ANOMALY DETECTION IN DATABASES BASED ON ONTOLOGY GRAPH

Reza Asgari[1], Ali Ahmadian Ramaki[2] and Reza Ebrahimi Atani[3]
[1]Department of Computer Engineering, Guilan University, Rasht, Iran
rezaasgari@msc.guilan.ac.ir
[2]Department of Computer Engineering, Guilan University, Rasht, Iran
ahmadianrali@msc.guilan.ac.ir
[3]Department of Computer Engineering, Guilan University, Rasht, Iran
rebrahimi@guilan.ac.ir

ABSTRACT

Using graphs for extracting data and presenting them gains more importance day by day within different scopes in order to detect structural and semantic patterns. One of the most significant scopes in this field is anomaly detection. This paper offers a framework for anomaly detection in information systems, especially in relational database systems, using ontology graph. In this paper, we first suggest a structure to present data with the help of ontology. Then we introduce our framework in order to detect anomalies in relational database systems. An example is put forward to prove the ability of the framework for anomaly detection.

KEYWORDS

Ontology graph, Anomaly, Relational database, Access control systems

1. INTRODUCTION

Anomaly detection within different scopes is one of challenges in data mining [9]. Using static methods contribute to success in several fields e.g. Intrusion Detection. Not only have recent studies on graph-based approaches helped complete previous methods but they have provided a mechanism to analyze data not processed in traditional ways [9]. Various approaches have been suggested to detect graph-based anomalies; however using graph, as an efficient approach, is still a big challenge in order to determine anomaly detection [2]. For example, in [3, 4], a method for resolving anomaly detection problems is offered by comparison of user's behavior graph and normal behavior patterns[5]. In an anomaly domain is first defined for the system, hence every behavior within such domain is considered as anomaly. A solution based on GBAD –Graph Based Anomaly Detection [6] is determined in order to detect destructive behaviors of insiders in [7, 8] so that graph-based anomaly detection is used to overcome insider attacks problems.

Methods stated above enjoy advantages but they suffer a weakness in common i.e. lack of attention on semantic relationships between data and user's behaviors. Consequently, this paper intends to offer a framework to detect anomalies in relational database systems according ontology graphs. This framework provides chances to study the issue of anomaly detection more conceptual. In section (2) a graph-based ontology is first introduced so as to present data and then

our framework for anomaly detection is brought in. In the end of the section an example, for illustration, is yielded. In section (3) system implementation and behavior assessment is discussed and finally an overall conclusion of the issue is drawn in section (4).

2. THE SUGGESTED APPROACH BASED ON ONTOLOGY

In this section an ontology-based approach is introduced in order to detect anomalies in relational database systems. Following the section, firstly ontology graph and secondly our framework are introduced.

2.1. Used Ontology Graph to Present Data

To create the in demand ontology, we employ rules in [10] for ontology graph construction to present our model. Generally an ontology is presented by three factors [11]:

- Explaining relationships between classes (concepts)
- Explaining relationships between instances
- Explaining relationships between classes, attributes and instances

So we define ontology as a definite set of O according to equation (1). We also use the set of R in equation (2) in order to specify relationships.

$$O = \{o_1, o_2, \ldots, o_n\} \quad (1)$$

$$R = \{\text{isA, synOf, partOf, atr, val}\} \quad (2)$$

In equation (1), O is a set forming the ontology of a database. $o_i$is the used ontology in the database.

In equation (2) “isA” is to describe relationships including inheritance between concepts, “synOf” is to describe a set of all current synonyms for a concept, “partOf” is to describe relationships of type Aggregation (relationships in which a concept is a subpart of a global concept. Although by omitting global concept subparts would not be deleted. For instance, imagine the relationship between a football player and a club. By deleting club, nature of players to it would not be deleted because they can be recruited in another club.) “Atr” and “Ins” are used to respectively represent presentation of attributes to a concept and having access to values of a concept.

Each $o_i \in O$ and $r_i \in R$ are respectively considered a concept and a relationship between two concepts. An ontology is defined as a graph G(V,E) [10] which V is a definite set of vertices and E is a definite set of edges. Every vertex is tagged with a concept and every edge is a connection between two vertices.

