Rule Induction using Ant-Miner Algorithm

NimmyCleetus, Dhanya K.A

Abstract—In this paper, we extracted robust rules for identifying different forms of network attacks. Initially, we mined set of rules using the data mining rule such as WEKA using conjunction, JRIP, NNge, OneR, part rules. It observed that the rule created with WEKA was not optimal as indicated with classification accuracy. We synthesized prominent rules by eliminating important ones. Swarm intelligence (SI) is a technique where the rules are discovered through the study of collective behaviour in decentralized, self-organized systems such as ants. Ant-miner is a rule induction algorithm that uses SI techniques to form rules. Using our customized rule synthesis parse, Later these synthesized rules where optimal using Ant-Miner. Ant-miner results in better synthesized WEKA generated rules.

Index Terms—Ant Colony Optimization, Ant-miner Algorithm, Intrusion Detection.

1 Introduction

SEVERAL approaches proposed for computer systems protection from the unauthorized uses. Intrusion detection system is established for security infra-structure. IDS are able to identify the malicious behaviour in the protected network or computer. An intrusion detection system (IDS) always monitors the entire network and check for the malicious activities. It reports to the management station. A host-based intrusion detection system (HIDS) monitors and analyses the internal of a computer network .In computer security, a Network Intrusion Detection System (NIDS) attempts to discover unauthorized access to a computer network by analysing traffic on the network for signs of malicious activity. Swarm intelligence is one of the techniques in bio inspired family. Evolutionary algorithms are proposed on the basis of natural biological evolution and the social behaviour of species. Swarm intelligence is used to detect the abnormal behaviour as well as the normal behaviour of a network.

In this paper we considered the KDD dataset [20] for the rule mining. We considered 5 types of attacks such as normal, smurf, satan, back, Neptune from the KDD dataset. We mined the rules using the data mining tool such as WEKA [7]. After the generation of each rules for the attacks, we determined the accuracy, which was found to be very less. So we generate rule synthesis parser which eliminated redundant rules generated by classification also implemented in WEKA. Subsequently found that our rule pruning method gave better accuracy in comparison to WEKA. Ant Colony Optimization (ACO) is a method for solving a complex problem. Ant-Miner algorithm is a typical type of a ACO. So the rules are generated by Ant-Miner. After the rule pruning we obtained that Ant-Miner results in better accuracy compared with the rules which are not synthesized. This customized rule parser gives the optimal result.

The report organized as follows. Section 2 is the related work of various authors. Section 3 shows the proposed methodology. Section 4 gives the experiment result. Section 5 follows the conclusion

- *NimmyCleetusis currently pursuing masters degree program in computer science and en engineering with information system in MG University, India, PH-8129614792. E-mail: cnimmy2008@mail.com*
- *Dhanya K.A is assistant professor of scms school of engineering and technolog , India, PH.9495026093 E-mail:dhannyashibu@gmail.com*

2 Related works

In [16] author incorporates the evolutionary algorithm which is used to find the near optimal solution. The result of the study shows that the PSO is more accurate than the other evolutionary algorithm. PSO always find the solution in the problem space and update its position and velocity.

In [9] author reviewed the PSO techniques and its varied applications. PSO was found to be solving the complex problems, with potential for hybridization and specialization, and demonstration of some interesting emergent behaviour.

In [2] author proposed the swarm intelligence in intrusion detection. An IDS uses the aspect of ACO and PSO for the detection. The result of the study shows the comparison of several SI based IDS approaches. This paper has offered three basic approaches which classify the attacks. This paper evaluates the accuracy of KDD dataset to classify the attacks.

In [5] author proposed fractional particle swarm optimization in multi-dimensional problem space. This paper proposes two techniques, which successfully gives several major problems in the field of Particle Swarm Optimization (PSO) and solve the complex, multimodal optimization problems at high dimensions.

