International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:4, No:4, 2010

Genetic-based Anomaly Detection in Logs of Process Aware Systems

Hanieh Jalali, Ahmad Baraani

***Abstract*—**Nowaday's, many organizations use systems that support business process as a whole or partially. However, in some application domains, like software development and health care processes, a normative **P**rocess **A**ware **S**ystem (PAS) is not suitable, because a flexible support is needed to respond rapidly to new process models. On the other hand, a flexible Process Aware System may be vulnerable to undesirable and fraudulent executions, which imposes a tradeoff between flexibility and security. In order to make this tradeoff available, a genetic-based anomaly detection model for logs of Process Aware Systems is presented in this paper. The detection of an anomalous trace is based on discovering an appropriate process model by using genetic process mining and detecting traces that do not fit the appropriate model as anomalous trace; therefore, when used in PAS, this model is an automated solution that can support coexistence of flexibility and security.

***Keywords*—**Anomaly Detection, Genetic Algorithm, Process Aware Systems, Process Mining.

I. Introduction

RECENTLY, management trends largely motivated organizations to use Process Aware Systems. The use of PASs leads to a shift from data-centric to process-centric systems. This causes business process logic to be completely separated from application programs, as a result it makes redesigning and extending of process models easy. Besides, legal requirements are also motivating companies to adopt PASs in order to support business process control.

However, in some application domains such as software development and health care processes, the business process control cannot be supported by normative PAS like a Workflow Management System (WMS). In these domains, the business process is not completely known before execution; moreover, they need to respond rapidly to new process models, so these application domains demand a flexible automation of their business processes. On the other hand, a flexible PAS may be vulnerable to undesirable and fraudulent executions, because users can execute these insecure tasks. These problems impose a tradeoff between flexibility and security. Thus, it is important to develop methods that support flexible PASs without compromising their security. Therefore, there is an absolute need for auditing systems and techniques to extract knowledge from the information recorded by nowadays information systems. Moreover, the vast growth of log data in the form of audit trail, transaction logs, and data warehouses and the requirement from a BPM (Business Process Management) perspective have resulted in the development of process mining techniques. "Process mining is mainly concerned with the discovery of process models from logs generated by information systems [1]". Recent developments in the field of process mining have led to a renewed attention in anomaly detection and security issues in Process Aware Systems [1].

A research work performed to detect anomalous processes in PASs [2] presents two methods based on α algorithm. For this work, there is a need to have a known "normal" log and then detect anomalous processes based on that log, so this approach is not suitable for application domains that need flexible support, because a "normal" log is not known before execution.

Other techniques to detect anomalous traces are presented in [3], [4], and [1]. [3], [4] present three different algorithms to detect anomalous traces: sampling, threshold, and iterative. In [1], anomalous traces are detected based on a formal definition of anomalous trace. These techniques are suitable for flexible application domains. However, [3], [4] cannot deal with larger logs, because of adopted process mining and [1] needs a precise appropriateness metric to select an appropriate model and also, an automated solution might be implemented.

In this work, the design goal is to look for an approach that can provide a desired level of tradeoff between flexibility and security and deal with large amount of logs and the last but not least, it is an automated solution.

With these design objectives in mind, a genetic-based anomaly detection model is presented to detect anomalous traces in Process Aware Systems. This model has three steps: Preprocessing, Genetic Process Discovery, and Classification. Genetic process mining is used to detect an appropriate process model and using that model in classification step, the preprocessed log gained from the first step is classified to anomalous and normal traces. If a trace does not fit the appropriate model, it will be an anomalous trace. A definition for an anomalous trace is presented. The Proposed model can be implemented in ProM framework. ProM is an open-source plug-able framework that provides a wide range of process mining techniques.

H. Jalali is a student with the Computer Department, University of Isfahan, Isfahan, Iran (phone: +98-311-6513041; fax: +98-311-7934095; e-mail: jalali@eng.ui.ac.ir).

A. Baraani, Ph.D., is a professor with the Computer Department, University of Isfahan, Isfahan, Iran (e-mail: ahmadb@eng.ui.ac.ir).

The approach provides security in PASs, by using a flexible automated solution, to discover anomalous traces. Flexibility, automation and ability to deal with large logs are provided through using genetic process mining.

The reminder of this paper is organized as follows. In section II, genetic process mining used in the genetic-based anomaly detection approach is described briefly. In section III, the genetic-based anomaly detection model for logs of PASs is presented. Also, in this section, it is presented that how the approach can be implemented with ProM framework. In section IV, some related works in the area of process mining, genetic process mining and anomaly detection in Process Aware Systems are reported. Furthermore, in this section, the proposed model is compared with other approaches and explained its advantages. Conclusion and future works are in section V.

