International Journal of Computer and Information Technology (ISSN: 2279 – 0764)
Volume 03 – Issue 01, January 2014

Learning Time-based Rules for Prediction of Alarms from Telecom Alarm Data Using Ant Colony Optimization

Imran Khan Joshua Z. Huang Nguyen Thanh Tung
Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences
Shenzhen 518055, China
Imran.khan@siat.ac.cn

***Abstract*—This paper proposes a new method to learn time based rules from telecom system alarm data for prediction of the classes of alarms. A time based rule associates an alarm class with the *StartTime* attribute and other attributes of alarms. The rules are evaluated with the coverage of the rules in the training data set. Given a new alarm generated at a particular time, its alarm class can be predicted with a set of time based rules. We present a new algorithm that extracts time based rules from alarm data through an ant colony optimization (ACO) process. Given an alarm training data, a search space is formulated as a square matrix indexed by distinctive attribute values. The pheromone at the search space is computed from the training data and a time based rule is discovered from the pheromone distribution. The pheromone distribution is updated after a time based rule is extracted and the search for a new rule starts. A rule pruning process is used to remove redundant rules and increase the prediction accuracy of the final rule set. We experimented the new method on Nokia Simmons (NSN) and Ericsson data sets and compared the results of the new method and the TimeSeluth system. The comparison demonstrated that the new method outperformed TimeSeluth in prediction accuracy.**

Keywords: ant colony optimization, time based rules, rules discovery, prediction

I. Introduction

Real time handling of alarm events is a critical function in large telecommunication network systems. Alarm events are signals of system faults that can be caused by malfunctioning of hardware or software or mal operations of operators or users. Alarm data is a major source for fault diagnosis and recovery. Therefore, handling of alarm data has significant impact on operation costs and quality of services in telecommunication industry.

An alarm event is described with a set of attributes, usually including start time, alarm code, alarm severity and alarm class. In a large scale complex telecommunication network, a system fault can cause a large number of alarm events generated. A particular alarm can also generate a set of cascaded alarms. Alarm events are recorded in system logs. By analyzing alarm logs, the root causes of system faults can be found and the problems can be fixed. Since the system log data is huge, quickly identifying critical alarms and tracing the root cause of system faults becomes a big challenge in improving the robustness of telecommunication network systems and quality of customer services. Data mining is a useful tool for analyzing alarm data.

Fault identification and prediction are two important research topics in telecom alarm data mining. Fault identification involves detection of critical alarms from log data and discovery of the root cause for a system fault. Several useful techniques for fault identification have been proposed [3], [11], [6], [2], [22], [16], [13], [1], [21], [4], [23], [10], [14], [7], [8]. Current concerns on these techniques are scalability because it is very difficult to extract critical alarms from huge log data.

Fault prediction is to use critical alarm training data to build prediction models to predict alarm classes that are the indicators of causes of system faults. Sequential data mining methods are often used to build the prediction models. In sequential

data mining, the input data is a sequence of alarm transactions ordered by alarm *StartTime*. However, sequential data mining is not scalable to huge alarm transactions. Genetic algorithm is also used in building prediction models but it also has the scalability problem in dealing with large data because it takes too much time to find a global optimal model. Decision tree algorithms are scalable to large alarm data but suffer low accuracy prediction problems.

In this paper, we propose a new method to build prediction models using a set of time based rules learnt from alarm data for prediction of alarm classes. A time based rule associates an alarm class with the *StartTime* attribute and other attributes of alarms. The rules are evaluated with the coverage of the rules in the training data set. Given a new alarm generated at a particular time, its alarm class can be predicted with a set of time based rules. We present a new algorithm that extracts time based rules from alarm data through an ant colony optimization (ACO) process. Given an alarm training data, a search space is formulated as a square matrix indexed by distinctive attribute values. The pheromone at the search space is computed from the training data and a time based rule is discovered from the pheromone distribution. The pheromone distribution is updated after a time based rule is extracted and the search for a new rule starts. We also define a rule pruning process to remove redundant rules and increase the prediction accuracy of the final rule set. This ACO based method avoids generation of all possible rules and rules for each class can be generated in a parallel way.