In [3] author was proposed multi-dimensional particle swarm optimization in dynamic environments. In order to solve the premature convergence problem and to increase the efficiency developed FGBF techniques. This paper introduced a new method for solving the complex problem which is moving peak benchmark (MPB) which simulates for a fixed dimension.

In [14] author was proposed rule induction using ant colony optimization for mixed variable attributes. Rule induction is the set of rules that characterize the data. In Swarm intelligence (SI) rules are discovered through the combined behaviour of decentralized, self-organized systems of ants. The objective of this paper is to compare study of ant-miner algorithm and improved continuous ant-miner algorithm.

IJSER

3 Proposed Methodolgy

As per the review of each paper, we implemented ACO. Ant-Miner is an ACO algorithm for rule discovery in database. The heuristic value used in Ant-Miner is based on the concept of entropy. The flow chart for the comparative study of Ant-Miner algorithm is given below. Knowledge Discovery in Database (KDD) is the process of extracting models and patterns in a large database. Rule Discovery is an important task since it generates set of rules that describes the class category. However, these rules are simple and comprehensive. Evolutionary algorithms are widely used for rule discovery.

We took 16,500 data value from the KDD dataset. Dataset includes five classes of attacks. In Training set 4999 Normal attacks, 5000 Neptune attack, 501 Satan attack, 5000 Smurf attack, 999 back attacks are included. Our aim is to create rules for classifying the attacks. For classifying the attacks, we use Ant-Miner Algorithm. First we created rules using WEKA. After that we synthesized the rules by removing repetition rules by custom developed rule mining parser which removes repeated rules obtained with algorithm in WEKA. This is later pruned using Ant-Miner.

We observed redundant rules obtained by classification also implemented in WEKA. Hence we performed rule synthesis where rejected rules are eliminated and smaller rule set is obtained. Later classification model using pruned rules are created.

IJSER

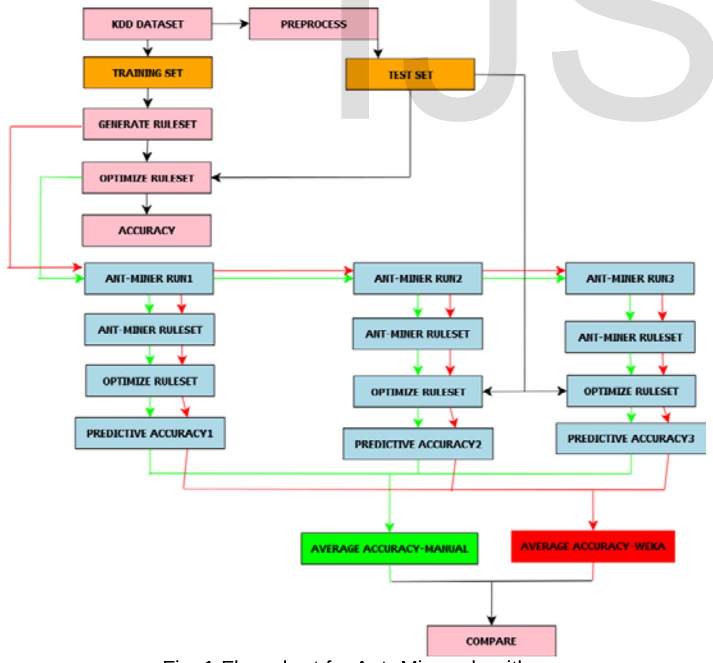


Fig. 1.Flow chart for Ant- Miner algorithm

The goal of the Ant-Miner algorithm is to classify the classes of attacks. The algorithm for the Ant-Miner is given below

```
Step 1   : Training Set=all training examples;
  Step 2   : WHILE (No. of uncovered example in
the Training set>Max covered
                Sample
Step 2.1  : REPEAT
Step 2.1.1:  i=i+1;
Step2.1.2:  Ant (i) constructs a
            Classification rule;
Step 2.1.3:  Prune the constructed rule;
Step 2.1.4:  Update the pheromone trails
of Ant;
   Step 2.2  :  UNTIL (i>=No of Ants) or
               (Ant (i) constructed the same rule
   MaxRulesConverge times);
                  Select the best rule from all
                 Constructed rules;
              Remove the rules that is covered by
               The selected rule from the training set;
                 END WHILE
```