Each tag for node $n \in N$ is defined by $N(n) = o_i \in O$ mapping each node onto a string of o_i. Each edge $e \in E$ is tagged through a function $T(e)$ mapping each edge onto a string of R. Edges tagged as “atr” point to a head of link list including set of attributes of a concept so that values of concepts are approachable through tracing nodes with tag “val”. The other edges point to current concepts in ontology. To illustrate the issue we take a look at Fig. 1 which is part of products ontology. Finally a database in form of equation (3) is defined:

$$DB = \{G(V,E), O, R, N, T\} \quad (3)$$

Such graph enjoys being simply updated. In case of inserting a new tuple into a table or deleting it, it is enough to modify the linked list connected to the table values. Such task is feasible at low cost and time consumption.

So as to compare two graphs and discover similarities between them, we apply the method [12] which we employed to reveal similitude in ontology graph in order to perform alignment operation. To do so, as for the method, we must take these three factors into consideration: (I) structural similarity, (II) semantic similarity. Structural similarity studies general graphs structure regardless to nature of concepts. Semantic similarity yields the number of namesake tags in total tags of ontology graph.

In order to calculate the factors many methods have been suggested; however we employ those in [11, 13, 14] with a couple of slight modifications for adjustment to defined concepts of ontology.

We define equation (4) as follows so as to measure structural similitude:

$$\text{Structure } (o_1, o_2) = \frac{num\_sim\_c + num\_sim\_p}{\max\left(num\_c1 + num\_p1\ , num\_c2 + num\_p2\right)} \qquad (4)$$

Where $c \in$ {set of classes}, $p \in$ {set of attributes},$c_1, p_1 \in O_1$and $c_2, p_2 \in O_2$ so that $O_1$and O_2denote respectively the set of ontologies indicating current behaviors and the set of ontologies of user's behavior. Furthermore num_c_1refers to the number of current concepts (classes) in O_1containing sub-concept num_p_1refers to the number of current attributes in O_1containing sub-attributes and so are $num_c_2$and num_p_2 defined. In order to determine num_sim_c and num_sim_p we practice as (5) wherein $length\_root\_x_i$ and $length\_leaf\_x_i$ are respectively distances from the root and the leaf.

After discovering structural similarity, we apply (6) in order to measure semantic similarity wherein$|C_1|$,$|C_2|$,$|P_1|$ and $|P_2|$ are numbers of total concepts and attributes in ontologies. num_equal_candnum_equal_p are also ,in order, numbers of similar tags among current concepts and attributes in two ontologies.

```
Begin
   x = p or c
num_sim_x =0
   insert all x ⊆ o1 in x1 and x ⊆ o2 in x2
   while x1 and x2 is not empty do
      if (lengh_root_x_i is equal to lengh_root_x_jandlengh_leaf_x_i is equalto lengh_leaf_x_j)
num_sim_x = num_sim_x +1
Remove x1_i and x2_j from x1 and x2
      end if
   end while
End       (5)
```

$$\text{Label } (o_1, o_2) = \frac{num\_equal\_c + num\_equal\_p}{\max\left(|c1| + |p1|\ , |c2| + |p2|\right)} \qquad (6)$$

Having discovered semantic and structural similarity factors between user's behavior ontology and each ontology related to current behavioral patterns, the pseudo code (7) is employed so as to find the most similar pattern

```
Begin
   for each (o_i, o_j) ∈ (O1, O2)
      add decision vector < Structure(), Lable() > to dv
      insert d_0 in p and remove it from dv
 for each d_i in dv do
    if Dominate(p, d_i) do
       p = d_i
       remove d_i from dv
       continue
    else remove d_i from dv
  end for
     /* p is pattern that is most similar to user behavioral    pattern   */
End                                    (7)
```

In algorithm (7), $O_1$and O_2 symbolize the set of ontologies indicating current behaviors and the set of ontologies of user's behavior in order. Function Structure() returns a numeric value for structural similitude regarding equation (4) and function Label() returns a numeric value for semantic similitude regarding equation (6). Function Dominate(p, d_i) examines with respect to Pareto domination notation [1] if decision vector p excels decision vector d_i. In other words if decision vector p dominates decision vector d_i. For more information about Pareto optimization and the used algorithm you may refer to our previous study on alignment of ontologies in [12].

2.2. The Suggested Framework

The suggested framework for anomaly detection consists of four phases so that each phase performs analysis and operations on data and the results are then passed on to the next phase. User's query alongside user's access level to the database are received Access Control System as system inputs. Eventually after application of certain actions, in case of anomaly detection, it communicates an appropriate warning message by Alert System to the administrative system of the organization (Fig. 2). We later study phases in detail.

In the first phase after receiving user's query, his access level is yielded to Accessibility Ontology through Access Control. This unit is responsible for construction ontologies concerning user's access level to the database. . In our system a user's access is defined this way: user is permitted to read tables data. The operation is called SELECT. Operation of UPDATE, INSERT and DELETE are respectively defined as modifying a record in a table, adding a new record into table and removing a record from a table.