II. Genetic Process Mining

Process mining techniques are used to discover the most appropriate model that best describes the behavior of the log generated by information systems. Genetic process mining is one of the most powerful techniques used for discovering process models.

"Genetic algorithms are adaptive search methods that try to mimic the process of evolution [13]". These algorithms start with a primary population of individuals (process models in this case). Every individual is assigned a fitness measure to show its quality. Populations evolve by selecting the fittest individual and generating new population using operators such as crossover and mutation. At the end of algorithm the globally optimal process model is selected [13].

In the next section, the genetic algorithm is introduced. In this work, the genetic process mining algorithm presented in [10] is used. For more information on the genetic process mining and its definitions the reader is referred to [10]–[12].

A. Genetic Algorithm

In this section, it is explained briefly how the genetic algorithm works. Fig. 1 describes its main steps: (i) the initialization process, (ii) the fitness calculation, and (iii) the genetic operators. These steps are described in the next subsections.

1) Initial Population

The genetic algorithm randomly builds initial population. In a given log, there is a set of activities, which contain all tasks available in the log. All individuals in every population of genetic algorithm have the same set of activities. However, the causality relations, inputs and outputs of every individual in the population may be different. Inputs and outputs of individuals in the initial population are randomly built by the genetic algorithm, so the initial population can have every trace in the search space defined by the set of activities. The higher the amount of tasks in a log, the bigger this search space is [10].

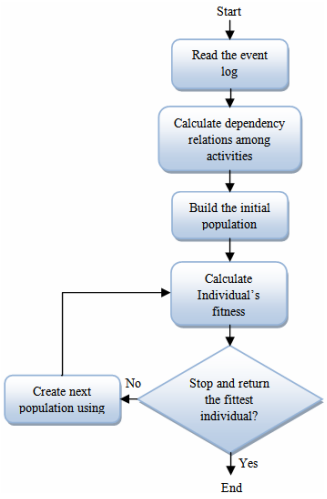

Fig. 1 Main steps of genetic algorithm

2) Fitness Calculation

In this approach, the fitness calculation is based on the number of correctly parsed traces in the event log. If an individual describes the behavior of a log correctly, this fitness value will be high. In a normal noise-free situation, the fitness measure is 1, but the fitness value is from 0 to 1 in practical situations [10].

3) Genetic Operations

Elitism, crossover and mutation are used to generate elements of the next genetic population. Elitism is used to copy a percentage of the fittest individuals in current generation to the next generation. Crossover and mutation are the basic genetic operators. Crossover builds new individuals of next generation (offsprings) based on the fittest individuals in the current population (parents). That is, crossover recombines inputs and outputs of individuals. The mutation operator changes some details of a population individual. This operation randomly changes subsets of inputs and outputs of an activity in an individual in order to add new useful individual in the population. The genetic algorithm stops when: (i) it finds an individual whose fitness value is 1; *or* (ii) it creates *n* generations, where *n* is the maximum number of generations allowed; *or* (iii) the fittest individual has not changed for *n*/2 populations generated in a row. If none of these conditions occurred, the Genetic Algorithm creates a new population as follows [10]:

"**Input**: current population, elitism rate, crossover rate and mutation rate

Output: new population

1. Copy "elitism rate × population size" of the best individuals in the current population to the next population.
2. While there are individuals to be created do:
   a. Use tournament selection to select parent1.
   b. Use tournament selection to select parent2.
   c. Select a random number r between 0 (inclusive) and 1 (exclusive).

d. If r less than the crossover rate:
then do crossover with parent1 and parent2. This operation generates two offsprings: offspring1 and offspring2.
else offspring1 equals parent1 and offspring2 equals parent2.

e. Mutate offspring1 and offspring2. (This step is only needed if the mutation rate is non-zero.)
f. Copy offspring1 and offspring2 to the new population.

3. Return the new population.

Tournament Selection To select a parent the tournament selection algorithm randomly selects 5 individuals and returns the fittest individual among the five ones [10]".

III. Genetic-based Anomaly Detection Model

Despite the different definitions available for an anomalous trace (e.g. noise, exception, or fraud), for this work, an anomalous trace is considered as an irregular execution that is different from an appropriate process model dynamically discovered during anomaly detection process. This approach is used, because in some application domains, a complete process model is not known before anomaly detection process, so it is very hard to define precisely an anomalous trace. On the other hand, another approach is to consider an anomalous trace as a rare event. However, classifying a trace just based on its frequency in the log is not simple, because it is possible for a rare event to be a "normal" trace and, in addition, it is very difficult to define a precise frequency, which describes an anomalous trace [1]. One of the advantages of the proposed model is that a low-frequent behavior [10] can also be discovered.