Here, we are analyzing the accuracy of the rules generated by WEKA as well as the rule parser generated by Ant Miner. The heuristic value used in Ant-Miner is based on the concept of entropy. This concept is from the heuristic value of the rule. Hence we can use the concept of pheromone concentration for discovering the rules discussed in subsequent subsection.

3.1 Pheromone initialization

The pheromone values are equally distributed in the table, which is given by,

$$\tau_{ij}(t=0) = \frac{1}{\sum_{i=1}^{a} b_i} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (1)$$

- a is the total no of attributes.
- b_i is the number of values in the domain of attribute i

3.2 Rule Construction

Ant miner rule contain the two parts such as conditional and predicated class. The conditional part is a combination of attribute-operator-value-tuple pairs. Let us consider the rule condition $term_{ij} \approx A_i = V_{ij}$, where A_i is thei^{th} attribute and V_{ij} is the j^{th} value in domain of A_i. The probability that this condition will be added to the current partial rule is given by,

$$P_{ij}(t) = \frac{\tau_{ij}(t) * \eta_{ij}}{\sum_{i}^{a} \sum_{j}^{b_i} \tau_{ij}(t) * \eta_{ij}, \forall i \in I} \quad \ldots..(2)$$

- η_{ij} is a heuristic value for term$_{ij}$
- $\tau_{ij}(t)$ is the amount of pheromone currently available in attribute i and value j.
- a is the total no of attributes.
- b_i is the total number of values in the domain of attribute i.
- $I$is the set of attributes that are not yet used in ant.

3.3 Heuristic value

For analyzing the quality of a rule we measure the information theoretic value known as the heuristic value. The equation is given by,

$$\eta_{ij} = \frac{\log_2(k) - InfoT_{ij}}{\sum_i^a \sum_j^{b_i} \log_2(k) - InfoT_{ij}} \text{..........(3)}$$

$$InfoT_{ij} = \left[\frac{freqT_{ij}^w}{|T_{ij}|}\right] * \log_2\left[\frac{freqT_{ij}^w}{|T_{ij}|}\right] \text{....(4)}$$

- K is the number of classes
- $freqT_{ij}^w$ is the number of cases in partition T_{ij} with class w.
- $|T_{ij}|$ is the total number of cases in partition T_{ij}
- a is the total number of attributes.
- b_i is the total number of values in the domain of attribute i

3.4 Rule Pruning

Rule pruning is used for increasing the accuracy and the comprehensibility of the rules. After the pruning step, according to the majority class in the cases covered the rule may be assigned a different predicted class. The rule pruning removes the term which its removal of redundant rules. The following equation is used for measuring the quality of a rule,

$$Q = \left(\frac{TP}{TP + FN}\right) * \left(\frac{TN}{TN + FN}\right) \quad \text{............ (6)}$$

- TP is number of cases that are covered by the rule and having the same class as that being predicated by the rule.
- FP is the number of cases covered by the rule and having different class from being predicated by the rule.
- FN is the number of cases that are not covered by the rule while having the class predicated by the rule.
- TN is the number of cases that are not covered by the rule and having a different class from the class predicated by the rule.

3.5 Pheromone update rule

After each ant constructs the rule, the pheromone update is given by,

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \tau_{ij}(t).Q \ \forall i, j \text{ ……….. (7)}$$

To simulate pheromone evaporation in real ant colony systems, the amount of pheromone present in each $term_{ij}$ that does not occur in the constructed rule has to be decreased. The pheromone reduction of an unused term is done by dividing the value of each τ_{ij} by the sum of all τ_{ij}.

