more time it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. A short path, by comparison, gets marched over more frequently, and thus the pheromone density becomes higher on shorter paths than longer ones. Pheromone evaporation also has the advantage of avoiding the convergence to a locally optimal solution. If there were no evaporation at all, the paths chosen by the first ants would tend to be excessively attractive to the following ones. In that case, the exploration of the solution space would be constrained [Wikipedia, 2013].

Thus, when one ant finds a good (i.e., short) path from the colony to a food source, other ants are more likely to follow that path, and positive feedback eventually leads to all the ants' following a single path. The idea of the ant colony algorithm is to mimic this behavior with "simulated ants" walking around the graph representing the problem to solve [Wikipedia, 2013].

Today it is very important to provide a high level security to protect highly sensitive and private information. Intrusion Detection System is an essential technology in Network Security. Nowadays researchers have interested on intrusion detection system using Data mining techniques as an artful skill. IDS is a software or hardware device that deals with attacks by collecting information from a variety of system and network sources, then analyzing symptoms of security problems [Jaiganesh, 2013].

Every computer is always at risk for unauthorized and intrusion, however, with sensitive and private information are at a higher risk. Intrusion Detection is a key technique in Information Security plays an important role detecting different types of attacks and secures the network system. Intrusion Detection is the process of observing and analysing the events arising in a computer or network system to identify all security problems. IDS provides three important security functions; monitor, detect and respond to unauthorized activities. Intrusion Detection System monitors the operations of firewalls, routers, management servers and files critical to other security mechanisms. Intrusion Detection System can make the security management of system by non-expert staff possible by providing user friendly interface [Jaiganesh, 2013].

Classification is classic data mining technique which is used to predict association for data instances. Classification techniques evaluate and classify the data into known classes. Each data sample is marked with a known class label. Also these techniques are used to learn a model using the training set data sample. This model is used to classify the data samples as anomalous behaviour data or the normal behaviour data [Jaiganesh, 2013].

2. Principles of Ant Behavior

Ants are *social* insects living within a collective (a family or colony). Some two percent of insects are social, and ants account for half of these. The number of ants in a single colony may vary from 30 to tens of millions. Ants are dominant in the Amazon basin, constituting more than 30% of the biomass of the local forests. The behavior of ants in transporting food, overcoming obstacles, building anthills, and other operations is almost optimal. Principles of ant behavior have withstood the proof of one hundred million years, after they had “colonized” the Earth. Ant colonies are amazingly survivable: a reduction of up to 40% of insects has practically no effect on the functioning of the whole society. A mass destruction of ants (for example, resulting from a chemical treatment of their habitat) leads to the consolidation of insects from the neighboring anthills into one family to save the society [Shtovba, 2005].

The social behavior of ants is based on *self-organization*, a set of dynamical mechanisms ensuring that the system can achieve its global aim through low-level interactions between its elements. A key feature of this interaction is that the system elements use*only local information.* In this case, any centralized control and reference to the global pattern representing the system in the external world are ruled out. Self-

organization is a result of the interaction between the following four components [Shtovba, 2005]:
(1) multiple renewal;
(2) randomness;
(3) positive feedback;
(4) negative feedback.

There are two ways of information transfer between ants: a direct communication (which includes food exchange and mandible, visual, and chemical contacts) and an indirect communication, which is called stigmergy. *Stigmergy* is a form of communication separated in time, when one participant of the communication modifies the environment, and the others make use of this information later, when they occur in a neighborhood of the modified environment. Biologically, stigmergy is realized through *pheromones*, a special secretory chemical that is deposited as trail by ants when they move. The higher the pheromone concentration on the path, the more the number of ants moving along it. With time, the pheromones evaporate, which allows the ants to adapt their behavior when the environment is modified. The distribution of pheromones is a sort of dynamically varying global memory of the anthill. At any moment, an ant can sense and change only one local cell of this global memory. On the example of experiments with ants on an asymmetric bridge, it demonstrated how the cooperative behavior of ants makes it possible to find the shortest path to food. The asymmetric bridge (Figure 2) connects the ant nest with the food source by two branches of different length. The experiments were carried out with a laboratory colony of Argentine ants (*Iridomurmex humilis*), which deposit pheromones on the paths both from and to the nest. The scheme of the experiments was as follows [Shtovba, 2005]:
(1) the bridge A-B-C-D was constructed;
(2) the gate at point A was opened;
(3) the numbers of ants selecting the longer (A-C-D) and shorter branches of the bridge were counted.