We have conducted a series of experiments on two real alarm data sets NSN and Ericsson. In these experiments, 10-fold cross validation was used to evaluate the performance of the proposed algorithm. We achieved the average accuracy of 94.2% and 94.0% from NSN and Ericsson data sets respectively. These results were significantly higher than the accuracies of the same data sets produced by the TimeSeluth system.

The rest of this paper is organized as follows. We briefly review some related work in Section 2. We present our new method in detail in Section 3. Experiment results are discussed in Section 4. The paper is concluded in Section 5.

II. Related work

TimeSleuth is a software used in Pakistan Telecommunication Limited (PTCL) to predict network faults from alarm data [15]. TimeSleuth uses causal rules as prediction models. It includes a C4.5 decision tree algorithm to discover causal rules from alarm training data and a C4.5T (Temporal) algorithm to indicate time information in each generated rule. For a tree construction, the values of each attribute are split into two or more equal parts. The information gain of each split is computed. The attribute with the highest information gain split is treated as the root of a tree. The edges from each node in a tree represent the values of the attribute. The same split process is repeated for descendent node selection in tree construction. The values of the target attribute appear at the levels of the tree.

Timeweaver[24] is another software that uses a genetic algorithm (GA) to identify the rules from telecom alarm data for prediction of hardware faults. The data pre-processing step removes all redundant alarms from a data set. Each alarm of a data set is considered as a sequence and an ordered sequence mining method is used to extract rules. The set of rules is used to predict a target event. There are two steps in model building. In step one, a genetic algorithm is used to optimize the search for a set of best rules for prediction. In step two, rules are sorted in the order from the best to the worst of accuracy of prediction. Redundant rules are removed from the worst part of rules.

The mining sequential alarm rules (MSAP) algorithm was proposed to discover sequential alarm rules from alarm sequences[25]. In MSAP, all rules are extracted using the information of time difference of two alarm events. The most commonly used sequential mining algorithms are Relim [5], Eclat [17], FP-Growth [9] and Apriori [19]. In MSAP, Apriori algorithm was used for candidate generation.

PrefixSpan [18] is another frequent sequential rule mining method that uses a divide-and-conquer strategy and rule growth method to extract rules from telecom alarm data. The sequential rule mining algorithm GSP [20] can also be used. PrefixSpan has a difficulty to differentiate actual associated data items from the data items which have frequent co-occurrence. The extracted rules by PrefixSpan are

not always robust and meaningful. The time complexity of MSAP is large because it requires to scan the data set many times for generation of candidate sets.

III. The ACO based algorithm

In this section, we present an Ant Colony Optimization (ACO) method to discover a set of time based rules from alarm data for prediction of alarm classes. The input data for discovery of time based rules is first described and a time based rule is defined with respect to the input data. To extract time based rules from the input data, we formulate a square matrix as the search space and define the formulae to compute the pheromone distribution in the search space from input data. We present the process to find time based rules in the search space and the methods to update the pheromone distribution of the search space. Finally, we sketch an ACO based algorithm in a flow chart to detail the process to discover time based rules from alarm data.

A. Input alarm data

An input alarm data set consists of a set of alarm records. Each record represents a particular alarm that is described by a set of attribute values. Table I is an example of a training alarm data set that has five attributes. The *AlarmType* attribute represents the classes of generated alarms which is important for building alarm prediction models. An alarm class is an indication on what causes an alarm. For instance, alarms can indicate loss of signals or battery failure. Alarm prediction is to predict the class of the alarm whenever an alarm occurs so as to diagnose the cause of the alarm. Therefore, the *AlarmType* attribute is considered as the target attribute in building prediction models.

The *StartTime* attribute represents the start time when an alarm was generated. In Table I, hourly intervals are used to represent alarm start times. In reality, small time intervals are used. The *AlarmSeverity* attribute indicates the potential risk of an alarm. An alarm with critical alarm severity requires immediate actions to be taken to handle the fault. The *AlarmOrigin* attribute represents the source where the alarm was generated. An origin in a telecom network can be associated with different types of resources. The *AlarmCodes* attribute describes identification of different alarms.

TABLE I
An alarm data set.