3.6 Evaluation Metrices

Various evaluation metrics like accuracy, precision, recall, F-measure etc can be found out from the classification results of WEKA.

$$Accuracy(Acc) = \frac{TP + TN}{TP + TN + FP + FN} \text{...(8)}$$

$$True\ Positive\ Rate(TPR) = \frac{TP}{TP + FN} \quad \text{---(9)}$$

$$False\ Negative\ Rate(FNR) = \frac{FN}{TP + FN} \text{...(10)}$$

$$False\ Positive\ Rate(FPR) = \frac{FP}{TN + FP} \text{....(11)}$$

$$True\ Negative\ Rate(TNR) = \frac{TN}{TN + FP} \text{...(12)}$$

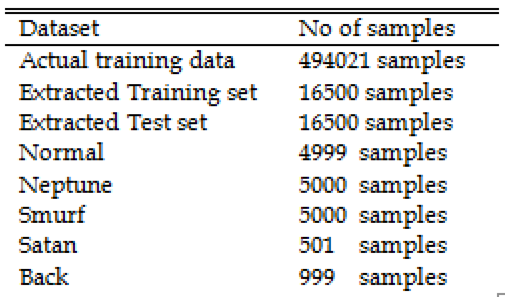
4 Experiments and Results

4.1 Experiment Setup

The goal of the Ant-Miner algorithm is to extract the rules from the database. We implemented Ant-miner algorithm and analyze the rule which is generated by WEKA tool and our custom generated parser table below shows details of KDD-dataset which is taken.

TABLE 1

INITIAL DATA IN THE DATASET



Dataset	No of samples
Actual training data	494021 samples
Extracted Training set	16500 samples
Extracted Test set	16500 samples
Normal	4999 samples
Neptune	5000 samples
Smurf	5000 samples
Satan	501 samples
Back	999 samples

We generated dataset with following data. In general KDD-dataset is classifying 24 types of attacks. But in our case we

IJSER

classify 5 types of attacks. First we generated the rules by WEKA with the training set. After the rule parser the set of rules which improves the accuracy in comparison to WEKA.

4.2 Result

The table below shows the percentage of true positive, true negative, false positive, false negative and the accuracy of each attack.

TABLE 2
ACCURACY VALUE FOR MANUALLY GENERATED RULES

	Normal	Neptune	Satan	Smurf	Back
TPR	99.94	99.64	99.40	100.00	98.00
TNR	99.70	99.97	100.00	100.00	100.00
FPR	0.30	0.03	0.00	0.00	0.00
FNR	0.06	0.36	0.60	0.00	1.40
Accuracy rate	99.70	99.80	99.99	100.00	99.99
Avg-Acc rate	99.99				

TABLE 3
ACCURACY FOR THE WEKA RULES ITERATION 1, ITERATION 2

Iteration1(WEKA)					
	Normal	Neptune	Satan	Smurf	Back
TPR	99.84	0.00	0.10	0.00	0.00
TNR	99.99	0.00	0.16	0.00	0.00
FPR	0.01	100.00	99.84	100.00	100.00
FNR	0.16	100.00	99.90	100.00	100.00
Accuracy rate	30.20	49.90	83.20	49.90	90.82
Quality rate	0.0008	0.00	0.002	0.00	0.00
Avg-Acc rate	60.80				
Avg-Quality rate	0.021				

Iteration1(After rule synthesis)					
	Normal	Neptune	Satan	Smurf	Back
TPR	80.84	98.78	89.22	100.0	15.12
TNR	4.60	0.01	9.08	0.00	0.00
FPR	95.40	99.99	90.2	100.0	100.0
FNR	19.16	1.22	10.78	0.00	84.88
Accuracy rate	90.82	99.52	90.87	100.0	94.75
Quality rate	77.12	98.76	81.16	100.0	15.15
Avg-Acc rate	95.14				
Avg-Quality rate	74.42				

After the implementation of the rules in the dataset, we get the accuracy as above. Now we have to implement the Ant-Miner algorithm in two set of rules. First one is rules generated by WEKA which is not modified and the last one is rules generated by WEKA which is manually modified. Finally we find the accuracy and compare with the two set of rules. The given table shows the percentage of accuracy and quality in three iterations.