Every user may have relationships in accord with his access level as follows:

- Admin: authorized for UPDATE, SELECT, INSERT and DELETE.
- Select: authorized for SELECT
- Update: authorized for UPDATE
- Insert: authorized for INSERT
- Delete: authorized for DELETE

In Fig. 3 four examples for users' communication with a part of the ontology table of Fig. 1 is brought in form of an ontology graph. In this figure, admin is authorized to read data in Calculator class or to modify them as well as to insert a tuple into the table of the class and to delete a tuple from it. User 1 has all admin's authorization except for modification. User 2 is only

capable of reading or modifying data in the field TYPE. User 3 is only capable of reading data in the field MODEL not any other operations.

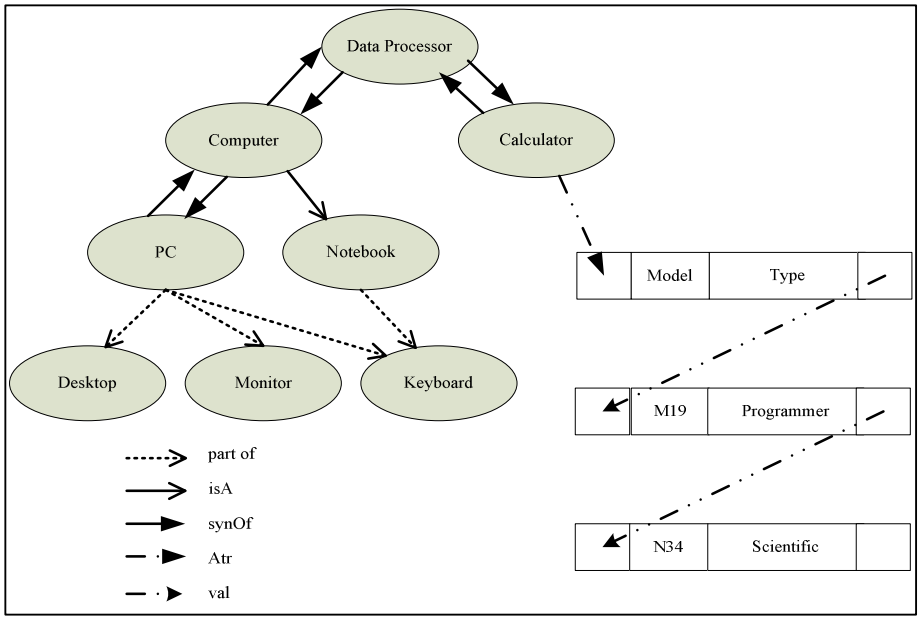

Figure 1.Ontology graph concerning a part of products of a computer equipment distribution firm.

In the second phase, the outcome ontology is transferred to the database as well as user's query (Data Ontology present data of the database in the form of a set of ontologies described in section II part A) in order that the most proper response is given to the query. The output of the current phase is an ontology containing information about database response to user's query plus sequential accesses to data. Such ontology indicates a certain user's behavioral pattern for an input query to the system and is transferred to the next phase in order to be compared with current behavioral patterns in the system.

In the third phase, we employ the method given in the section II part A so as to compare user's behavioral pattern with current normal behavioral patterns in Pattern Ontology. In the end of the phase, the most similar behavioral patterns to Analysis Phase for final examination.
Fourth phase analyzes behavioral patterns corresponding with user's current behavior. The phase checks level of similitude between analyzed behavioral patterns. In case of agreement it is considered as a normal behavior and the feedback of such analysis is then sent to Pattern Ontology unit to be used in the future accesses, otherwise it is realized as an anomaly in the system and the information is sent to Alert System in order to yield a warning to the system administration corresponding to anomaly type.

Since user's behaviors sequences are registered in the system, if any attack detected, these sequences can be applied to discover the logic of the recent behaviors which may be an attack and troubleshoot security bugs in order that the system become more secured against next behaviors.