StartTime	AlarmSeverity	AlarmOrigin	AlarmCode	AlarmType
8:00	MAJOR	OSS	RXETRX/1AMAP/10	LossOfSignal
12:00	MAJOR	OSS	RXETRX/1AMAP/10	LossOfSignal
1:00	MAJOR	OSS	RXETRX/1AMAP/10	LossOfSignal
8:00	MAJOR	OSS	RXETRX/1AMAP/10	LossOfSignal
12:00	MAJOR	OSS	RXETRX/1AMAP/10	LossOfSignal
12:00	MINOR	OSS	RXETRX/1AMAP/10	BatteryFailure
12:00	MAJOR	OSS	RXETRX/1AMAP/20	Processing
1:00	MAJOR	OMCR	RXETRX/1AMAP/20	Equipment
1:00	MAJOR	OSS	RXETRX/1AMAP/20	Environmental
8:00	MINOR	OSS	RXETRX/1AMAP/10	Processing

B. Time Based Rule

A time based rule is an association between *StartTime* attribute and *AlarmType* attribute. A rule is represented as an *IF* (...); *THEN* (...). The *IF* antecedent part always contains one pair of $StartTime = value$ and the *THEN* consequent part always contains one pair of $AlarmType = value$. The antecedent part is allowed to have other pairs of $Attribute = value$ connected with AND operator but each rule can only have at most one pair for each attribute. The rule below is an example of time based rules.

$$IF\ StartTime = 12:00\ \ AND\ \ AlarmOrigin = OSS$$
$$AND AlarmCode = \text{RXETRX/diagup1AMAP/diagup20}$$
$$THEN\ \ AlarmType = LossOfSignal \quad (1)$$

Every row in Table I can be written as a time based rule. However, a rule from a row in the table does not have the power to be used to accurately predict the alarm class of an alarm. A rule must be evaluated with the training alarm data set. The coverage of a rule is defined as the total number of alarm records in the training data, which satisfy the antecedent conditions of the rule. The higher the coverage, the more important the rule.

For a large training alarm data set, it is very time consuming to evaluate the coverage of all possible rules. We present an ACO based algorithm to extract a set of important rules from a training data set without exhausted evaluation of all possible rules.

C. The search space

Let X be a table of alarm training data with m attributes A_1, A_2, $\cdots$, A_m and the target attribute C for classes, where all attributes A_j (1≤ j ≤ m) are categorical, i.e., containing only finite categorical values. Let V_j be the set of distinctive values of attribute A_j for $1 \leq j \leq m$ and $V = V_1 \cup V_2 \cup \cdots \cup V_m$ is the union of distinctive values of all attributes.

Definition 1: The search space for X is a square matrix $T_{N \times N}$ indexed with distinctive values of V, where N is the total number of values in V.

Fig.(1) shows the search space for Table 1. The rows and columns of the matrix are indexed by the same set of 9 categorical values {8:00, 12:00, 1:00, MAJOR, MINOR, OSSS, OMCR, RX10, RX20} where the last two elements use abbreviations of attribute values for *AlarmCode*.

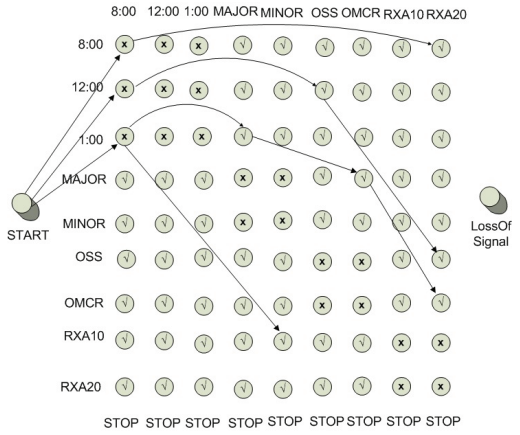

Fig. 1. The search space of Table 1.

Assume the search matrix represents a search space where each point, identified by a combination of a pair of row and column indices, contains some amount of pheromone that ants can follow from one place to a destination. Using the training alarm data set, the initial pheromone of the search space can be computed as follows.

$$\tau_{xy}\,(i=0) \;=\; \frac{1}{N} \tag{2}$$

where x and y are row and column indices, i is the number of iterations starting from 0, and N is the total number of distinctive values of all attributes in the training data excluding the target attribute. The initial pheromone in all places are the same.