Now the iteration is stopped when it gives the lesser accuracy compared with the previous iterations.

TABLE 4
ACCURACY FOR WEKA RULES IN ITERATION3

Iteration3(WEKA)					
	Normal	Neptune	Satan	Smurf	Back
TPR	99.84	0.00	0.10	0.00	0.00
TNR	99.99	0.00	0.16	0.00	0.00
FPR	0.01	100.00	99.84	100.00	100.00
FNR	0.16	100.00	99.90	100.00	100.00
Accuracy rate	30.20	49.90	83.20	49.90	90.82
Quality rate	0.0008	0.00	0.0002	0.00	0.00
Avg-Acc Rate	60.80				
Avg-Quality rate	0.021				

IJSER

Now we can see the accuracy for the rule generated by the WEKA which is modified. In the previous table shows that there are so many misclassifications due to overwrite of rules, so that the accuracy and the quality is less compared to the manual rule that we generated.

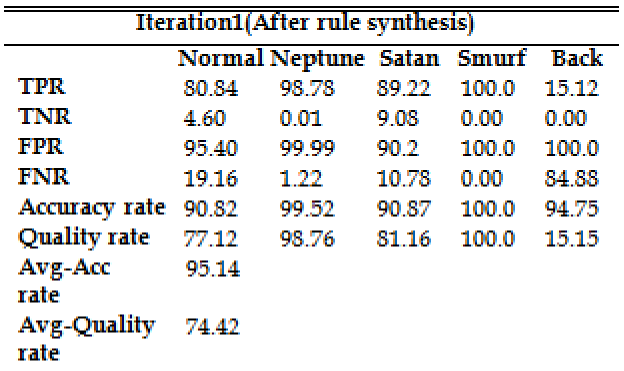
After the iteration 1 we discovered the accuracy of the rule using ACO. The step is the rule pruning which is carried out by some set of rules. The selected attribute should change to zero for calculating the entropy as well as the probability. Then the rule pruning can be carried by taking this probability and for finding the accuracy. The iteration is stopped until degrades or same as the previous iteration

Hence from the analysis we can see that in second iterations the quality is higher compared to the remaining iterations. Among the three iterations the second iteration generated the best rule for classification.

We know that the rules are generated by some certain type of algorithm such as Conjunction rule, JRIP, NNge, OneR, part. For example,
If flag=SF and count >150 then label=Satan

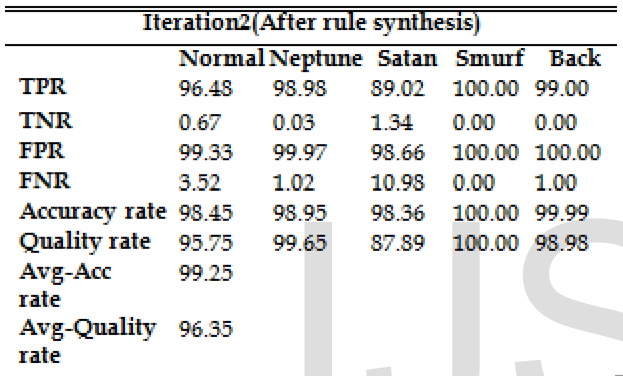
Likewise we generate some rules using the above algorithms. Rule parser always removes the redundant rules to improve the accuracy. Ant Miner further optimizes the rules generated by custom parser, indicated by higher accuracy. This is due to the fact that at each iteration informative rules are removed