Fig. 2 provides an overview of the proposed approach which has three steps: (i) Preprocessing, (ii) Genetic Process Discovery, and (iii) Classification of log. The preprocessing phase is a domain dependent step that keeps (or removes) some important (or irrelevant) traces and activities from the original log. The next step deals with discovery of the most appropriate model. In this approach, genetic process mining is used for process discovery. In genetic process mining, a fitness measure is assigned to every individual in the population and purpose of algorithm is to discover the fittest individual by generating new populations. The most appropriate model is the fittest individual that best describes the behavior of the log. Finally, in the third step, the traces in the log are classified to normal and anomalous traces. Anomalous traces are those that are not instances of the most appropriate model discovered in the second step. In this paper, the term trace will be used to refer to process instances of a process model and represents the order of completing activities. The approach presented in [1] is followed for the proposed model.

The term trace is defined first and by the use of this definition, an event log is defined.

Definition 1. Trace [1].

"Given that A is a set of activities. Then, a trace t represents a sequence of activities such that $t \in A^*$. That is, assuming that A is an alphabet, and A^* denotes all possible words over A, then t is a word based on this alphabet."

Definition 2. Log [1].

"Given T as the set of all traces defined over A and $T' \subseteq T$, then a log L is defined as $L \subseteq T'^2$."

In the following, three steps of the proposed approach are described.

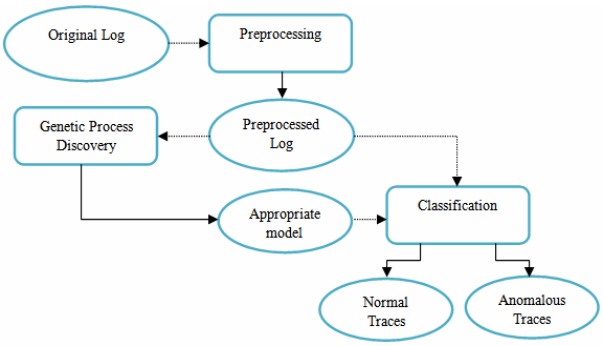

Fig. 2 Overview of the proposed genetic-based anomaly detection model

A. Preprocessing

The first step of the proposed anomaly detection approach is concerned with keeping (or removing) some activities or traces from log that are (not) appropriate and important for analysis based on the decision of domain analyst. In this step, domain analyst can remove incomplete traces that are not recorded completely in the log. These traces are not started and ended with certain expected start and end events. However, removing all incomplete traces may cause some anomalous traces not to be discovered, so domain analyst can add artificial start and end tasks to incomplete traces and keep these traces for analysis. Domain analyst can remove traces with activities that are not important for analysis. Furthermore, domain analyst can remove or keep traces with certain properties such as defined properties for originators, attributes of activities, or relations between activities. The final preprocessed log is used as an input for the second step

(see Fig. 2). This preprocessed log is called L^P.

For implementing this step, ProM has a lot of log filtering and analysis tools that can be used. To remove incomplete traces from log, *simple log filtering* tools can be used to define expected start and end activities. Then, traces that are not started and ended with these activities are removed. Moreover, *simple log filtering* tools can be used to define traces with certain activities to be remained or removed. To keep incomplete traces and artificially complete them, *advanced log filtering* tools can be used to define artificial start and end tasks to be added to incomplete traces. Analysis tools such as *LTL Checker* plug-in can be used to define certain properties in the form of formula and remove or keep traces with these properties.

B. Genetic Process Discovery

In order to classify normal and anomalous traces in a log, an appropriate process model should be used. This appropriate process model should be complete and precise to correctly present the behavior of the log. Process mining can be considered as a search for the most appropriate process among the search space of candidate process models.

Mining algorithms can use two different strategies to find the most appropriate model: (i) *local strategies* based on step by step constructing of the optimal model using existing local information. In these local strategies, there is no guarantee that the optimal process model constructed from locally optimal steps is globally optimal too. Hence, the performance of these local mining techniques are highly decreased when the necessary local information are not available. (ii) *Global strategies* based on constructing the optimal process model in a strike search. A genetic search is an example of a global strategy; because the search space is in a global level and the quality or fitness of the candidate model is calculated by comparing it with all traces in the event log [10]. On the other hand, genetic algorithms are naturally parallel. Since they have multiple offspring, they can traverse the search space in different directions simultaneously. If a path is not correct, they change their path to another direction [14]. Therefore, it is started to use genetic algorithms. For these reasons, genetic process mining described in section II is used for discovering an appropriate model.