Next, we take one class C_z of the target attribute and calculate the correlation between the class and each location in the search space, i.e., the correlation between the class and the combinations of two categorical values. The correlation is computed as follows.

$$\eta_{xy} = \frac{|t_x, t_y, C_z|}{|t_x, C_z|} * \frac{|t_y, C_z|}{|t_y|} \tag{3}$$

where x and y are row and column indices, t_x and t_y are two categorical values of indices x and y, C_z is a class. $|t_x, t_y, C_z|$ is the number of alarm records in the training data that contain values of t_x, t_y and C_z, i.e., the support of $\{t_x,\, t_y,\, C_z\ \}$, $|t_x, C_z|$ is the support of $\{\ t_x, C_z\}$, $|t_y, C_z|$ is the support of $\{t_y, C_z\}$, and $|t_y|$ is the support of $\{t_y\}$.

Given the initial pheromone and the correlation, we can compute the probability of each location that an ant can choose to pass. The probability is calculated as follows.

$$P_{xy}(i) = \frac{\tau_{xy}^{r1} \eta_{xy}^{r2}}{\sum_{y=1}^{N} \tau_{xy}^{r1} \eta_{xy}^{r2} \sum_{x=1}^{N} n_x} \tag{4}$$

where x and y are row and column indices, i is the iteration number, $\tau_{xy}(i)$ is the pheromone at location $(x,\ y)$ in iteration i, $\eta_{xy}(i)$ is the corresponding correlation, N is the total number of distinct values of all attributes, and $r1$ and $r2$ are two parameters. Here, n_x is a binary variable of {0, 1}. In the rule building process, if an attribute is not considered, $n_x = 1$, otherwise $n_x = 0$.

Given the search space filled with probability values at all locations, we will be able to search for time based rules using the ant colony optimization process. We consider that ants always start from a time value of the *StartTime* attribute and choose a class as the destination. Ants go through the search space from one location of an attribute to one location of another attribute and finally arrive the destination. A pathway of an ant forms a time based rule. Fig.(1) illustrates the start points, some pathways and the final destination of class *LossOfSignal*. The next section presents the process for searching a time based rule and updating the pheromone of the search space.

D. Search for a time based rule

Building a time based rule starts from the consequent part. We first select the most frequent class from the target attribute and put it in the *THEN* consequent part of the rule. For instance, class *LossOfSignal* in Table I was put in the *THEN* part of rule (1).

Secondly, we start to build the *IF* antecedent part by randomly selecting a value from *StartTime* attribute as the first term in the *IF* part. For instance, value *12:00* was selected in rule (1). Then, we look into the search matrix and find the value of other attributes that has the highest joint probability with *12:00* of *StartTime* attribute. In this case, it was value *OSS* in *AlarmOrigin* attribute for rule (1). We add the pair of $AlarmOrigin = OSS$ as the second term to the *IF* part connecting it to the the first term with operator AND. After this, we look into the search matrix again to find the next value of other attributes which have not been searched and select the value with the highest joint probability with the value of the last attribute *AlarmOrigin*. In this case, value *RXETRX/1AMAP/20* from attribute *AlarmCode* was selected for rule (1). The new pair of $AlarmCode = RXETRX/1AMAP/20$ is added to the *IF* part connecting to the last term with AND. Since the next attribute is the target attribute and its class value *LossOfSignal* was already selected in the $THEN$ consequent part, a time based rule is found, see the example of rule (1).

In each iteration of the rule building process, we use a counter variable G_n to count the number of pairs of attributes and values selected in the antecedent part of the rule. If two pairs are selected, $Gn = 2$. Here, each attribute-value pair is equivalent to the number of items in an association rule. For each counter value, different rules are generated for a selected target value.

After a time based rule is built, it is evaluated with the support, confidence and coverage of the rule by the alarm training data. A selected rule must have a support and confidence greater than the given minimal support and confidence thresholds respectively. Any rule with a support or confidence smaller than the threshold is dropped. By using the same search space another rule is created by an ant for the same target value.