TABLE 5
ACCURACY FOR THE SYNTHESISED RULES USING ANT-MINER



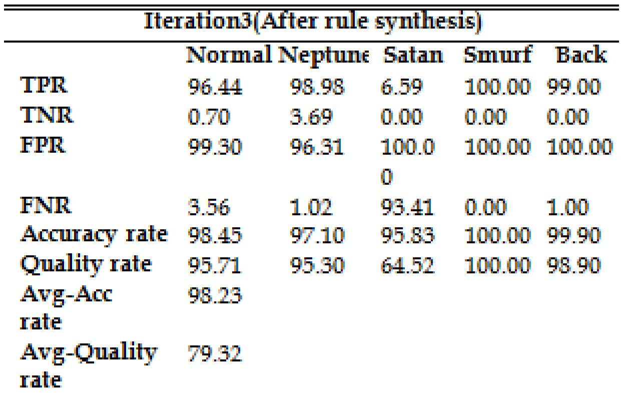
Iteration1(After rule synthesis)					
	Normal	Neptune	Satan	Smurf	Back
TPR	80.84	98.78	89.22	100.0	15.12
TNR	4.60	0.01	9.08	0.00	0.00
FPR	95.40	99.99	90.2	100.0	100.0
FNR	19.16	1.22	10.78	0.00	84.88
Accuracy rate	90.82	99.52	90.87	100.0	94.75
Quality rate	77.12	98.76	81.16	100.0	15.15
Avg-Acc rate	95.14				
Avg-Quality rate	74.42				

TABLE 6
ACCURACY FOR THE ITERATION 2 IN ANT-MINER



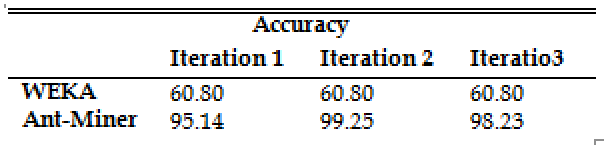
Iteration2(After rule synthesis)					
	Normal	Neptune	Satan	Smurf	Back
TPR	96.48	98.98	89.02	100.00	99.00
TNR	0.67	0.03	1.34	0.00	0.00
FPR	99.33	99.97	98.66	100.00	100.00
FNR	3.52	1.02	10.98	0.00	1.00
Accuracy rate	98.45	98.95	98.36	100.00	99.99
Quality rate	95.75	99.65	87.89	100.00	98.98
Avg-Acc rate	99.25				
Avg-Quality rate	96.35				

TABLE 7
ACCURACY FOR THE ITERATION 3 IN ANT-MINER



Iteration3(After rule synthesis)					
	Normal	Neptune	Satan	Smurf	Back
TPR	96.44	98.98	6.59	100.00	99.00
TNR	0.70	3.69	0.00	0.00	0.00
FPR	99.30	96.31	100.00	100.00	100.00
FNR	3.56	1.02	93.41	0.00	1.00
Accuracy rate	98.45	97.10	95.83	100.00	99.90
Quality rate	95.71	95.30	64.52	100.00	98.90
Avg-Acc rate	98.23				
Avg-Quality rate	79.32				

TABLE 8
COMPARISON OF ACCURACY



	Accuracy		
	Iteration 1	Iteration 2	Iteratio3
WEKA	60.80	60.80	60.80
Ant-Miner	95.14	99.25	98.23

The above table gives the accuracy for different iterations. It gives the comparative study for both WEKA and rules which are synthesized for optimal result. From this we understood that the synthesized rules having higher accuracy compared with WEKA rules.

5 INFERENCE

Following are the inference of our study

(1). The rules generated with WEKA (Conjunction rule, JRIP, NNge, OneR, part) are redundant and is unable to identify attacks.
(2). Custom developed rule parser eliminates redundant and repetitive rules there by increasing accuracy.
(3). Ant Miner further optimizes the rules generated by custom parser, indicated by higher accuracy. This is due to the fact that every iterations informative rule is removed.