The genetic algorithm used in this step is supported by the *genetic algorithm* plug-in of the ProM framework. In order to use *genetic algorithm* plug-in, value of some parameters such as population size, maximum number of generations, elitism rate, crossover rate, and mutation rate must be defined. The best value of these parameters for the proposed approach can be defined when implementing the model with ProM framework. The preprocessed log gained from the first step is used as an input for *genetic algorithm* plug-in. Output of this step is the most appropriate model discovered by genetic process mining that best describes the behavior of the preprocessed log in the form of *Heuristic Net*.

C. Classification

Finally, after selecting an appropriate process model in second step, we perform the last step of our anomaly detection approach, the classification of log in two sets: anomalous traces and normal traces. Using the definition of a fitness instance test function, a definition for anomalous trace is defined below.

Definition 3. Fitness Instance Test Function [1].

"f_M: L → IB is the fitness instance test function that indicates if a trace from a log L is an instance of a model M. A trace t is instance of a model M if t can be completely parsed by M. It can be defined as follows:"

$$f_M(t) = \begin{cases} true, if\ t\ can\ be\ replayed\ by\ model\ M \\ false, otherwise \end{cases}$$

Definition 4. Anomalous Trace.

Given log L, and M^* an appropriate model with maximum fitness value, then an anomalous trace $t \in L$ is defined as follows: $\neg f_{M^*}(t)$, i.e. $\{t \in L \mid \neg f_{M^*}(t)\}$ is the set of anomalous traces.

To summarize, between models that can be discovered from preprocessed log L^P, a model M^* is interested that has a maximum fitness value and is called the appropriate model, so **anomalous traces** are those in log that do not fit the appropriate model M^*.

The last step of the proposed anomaly detection approach can be implemented using *conformance checker* plug-in of ProM. Fitness instance test function is obtained indirectly by using *conformance checker* plug-in, since the function of this plug-in is the same as fitness instance test function. This plug-in checks the conformance of all traces in a log with a defined process model. The processed log L^P and the appropriate model M^*, in the form of *Petri Net*, are inputs of the *conformance checker* plug-in. Every trace in log L^P is checked whether it can be replayed by appropriate model M^* or not. In this manner, traces in the preprocessed log can be easily classified as follows: (i) fitting traces as normal traces; and then (ii) other traces as anomalous traces.

The input process model of *conformance checker* plug-in must be in the form of *Petri Net*, thus the most appropriate model discovered in the second step must be converted from *Heuristic Net* to *Petri Net* before using in *conformance checker* plug-in. *Conversion* plug-in of ProM is used for this purpose.

IV. Related Work and Discussion

Process mining is a way to analyze systems and their actual use based on the event logs they produce and its goal is to extract information about processes from transaction logs [5]–[8]. Process mining techniques can be used to discover how people really work, support business process modeling, and detect the enterprise work practice. Another usage of process mining techniques is to construct social networks as in [9].

The focus of this paper is on the genetic process mining. For more information on the genetic algorithm, the reader is referred to [10]–[13].

In the anomaly detection domain, recent researches have been performed to tackle the problem of identifying anomalous traces in PASs [2]–[4], and [1]. In [2], the authors present two methods based on α algorithm. In these methods, a known "normal" log or model is needed to be mined to define a classifier, whose function is to audit a separated log, whereas these methods are not suitable for application domains that need flexibility, because a "normal" log is not known before execution in these domains. In [3], [4], the authors present three different methods to detect anomalous traces: sampling, threshold, and iterative methods. However, these methods have practical limitations, because of adopted process mining that cannot deal with larger logs. In [1], the authors present an approach based on a formal definition of anomalous trace, which is defined through two parameters: (i) fitness model degree (p %); and (ii) appropriateness of model (*a*). Then, it is described how ProM framework can be utilized to support this formal definition. However, as mentioned by the authors, in this approach, a precise appropriateness metric is not defined for selection of an appropriate model. Moreover, an automated solution might be implemented, for example through the use of genetic algorithms.