When a time based rule is selected, it is added to the rule set and the fitness of the rule is computed as follows.

$$\mathrm{F} = \frac{TP}{C} \tag{5}$$

where TP is the number of alarm records in the training data correctly classified by the rule and C is the number of alarm records satisfying the IF part, i.e., the support of the set of values.

With the fitness value, we update the pheromone of the search space for the locations passed by the ant in searching the rule as follows.

$$\tau_{xy}\,(i+1) \;=\; \tau_{xy}(i)(1-\gamma) + \tau_{xy}(i)(1 - \frac{1}{1+F}) \tag{6}$$

where x and y are row and column indices, i is the iteration number, $\tau_{xy}(i)$ is the pheromone of the current iteration, F is the fitness of the current rule and γ is the pheromone evaporation given by the user. We can see that the change of pheromone on the locations is proportional to the fitness of the current rule.

The locations of pheramone matrix are updated by rule (1). The locations are $\tau(12:00, OSS)$ and $\tau(OSS, RXA20)$. In the meantime, we also update the pheromone for rows *12:00* and *OSS* separately by dividing each pheromone by the sum of the pheromone value of that row.

After the pheromone of the search space is updated, we can start the process to build a new time based rule by randomly selecting a new start time from *StartTime* attribute. This process continues until the total coverage of generated rules of a class *LossOfSignal* is less then minimum coverage threshold. All possible generated rules of a class with the total coverage less than minimum coverage threshold are not considered final rules.

When the total coverage of generated rules of a class exceeds minimum coverage threshold, we choose a new class from the target attribute as the new consequent part of rules for this class. We construct a new search space for the new class and repeat the above process to build a new set of time based rules for this class. This process continues until all classes in the target attribute have exhausted.

At the end of rule generation, a rule pruning process is used to remove redundant rules from the

set of generated rules. In the pruning process, the accuracy of each rule is increased by removal and addition of an attribute value at the antecedent part. A test data set with unknown class labels is used to measure the performance of the final set of generated rules.

Fig.(2) shows the flow chart of the ACO based algorithm for building a set of time based rules as prediction models to predict alarm classes. The time complexity of this algorithm is O (t× a × n × m), where t is the number of classes in the target attribute, a is the number of iterations or ants, n is the number of alarm records in the training data and m is the number of attributes in the training data excluding the target attribute.

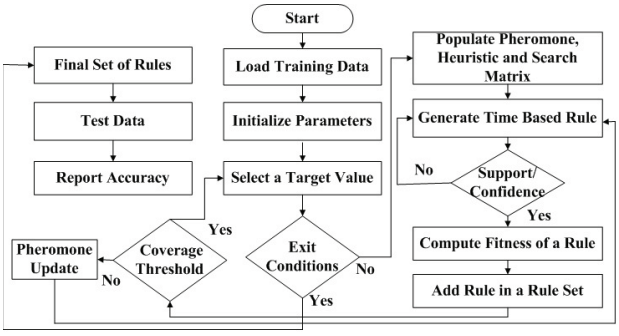

Fig. 2. The flow chart of the ACO based algorithm.

IV. Experiments

We have conducted a series of experiments to evaluate the ACO based algorithm for building time based rules for alarm prediction. In the experiments, we used two real world alarm data sets. The data sets were collected by the subcontractor (http://www.turnotech.com) who works for different telecom operators. The alarms were generated from Ericson GSM systems and Nokia Siemens GSM systems. The numeric attributes were transformed into categorical values in the data pre-processing stage using the discretization tools in Weka-3.4. Table II shows the characteristics of the two real world alarm data sets.

The ACO based algorithm was implemented in Matlab 9.0. The experiments were carried out on a machine with a 1.9GHZ processor and 4GB RAMS. We also used the java based software TimeSleuth to discover causal rules from the data sets. TimeSleuth uses a tree based algorithm for building the rules. TimeSleuth also signifies the causal relationship between the conditional attributes and the decision attribute [12].