6 CONCLUSION

There are different ways to classify the attacks in the database. Ant-miner is the one of the method which is mainly used for rule construction, which is computationally less expensive. These methods are reliable and robust. In this paper we generated the set of rules using Ant-Miner and calculated the quality of the rule. After each iteration, the result is checked with the previous iterations. Finally we analyze the result with both WEKA generated rules and the manual generated rules. Hence we obtained that manual rule is having higher quality.

IJSER

REFERENCES

[1] AtulGarg, Dimple Juneja, "A Comparison and Analysis of Various Extended Techniques of Query Optimization", International Journal of Advancements in Technology, 2012.

[2] C.Kolias, G.Kambourakis, M.Maragoudakis "Swarm Intelligence in Intrusion Detection", ElSEVIER 2011.

[3] Serkankiranyaz, JenniPulkkinen, MoncefGabbouj, "Multidimensional Particle Swarm Optimization in Dynamic Environments", ELSEVIER, 2011

[4] Ammar W Mohemmed,Mengjie Zhang, Will Browne, "Particle Swarm Optimization for Outlier Detection", 2010.

[5] SerkanKiranyaz, TurkerInce,AlperYildirim, MoncefGabbouj, "Fractional Particle Swarm Optimization in Multi-dimensional Search Space", IEEE transactions on Systemsand Cybernetics, 2010.

[6] Li Y, Yang G, Xu J, Zhao B., "Anomaly Detection for Clustering Algorithm based on Particle Swarm Optimization", Journal of Jiangsu University of Science and Technology(Natural Science Edition); 2009.

[7] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten, "The WEKA Data Mining Software: An Update", SIGKDD Explorations, Volume 11, Issue 1, 2009

[8] MahbodTavalaee, EbrahimBagheri, Wei Lu & Ali A. Ghorbani, "A detailed analysis of KDD cup99 dataset", 2009

[9] Alec Banks, Jonathan Vincent, ChukwudiAnyakoha, "A Review of Particle Swarm Optimization. Part II: Hybridization, Combinatorial, Multicriteria and Constrained Optimization, and Indicative Applications", 2008

[10] Chen, X., & Li, Y, 2007, "A Modified PSO Structure Resulting in High Exploration Ability with Convergence Guaranteed", IEEE Transactions on Systems, Man, and Cybernetics, Part B, 37(5), 1271–1289

[11] A. Abraham, S. Das, and S. Roy, "Swarm Intelligence IV. Algorithms for Data Clustering", in Soft Computing for Knowledge Discovery and Data Mining Book. New York: Springer-Verlag, Oct. 25, 2007, pp. 279–313.

[12] Bo Liu, Hussien A. Abbas, Bob Mckay, "Density based Heuristic for Rule Discovery with AntMiner", 2006

[13] Tsang W, Kwong S, “Ant Colony Clustering and Feature Extraction for Anomaly Intrusion Detection”, Swarm Intelligence in Data Mining; 2006:101e21

[14] SanthoshSwaminathan, “Rule Induction using Ant Colony Optimization for Mixed Variable Attributes”, 2006

[15] A. P. Engelbrecht, “Fundamentals of Computational Swarm Intelligence”, Hoboken, NJ: Wiley, 2005.

[16] EmadElbeltagi, TarekHegazy, Donald Grierson, ”Comparison among Five Evolutionary based Optimization Algorithms”, 2005

[17] Yang S, Wang M, Licheng J., “A Quantum Particle Swarm Optimization”, In: Proceedings of the Congress on Evolutionary Computation 2004 (CEC2004).p. 320e324.

[18] W.-J. Zhang and X.-F. Xie, “DEPSO: Hybrid Particle Swarm with Differential Evolution Operator”, in Proc IEEE Int. Conf. Syst., Man, Cybern, 2003, vol. 4, pp. 3816–3821.

[19] Lovberg, M., & Krink, T, 2002. “Extending Particle Swarm Optimizers with Self-organized Criticality”, Proceedings of the IEEE Congress on Evolutionary Computation, 2, 1588–1593.

[20] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

IJSER