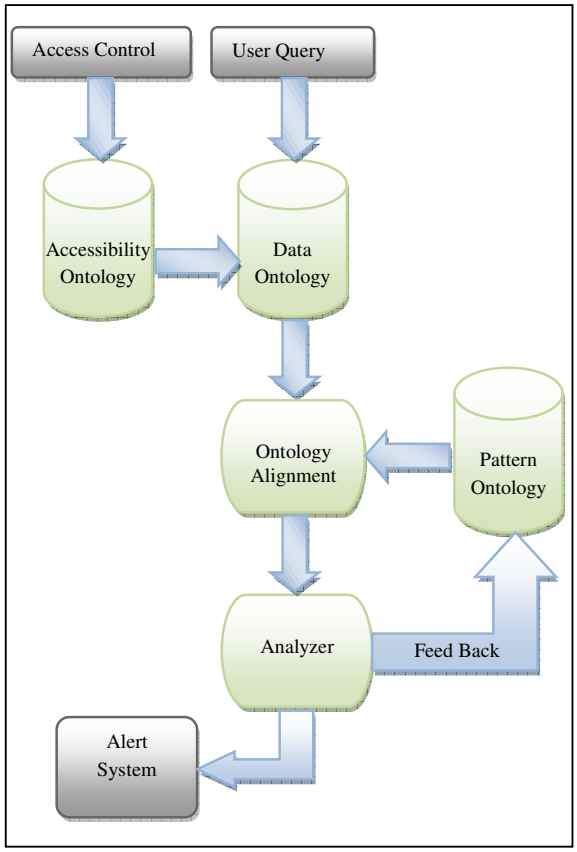

Figure 2.The suggested framework for anomaly detection in relational databases based on ontology graph.

2.3. Case Study

To illustrate the issue we follow up with an example. Let's assume a user has a behavior similar to current graph (a) Fig. 4 and corresponding behavioral patterns to his behaviors stored in Pattern Ontology are similar to (b) and (c) Fig. 4 (to simplify the figure we avoid depict patterns in the form ontology thus the it only presents a simple graph of them).

Here after examining user's access level and response to his query, in the phase three user's behaviors and stored behavioral patterns must be compared to measure normality degree. First, patterns (a) and (c) in Fig. 4 are compared to determine similitude degree between these two behaviors. Their similitude degree is depicted in dashed line in (d). Thus, the left part of user's behavioral graph corresponds to a normal pattern. So far user's behavior is considered normal. In the next level, (a) and (b) are compared and their similitude degree is in dashed line in (e) in Fig. 4. It is evident that the right part of the graph shows a normal behavior. In the end, the overall similitude of user's behavior with current behaviors is depicted in (f) in Fig. 4. As dash lined show user's behavior corresponds to two current normal behaviors consequently it can be a normal behavior and is checked in Analysis Phase.

In Analysis Phase you simply notice that behavior of (g) does not correspond to any other patterns hence this part of user's behavior is identified as an anomaly and is transferred to Alerting System to inform the administrative system with an appropriate message. Such behavior

can also be assessed in the system to detect the vulnerable section allowing for such malfunction as well as the probable logic behind the behavior which may be an attack on the system.

3. IMPLEMENTATION AND EVALUATION

To implement our approach a benchmark of more than hundred different sets of ontologies, OAEI-2010 [15], is employed as the database. Code pieces are written in Java to transform ontologies in the benchmark stored in XML to the graph defined in section II part A. We also consider 20 users with different access levels to various parts of the database stored in the form the ontology in Fig. 3. In order to specify normal behavioral patterns, a certain amount of normal behaviors are stored for each user which are updated and completed in the course of executing requests. The suggested system is programmed in Java and a program is written to cast random requests so as to test the system. The program creates requests with normal and abnormal access levels corresponding to access levels specified for users. In order to check various conditions, each request falls under one group below:

A. A 100% normal request

B. A request of at least a node of anomaly but no edge of anomaly – e.g. a node with different tag

C. A request of at least a node of anomaly and at least an edge of anomaly – e.g. a node inserted or deleted correspondingly an edge is inserted or deleted.

D. A request of at least an edge of anomaly without any node of anomaly – e.g. having an edge more or less or an edge with a different tag

Requests above may be delivered to the system iteratively to examine conditions which user's request exists in Pattern Ontology.

After executing the program and checking several instructions to evaluate the system performance, the conditions below are examined:

I. The behavior is normal and detected normal by the system - specified as TP (true positive).

II. The behavior is normal but detected abnormal by the system - specified as TN (true negative).

III. The behavior is abnormal but detected normal by the system - specified as FP (false positive).

IV. The behavior is abnormal and detected abnormal by the system - specified as FN (false negative).

For requests at 100% normality conditions I and II are likely to occur. Other requests are checked only in conditions III and IV. Fig. 5 shows detection probability TP and TN for first and second executions of requests –during request execution, Pattern Ontology is updated according to the first round requests execution. As indicated in this figure, in case of proximity between behavioral patterns and user's behavior in the system, the false detection is drastically reduced.