TABLE II
Two real world alarm data sets.

datasets	Attributes	Instances	Classes
Ericsson Alarm	05	13112	06
NSN Alarm	05	12877	06

Table III shows the values for input parameters for the ACO based algorithm used in the experiments. r1 and r2 were used to balance the proportions of pheromone and correlation in computing the location probability. Minimum support and minimum confidence thresholds were used to select time based rules in the rule building process. Changing the minimum confidence and minimum support thresholds affected the size of the rule set and execution time of the algorithm.

TABLE III
Input parameters of the the ACO based algorithm.

Parameter	Value
Total Ants	500
Support Threshold	0.008
Confidence Threshold	0.5
Coverage Threshold	0.98
γ	0.09
r1	01
r2	01
Folds	10

We used 3 evaluation methods to evaluate the performance of the algorithms. Accuracy (A) is the proportion of the number of correct predictions over the number of alarm records in the data set. Precision (P) is the proportion of the true positive cases over the number of the predicted positive cases. Recall (R) is the proportion of the predicted true positive cases over the number of the true positive cases in the data set. These measures are calculated as follows.

$$\mathrm{A} = \frac{TP + TN}{Numberofrecordsindata} * 100 \qquad (7)$$

$$\mathrm{P} = \frac{TP}{TP + FP} * 100 \qquad (8)$$

$$\mathrm{R} = \frac{TP}{FP + FN} * 100 \tag{9}$$

where TN (True Negative) is the number of the true predicted negative cases and TP (True Positive) is the number of the true predicted positive cases. FN (False Negative) is the number of the false predicted negative cases and FP (False Positive) is the number of the false predicted positive cases.

$$\sigma = \sqrt{\frac{1}{N}\sum(x_i - \mu)^2} \tag{10}$$

10-fold cross validation was used in these experiments to evaluate the algorithm. In 10-fold cross validation, each data set was divided into ten subsets with equal numbers of alarm instances. One subset was used for testing and other nine subsets were grouped into the training data set. Ten results were obtained from each data set and the average result was used in comparison.

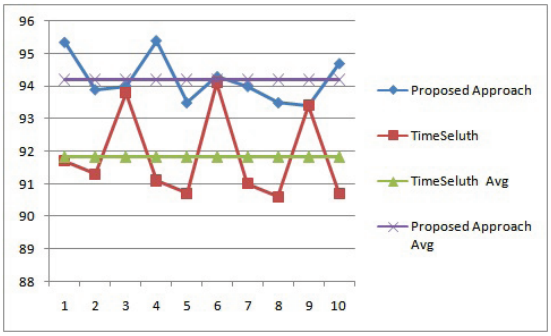

Fig. 3. NSN 10-fold accuracies.

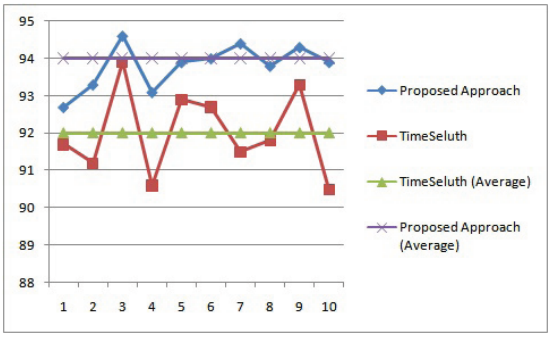

Fig. 4. Ericsson 10-fold accuracies.

Fig.(3) shows the results of 10 fold cross validations of NSN data by the ACO based algorithm and the TimeSeluth system. The light green line shows the average accuracy 91.8% of the TimeSeluth results and the purple line shows the average accuracy 94.2% of the results by the ACO based algorithm. We can see that the ACO based algorithm significantly outperformed TimeSeluth on this data set. The red line shows the accuracies of 10 fold cross validation results by TimeSeluth and the blue line shows the accuracies of 10 fold cross validation results by the ACO based algorithm. From these results , we can see that the performance of the ACO based algorithm was more stable than TimeSeluth. Similar results were obtained from the Ericsson data, as shown in Fig(IV).