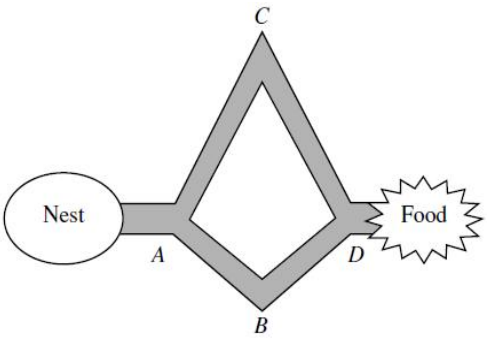

Figure 2: An asymmetric bridge [5]

At the early stage of the experiments, both branches have been selected by the ants at about the same rate. In some time, almost all ants choose to move along the shortest route A-B-D, which is explained in the following way. First, the branches were free of pheromones; therefore, the A-C-D and A-B-D branches have been selected with equal rate. The ants that selected the shorter route A-B-D-B-A returned sooner with food to the nest and laid pheromone trails on this shorter branch. When they had to select the next time, the ants preferred to move along the shorter branch of the bridge, since the concentration of pheromones on it is higher. Therefore, the pheromones are accumulated faster on the branch A-B-D, attracting the ants to select the shortest route.

In general, an ACO algorithm can be applied to any optimization problem as far as it can be defined [Shi, 2011]:

(1) An appropriate graph representation for problem. The problem must be described as a graph with a set of nodes and edges between nodes. The graph should accurately represent all states and transitions between states.
(2) Heuristic desirability. A suitable heuristic measure can be defined to describe the goodness of edges from one node to every other linked node in the graph.
(3) Solution construction mechanism. A method must be defined for building possible solutions.
(4) An autocatalytic feedback process. A suitable method of updating the pheromone levels on edges is required.
(5) A constraint-satisfaction method. A mechanism is needed to ensure that only feasible solutions are constructed.

3. Applications

Ant colony optimization algorithms have been applied to many combinatorial optimization problems, ranging from quadratic assignment to protein folding or routing vehicles and a lot of derived methods have been adapted to dynamic problems in real variables, stochastic problems, multi-targets and parallel implementations. It has also been used to produce near-optimal solutions to the travelling salesman problem. They have an advantage over simulated annealing and genetic algorithm approaches of similar problems when the graph may change dynamically; the ant colony algorithm can be run continuously and adapt to changes in real time. This is of interest in network routing and urban transportation systems [Wikipedia, 2013].

Other applications of ant colony optimization include routing optimization in telecommunications networks, graph coloring, scheduling and solving the quadratic assignment problem. Studies of the

nest building of ants and bees resulted in the development of clustering and structural optimization algorithms.

Recently, ACO algorithm has been successfully applied in many fields, such as vehicle routing and data mining. And it is particularly attractive for feature selection as there seems to be no heuristic that can guide search to the optimal minimal subset every time [Shi, 2011].

4. Ant Colony Optimization

ACO is a class of algorithms, whose first member, called Ant System, was initially proposed by Colorni, Dorigo and Maniezzo. The main underlying idea, loosely inspired by the behavior of real ants, is that of a parallel search over several constructive computational threads based on local problem data and on a dynamic memory structure containing information on the quality of previously obtained result. The collective behavior emerging from the interaction of the different search threads has proved effective in solving combinatorial optimization (CO) problems[Wong, 2011].

The original idea comes from observing the exploitation of food resources among ants, in which ants' individually limited cognitive abilities have collectively been able to find the shortest path between a food source and the nest [Wikipedia, 2013][Wong, 2011], see previous figure 1.

1. The first ant wanders randomly until it finds the food source (F), then it returns to the nest (N), laying a pheromone trail.
2. Other ants follow one of the paths at random, also laying pheromone trails. Since the ants on the shortest path lay pheromone trails faster, this path gets reinforced with more pheromone, making it more appealing to future ants.
3. The ants become increasingly likely to follow the shortest path since it is constantly reinforced with a larger amount of pheromones. The pheromone trails of the longer paths evaporate.

In a series of experiments on a colony of ants with a choice between two unequal length paths leading to a source of food, biologists have observed that ants tended to use the shortest route [Wikipedia, 2013].