The proposed genetic-based anomaly detection model in this paper has some advantages, in comparison with prior approaches:

- In this approach, the appropriate model is discovered during anomaly detection process, so a "normal" log is not needed before execution. In addition, a GA-based system can be retrained easily for changes. Therefore, this anomaly detection approach is flexible and it can respond rapidly to new market strategies and process models.
- By using genetic process mining, the most appropriate model is discovered automatically and it is not needed to define a precise appropriateness metric for selecting the appropriate model.
- Because of the parallelism in GAs, they can evaluate many process models at once; hence they are suitable to solve problems with a large search space in a reasonable amount of time.
- In comparison with other process mining techniques, GA can deal with all the constructs possible in a log (sequences, parallelism, choices, loops, and non-free-choices, invisible tasks and duplicate tasks) and is naturally robust to noise.

V. Conclusion and Future Work

Nowadays, organizations are motivated to use Process Aware systems in order to support their business process control better, but normative PASs are not appropriate for all application domains, since in some domains a flexible support is needed to respond rapidly to new market strategies and process models. On the other hand, a flexible PAS may compromise security issues. In this paper, a genetic-based anomaly detection model was presented to provide a tradeoff between flexibility and security in Process Aware Systems. This approach is an automated solution that uses genetic process mining to discover an appropriate model of event logs and then detecting anomalous traces based on formal definition presented. It was described how ProM framework can be utilized to implement this approach. In the future, the proposed approach will be evaluated with a real log in ProM framework.

The proposed anomaly detection approach is concerned with the control-flow perspective. For example, a fraudulent execution may have a normal path, but it is executed by unauthorized roles or users or produce anomalous data and it is not discovered. Therefore, in the future, data and organizational perspectives will be considered to provide more accuracy.

References

[1] Fabio Bezerra, Jacques Wainer, and W. van der Aalst, "Anomaly detection using process mining," Springer-Verlag Berlin Heidelberg, 2009, pp. 149–161.

[2] W.M.P. van der Aalst, and A.K.A. de Medeiros, "Process mining and security: Detecting anomalous process executions and checking process conformance," Electronic Notes in Theoretical Computer Science, vol. 121(4), 2005, pp. 3–21.

[3] F. Bezerra, and J. Wainer, "Anomaly detection algorithms in logs of process aware systems," SAC 2008: Proceedings of the 2008 ACM symposium on Applied computing, ACM Press, New York, 2008, pp. 951–952.

[4] F. Bezerra, and J. Wainer, "Anomaly detection algorithms in business process logs," ICEIS 2008: Proceedings of the Tenth International Conference on Enterprise Information Systems, Barcelona, Spain, June 2008. AIDSS, pp. 11–18.

[5] W.M.P. van der Aalst, B.F van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M Weijters, "Workflow mining: A survey of issues and approaches," Data & Knowledge Engineering, vol. 47(2), 2003, pp. 237–267.

[6] W. van der Aalst, A. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," IEEE Transactions on Knowledge and Data Engineering, vol. 16(9), 2004, pp. 1128–1142.

[7] R. Agrawal, D. Gunopulos, and F. Leymann, "Mining process models from workflow logs," Sixth International Conference on Extending Database Technology, 1998, pp. 469–483.

[8] J. Cook and A. Wolf. "Discovering models of software processes from event-based data," ACM Transactions on Software Engineering and Methodology, vol. 7(3), 1998, pp. 215-249.

[9] W.M.P. van der Aalst and M. Song, "Mining social networks: Uncovering interaction patterns in business processes," M. Weske, B. Pernici, and J. Desel, editors, International Conference on Business Process Management (BPM 2004), Lecture Notes in Computer Science, Springer-Verlag, Berlin, 2004.

[10] W.M.P. van der Aalst, A.K. Alves de Medeiros, and A.J.M.M. Weijters, "Genetic process mining,", Applications and theory of Petri nets, 2005 - Springer.

[11] A.K.A. de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst, "Using genetic algorithms to mine process models: Representation,

operators and results," BETA Working Paper Series, WP 124, Eindhoven University of Technology, Eindhoven, 2004.
[12] Ana Karla Alves de Medeiros, "Genetic Process Mining," Eindhoven University of Technology, ISBN 978-90-386-0785-6, 2006.
[13] A.K. Alves de Medeiros, A.J.M.M. Weijters and W.M.P. van der Aalst, "Genetic Process Mining: A Basic Approach and its Challenges," .
[14] Zorana Bankovic, José M. Moya, Álvaro Araujo, Slobodan Bojanic, and Octavio Nieto-Taladriz, "A Genetic Algorithm-based Solution for Intrusion Detection,", Journal of Information Assurance and Security 4, 2009, pp. 192-199.