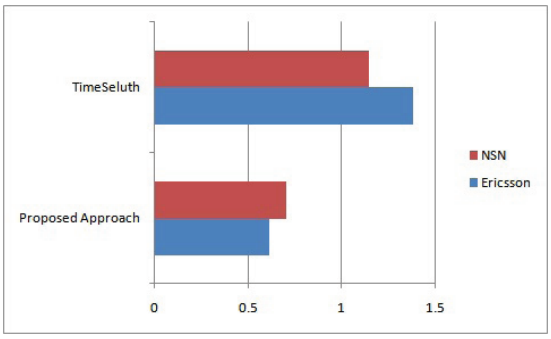

Fig. 5. Standard deviations of 10-fold cross validation accuracies.

Fig.(5) shows the standard deviation of accuracies of 10-fold cross validation results from Ericsson and NSN data sets. The standard deviations of the ACO based algorithm is much smaller than those of TimeSeluth on both data sets. This indicates that the ACO based algorithm is more stable than TimeSeluth. Therefore, it is easy to use in practice.

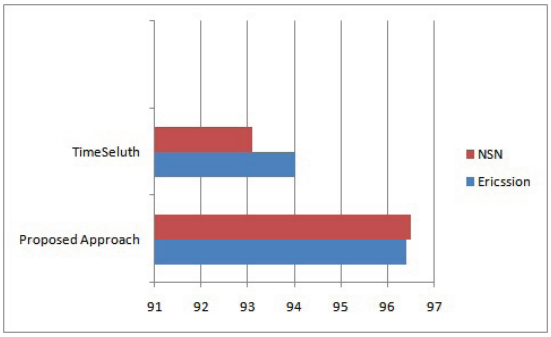

Fig. 6. Precision 10-fold cross validation.

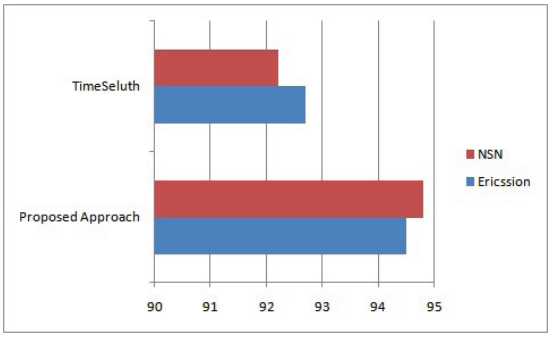

Fig. 7. Recall 10-fold cross validation.

Fig.(6) and Fig.(7) are the average precisions and recalls on the two data sets by the two algorithms. We can see that the ACO based algorithm again outperformed TimeSeluth on these two evaluation measures.

V. Conclusions

In this paper, we have presented a new method to use the ant colony optimization process to discover a set of time based rules from alarm training data and use the set of time based rules as a prediction model to predict the class of new alarms. We have formulated the search space for an ant to find a pathway as a square matrix indexed by the distinctive values of attributes in the training data. We have presented the methods to compute the pheromone distribution on the locations in the search space, the process to find a time based rule from the search space and a method to update the search space whenever a time based rule is found. We have proposed the ACO based algorithm to construct a set of time based rules for alarm prediction. The experiment results on two real world alarm data sets have demonstrated that the new algorithm improved prediction accuracy significantly in comparison with the existing alarm prediction system TimeSeluth. Cross validation results have also shown that the new algorithm is stable and easy to use in practice.

One direction for the future work is to develop the same kind of algorithms using particle swarm intelligence (PSO) to cater continuous values. Further investigation of input parameters such as support, confidence and coverage is also necessary to better understand the time based rule building process.

References

[1] A Finkel A T Bouloutas, S B Cola. "distributed fault identification in telecommunication network". *Journal of network and system management*, pages 295 – 312, 1995.

[2] Bouloutas A.Galo and Finkel A. "alarm correlation and fault identification in communication networksp". *IEEE Trans. on Communications 4(2/3/4)*, pages 523–533, 1994.

[3] M-Tahar Kechadi Jacques Henry Bellec and AbdelKamel Tari. "behavioural proximity discovery: an adaptive approach for cause analysis". *Int. J. Business Intelligence and Data Mining*, pages 259–285, 2011.

[4] Hanen Brahmi Imen Brahmi and Sadok Ben Yahia. "omc-ids: At the cross-roads of olap mining and intrusion detection ". *Knowledge Discovery and Data Mining*, pages 13–24, 2012.