The paradigm for optimization problems that can be expressed as finding short paths in a graph as in table (1), and has the following goals [Wong, 2011]:

- To design technical systems for optimization, and
- NOT to design an accurate model of nature.

Table 1 : Paradigm of Ant's optimization problems [Wong, 2011].

Nature	**Computer Science**

Natural habitat	Graph (nodes and edges)
Nest and food	Nodes in the graph: start and destination
Ants	Agents, our artificial ants
Visibility	The reciprocal of distance, η
Pheromones	Artificial pheromones ,τ
Foraging behavior	Random walk through graph (guided by pheromones)

5. Antminer Algorithm

The core of the algorithm is the incremental construction of classification rules of the form IF (term1) AND (term2) AND ... AND (termn) THEN (class) from data. Each term is an attribute-value pair related by an operator (relational operator). The IF part corresponds to the rule's antecedent and the THEN part is the rule's consequent, which represents the class to be predicted by the rule. An example term is "Color = red". The attribute's name is "color" and "red" is one of its possible values. Since we use only "=" any continuous (real-valued) attributes present in the data have to be discretized in a preprocessing step. Pseudo-code of AntMiner is illustrated [Seidlová, 2005]:

TrainingSet = {all training cases};
DiscoveredRuleList = [];/*rule list is initialized with an empty list*/
WHILE (TrainingSet >= Max_Uncovered_Cases)
  i = 1; /* ant index*/
  No_Ants_Converge = 1; /* convergence test index*/
  Initialize all trails with the same amount of pheromone;
  REPEAT
    Ant_i starts with an empty rule and incrementally constructs a classification rule R_i, by adding one term at a time to the current rule;
    Prune rule R_i;
    Update the pheromone of all trails, by increasing pheromone in the trail followed by Ant_i, (in proportion to the quality of R_i) and decreasing pheromone in the other trails (simulating pheromone evaporation);
    IF (R_i is equal to R_i – 1) /*update convergence test*/
      THEN No_Ants_Converge = No_Ants_Converge +1;
      ELSE No_Ants_Converge = 1;
    ENDIF
    I=i+1;
  UNTIL (I >= No_of_Ants) OR (No_Ants_Converge >= No_Rules_Converge)
  Choose the best rule R_{best} among all rules R_i constructed by all the ants;

Add rule R_{best} to DiscoveredRuleList;
TrainingSet = TrainingSet – {set of cases correctly covered by R_{best}};
END WHILE

AntMiner algorithm works as follows: It starts with an empty rule list and iteratively adds one rule at a time to that list while the number of uncovered training examples is greater than a user specified maximum value. A single ant starts with an empty rule and adds one term at a time to the rule antecedent. The ant keeps adding a term to the partial rule until any term added to the antecedent would make the rule cover less training examples than a user-specified threshold, which would make the rule too specific and unreliable, or all attributes have already been used by the ant.

Once this process of rule construction has finished, first the rule constructed by the ant is pruned to remove irrelevant terms from the rule antecedent. Then, the consequent of the rule is chosen to be the class value most frequent among the set of training examples covered by the rule. Finally, pheromone trails are updated and another ant starts to construct a new rule. The process of constructing a rule is repeated until a user-specified number of rules have been reached, or the current ant has constructed a rule that is exactly the same as rules constructed by a predefined number of previous ants, which works as a rule convergence test. The best rule, based on a quality measure Q, found along this iterative process is added to the rule list and the correctly classified training examples are removed from the training set [Seidlová, 2005].

The heuristic function η is based on the amount of information associated with the attribute i with value j [Mahmood, 2011][Seidlová, 2005].

$$infoT_{ij} = -\sum_{w=1}^{k}\left(\frac{freqT_{ij}^{w}}{|T_{ij}|}\right) * \log_2\left(\frac{freqT_{ij}^{w}}{|T_{ij}|}\right) \qquad \ldots\ldots\ldots\ldots \quad (1)$$

where: k is the number of classes in the dataset; $|T_{ij}|$ is the total number of cases in the data partition T_{ij} (partition that contains the cases where the attribute i is equal to the value j); $freq\ T_{ij}^{w}$ represents the number of cases in T_{ij} that belong to class w.

In Ant-Miner, the heuristic value is taken to be an information theoretic measure for the quality of the term to be added to the rule. The quality here is measured in terms of the entropy for preferring this term to the others; and is given by [Mahmood, 2011][Seidlová, 2005]:

$$\eta_{ij} = \frac{log_2(k) - infoT_{ij}}{\sum_i^a \sum_j^{b_i} log_2(k) - infoT_{ij}} \qquad \ldots\ldots\ldots\ldots \quad (2)$$

where a is the total number of attributes and b_i is the number of values in the domain of attribute i.

Authors believe that the AntMiner algorithm does not need accurate information in this heuristic value and propose the following easily computable equation [Mahmood, 2011][Seidlová, 2005]:

$$\eta_{ij} = \frac{majority\_class\ T_{ij}}{|T_{ij}|} \quad \ldots\ldots\ldots\ldots (3)$$

where $majority\_class T_{ij}$ is the majority class in partition T_{ij}. This heuristic has less computational cost and has the same accuracy as original AntMiner.

After each ant completes the construction of its rule, pheromone level updating is carried out in original algorithms as follows [Mahmood, 2011][Seidlová, 2005]:

$$\tau_{ij}(t) = (1-\rho) \cdot \tau_{ij}(t-1) + \left(1 - \frac{1}{1+Q}\right) \cdot \tau_{ij}(t-1) \ldots\ldots\ldots\ldots (4)$$

where ρ is the pheromone evaporation rate, large value of ρ indicates a fast evaporation. Typical values for this evaporation factor lie in the range <0.8,0.99>

The quality Q of a rule is measured as [Seidlová, 2005]:

$$Q = \left(\frac{TruePos}{TruePos + FalseNeg}\right) * \left(\frac{TrueNeg}{FalsePos + TrueNeg}\right) \ldots\ldots\ldots\ldots (5)$$

where:

- *TruePos* (true positives) is the number of cases covered by the rule that have the class predicted by the rule;
- *FalsePos* (false positives) is the number of cases covered by the rule that have a class different from the class predicted by the rule;
- *FalseNeg* (false negatives) is the number of cases that are not covered by the rule but that have the class predicted by the rule;
- *TrueNeg* (true negatives) is the number of cases that are not covered by the rule and do not have the class predicted by the rule.

6. Datasets used in the Experiments

KDD-Cup is the widely used dataset for training and testing of IDS. There are a total of 41 features which are classified into Basic, Content and Traffic features. KDD-Cup is developed on the basis of DARPA'98 data and this data has been criticized by McHugh. As a result, some of the inherited issues also exist in KDD-Cup like redundancy of similar records and complexity level of data behavior. NSL-KDD is an advanced

version of KDD-Cup dataset and doesn't suffer from the shortcomings in KDD-Cup. The following are unique features for which we preferred NSL-KDD over KDD-Cup [Imran, 2012][Tavallaee, 2009].

No Redundancy of Records: NSL-KDD doesn't include redundant records in the train set; hence the classifiers would not be biased towards more frequent records.

No Duplication: There is no duplicate record in the proposed test sets; therefore, the performance of the learners is not biased by the approaches which have better detection rates on the frequent records.

Less Complexity Level: The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary widely, which makes it more efficient to have an accurate evaluation of different learning techniques.

Reasonable Records: The number of records in the train and test sets is reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

NSL-KDD dataset is a reduced version of the original KDD 99 dataset. NSL-KDD consists of the same features as KDD 99. The KDD99 dataset consists of 41 features and one class attribute. The class attribute has 21 classes that fall under four types of attacks: Probe attacks, User to Root (U2R) attacks, Remote to Local (R2L) attacks and Denial of Service (DoS) attacks. This dataset has a binary class attribute. Also, it has a reasonable number of training and test instances which makes it practical to run the experiments on. The numbers of records in the train and test sets are reasonable, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion. Consequently, evaluation results of different research works will be consistent and comparable.

The only preprocessing operation is convert characters value to numeric values: There are three features in each packets have characters values (protocol type, Service, Flag), which must converted to numeric value.

7. Results and Discussion

AntMiner algorithm includes several parameters: Number of Ants, Minimum number of cases per rule, Maximum number of uncovered cases in the training set, Number of rules used to test convergence of the ants, Evaporation factor. The greatest influence on the accuracy and calculation time have Number of Ants and Evaporation factor.