[5] C.Borgelt. "keeping things simple: Finding frequent item sets by recursive elimination". *1st IEEE ICDM Workshop on Frequent Item Set Mining Implementations*, 2003.

[6] Gardner R Harle D. "alarm correlation and network fault resolution using kohonen selforganising map". *In: IEEE Global Telecom. Conf*, pages 1398–1402, 1997.

[7] Robert D. Gardner and David A. Harle. "alarm correlation and network fault resolution using the kohonen self-organising map". *Global Telecommunications Conference, 1997. GLOBECOM '97., IEEE*, pages 1398–1402, 1997.

[8] Hannu toivonen Heikki Mannila and A. Inkeri verkamo. "discovery of frequent episodes in event sequences". *Data Mining and Knowledge Discovery*, pages 259 – 289, 1997.

[9] J.Han J.Pei and Y. Yin. "freuqent patterns without candidate generation". *SIGMOD'*, pages 1–12, 2000.

[10] Jukic and Oliver. "abcde - alarm basic correlations discovery environment". *MIPRO*, pages 528 –533, 2010.

[11] Klaus Julisch. "mining alarm clusters to improve alarm handling efficiency". *Computer Security Applications Conference, ACSAC*, pages 12–21, 2001.

[12] Kamran Karimi and Howard J. Hamilton. "generation and interpretation of temporal decision rules". *International Journal of Computer Information Systems and Industrial Management Applications*, pages 314–323, 2011.

[13] Stefan Wallin Viktor Leijon and Leif Landen. "statistical analysis and prioritisation of alarms in mobile networks". *Int. J. Business Intelligence and Data Mining*, pages 4–21, 2009.

[14] Lundy L.ewis. "a case-based reasoning approach to the management of faults in communications networks". *INFOCOM '93. Proceedings.Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies. Networking: Foundation for the Future, IEEE*, pages 1422 –1429, 1993.

[15] Mohammad Jaudet Naeem lqbal and Amir Hussain. "temporal classification for fault-prediction in a real-world telecommunications network". *Emerging Technologies, Proceedings of the IEEE Symposium*, pages 209 – 214, 2005.

[16] Steinder M. "probabilistic fault localization in communication systems using belief networks". *IEEE/ACM Transactions*, pages 809–822, 2004.

[17] M. Zaki S.Parthasarathy M. Ogihara and W. Li. "new algorithms for fast discovery of association rules". *Knowledge Discovery and Data Mining*, pages 283–296, 1997.

[18] Jian Pei. "mining sequential patterns by pattern-growth: the prefixspan approach". *Knowledge and Data Engineering, IEEE Transactions*, pages 1424 –1440, 2004.

[19] Agrawal R and Srikant R. "fast algorithms for mining association rules". *Proceedings of the 20th Int'1 Conference on Very Large Databases*, pages 487–499, 1994.

[20] R.Srikant and R. Agrawal. "mining sequential patterns: Generalizations and performance improvements". *Lecture Notes in Computer Science*, pages 3–17, 1996.

[21] Wallin S.Ahlund and Nordlander. "rethinking network management: Models, data-mining and self-learning". *Proceedings of the 2012 IEEE Network Operations and Management Symposium*, pages 880 – 886, 2012.

[22] Yufeng Kou Chang-Tien Lu Sirirat Sinvongwattana and Yo-Ping Huang. "survey of fraud detection techniques". *Networking, Sensing and Control, IEEE International Conference*, pages 1201–1206, 2004.

[23] Himberg J Korpiaho K Mannila H Tikanmaki and Toivonen H. "time series segmentation for context recognition in mobile devices". *In: Proc. of the IEEE International Conference on Data Mining*, pages 203–210, 2001.

[24] Gary M. Weiss. "predicting telecommunication equipment failures from sequences of network alarms". *Handbook of Knowledge Discovery and Data Mining. Oxford University Press*, pages 891–896, 2002.

[25] Jain-Zhi Ouh Pei-Hsin Wu and Ming-Syan Chen. "experimental results on a constrained based sequential pattern mining for telecommunication alarm data". *Web Information Systems Engineering*, pages 186 – 193, 2001.