Increasing evaporation factor will result in a slower convergence process and no significant increase in accuracy. The calculation time increases with the number of ants and with the evaporation factor. Figure 3 show the flowchart of the work. The work using the ACO (Ant-Miner algorithm) contain four stages as follow:

1. **Initiating data stage:** read the training datasets (125973 records), separate them into five classes: Normal, DOS, U2R, Probe, and R2L. Which is mean that all the attacks types belong to one from the four classes (DOS, U2R, Probe, R2L) like smurf attack is belong to DOS class as in table 2.
2. **Calculate features repetition stage**: for each one of five classes and for each one of 41 features of records calculate the repetition number of each feature value. These values used in heuristic function and for initiate pheromone.
3. **Training stage**: This is a building rules operation and is done for each one of five previously mentioned classes. At first determine ants number, Max_Uncovered_Cases, No_Rules_Converge, then start building the rules in calculating the probability of adding each feature value to the classification rule by merge the pheromone with heuristic function value. Each produced rule was pruned to by eliminated some inefficient features value and this is done by calculating Quality measure. At last the best rule is chosen (saved) and eliminate the rest produced rules from this step.
4. Testing stage: apply the produced rules from training stage on the completed testing data sets (22544 records). And the detection rates obtained are: 82.81% DOS by 1708 production rules, 97.85% U2R by 3 production rules, 96.23% probe by 59 production rules, 38.33% R2L by 2 production rules, and 23.80% Normal by 60 production rules. These results show the effectiveness of ACO algorithms in the field of network security.

Table 2: Attacks types in training and testing dataset.

No.	Attack Name	Classes	No.	Attack Name	Classes
1	Smurf	DOS	20	Xlock	R2L
2	Buffer_overflow	DOS	21	multihop	R2L
3	Pod	DOS	22	xsnoop	R2L
4	Apache2	DOS	23	Sendmial	R2L
5	Udpstorm	DOS	24	Guess-passwd	R2L
6	Process table	DOS	25	Phf	R2L
7	Neptune	DOS	26	Warezmaster	R2L
8	Back	DOS	27	Imap	R2L
9	Mailbomb	DOS	28	Httptunnel	R2L

10	Teardrop	DOS	29	Worm	R2L
11	Land	DOS	30	ftp-write	R2L
12	Ipsweep	PROBE	31	Sqlattack	R2L
13	Saint	PROBE	32	snmpguess	R2L
14	Satan	PROBE	33	Perl	U2R
15	Mscan	PROBE	34	Xterm	U2R
16	Portsweep	PROBE	35	Ps	U2R
17	Nmap	PROBE	36	Root kit	U2R
18	Named	R2L	37	load module	U2R
19	Snmpgetattack	R2L			

8. Conclusions

The general paradigm for optimization problems is inspired from nature, but with smarter agents. Paths found by ant represent solutions for the problem, and the choice of path influenced by previous experience. Pheromones as model of collective memory of a swarm and tunable parameters that affect performance.

The Ant-Miner algorithm approved it is better than neural network and genetic algorithm in classification and in the time taken up with training and testing, it deal with numerical and alphabetic data without transformation, the rules products from it easy to understand, and it is also flexible with fuzzy data.

The suggestion for more development solutions are merging the Ant-Miner algorithm with neural network or genetic algorithm, feature reduction or selection.

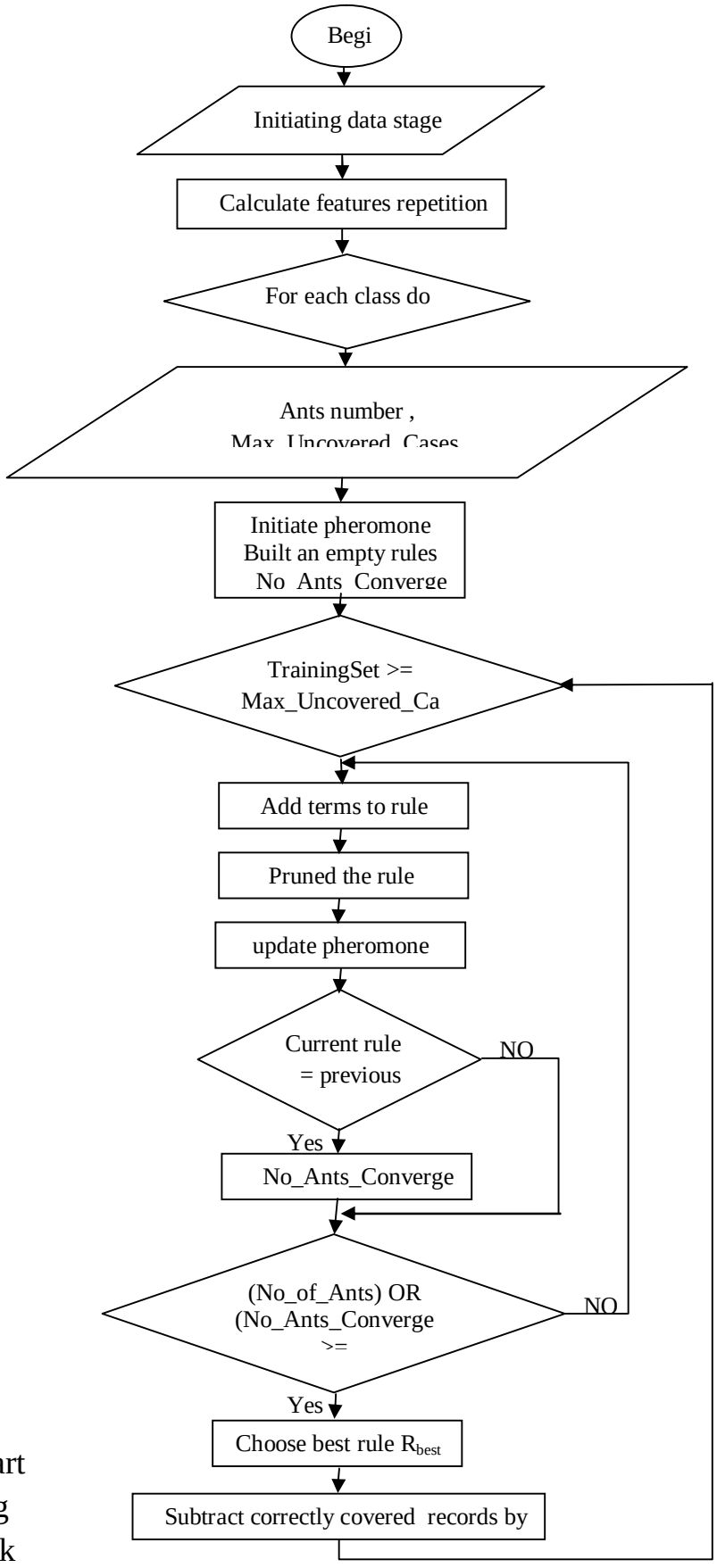

Figure 3: Flowchart show the training stages of the work using Ant-Miner

References

[1] Imran H. M., Abdullah A., Hussain M., Palaniappan S., and Ahmad I., 2012, "Intrusions Detection based on Optimum Features Subset and Efficient Dataset Selection", International Journal of Engineering and Innovative Technology (IJEIT), pp 265-270, Volume 2, Issue 6, December 2012.

[2] Jaiganesh V., Mangayarkarasi S., and Dr. Sumathi P., 2013, "Intrusion Detection Systems: A Survey and Analysis of Classification Techniques", International Journal of Advanced Research in Computer and Communication Engineering Vol. 2, Issue 4, pp1629-1635.

[3] Mahmood S. M., 2011,"Using Ant and Self-Organization Maps Algorithms To Detect and Classify Intrusion In Computer Networks", M.Sc./Thesis, University of Mosul College of Computer Sciences And Mathematics.

[4] Maniezzo V., Gambardella L. M., de Luigi F., 2003, " Ant colony optimization", work was partially supported by the Future & Emerging Technologies unit of the European Commission through Project BISON (IST-2001-38923).

[5] Seidlová R., Poživil J., 2005, " Implementation Of Ant Colony Algorithms In Matlab", Institute of Chemical Technology, Department of Computing and Control Engineering Technická 5, Prague 6, 166 28, Czech Republic.

[6] Shi L., Xi L., Ma X., Weng M, and Hu X., 2011, "A novel ensemble algorithm for biomedical classification based on Ant Colony Optimization", pp5674–5683, Applied Soft Computing.

[7] Shtovba S. D., 2005, "Ant Algorithms: Theory and Applications", Programming and Computer Software, Vol. 31, No. 4, 2005, pp. 167–178.

[8] Tavallaee M., Bagheri, E., Wei Lu, and Ghorbani A., 2009, "A detailed analysis of the KDD CUP 99 data set", Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009), 1-6.

[9] Wikipedia, the free encyclopedia, 2013, "Ant colony optimization algorithms".

[10] Wong E., Summers P., Ku R., and Xie P., 2011, "Ant Colony Optimization", PIC 10C SPRING 2011.