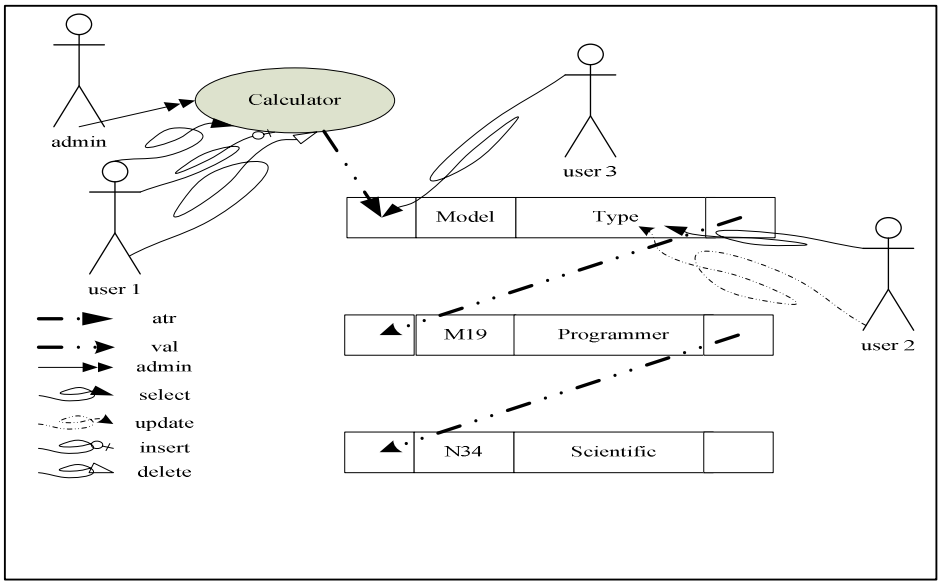

Figure 3.An ontology that represent relations between four users and ontology in Fig.1.

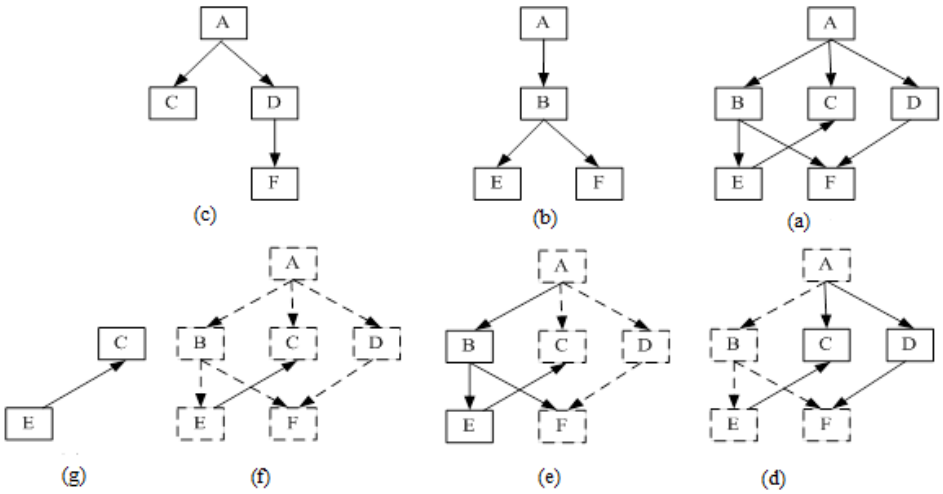

Figure 4. (a) an example of comparison of user's behavior, (b) and (c) current pattern, similitude degree is shown in dashed line in (d), (e) and (f), the part of user's behavior indicating anomaly is presented in (g).

Fig. 6 shows conditions in which one edge or more have anomaly with no node of anomaly alongside FP and FN occurrence probability for these conditions.

FP and FN occurrence probability, for conditions one node or more are of anomaly with no change in edges is depicted in Fig. 7. As concluded from Fig. 6 and Fig. 7, anomaly detection of nodes are simpler than that of edges. Also, Fig. 8 depicts FP and FN occurrence probability for conditions one node or more and at least one edge is of anomaly.

4. CONCLUSIONS

In this paper we offered an approach to anomaly detection in relational databases based on ontology graphs. Consequently we suggested a structure to present data according to ontology. Then we introduced a framework, based on users' behavioral patterns and ontology in order to detect anomaly in the system and finally we employed a method based on Pareto Optimization to measure similitude level of user's behavior.

By using such approach we discover probable anomalies in the system as well as the logic behind these anomalies and data streams contributing to such anomalies in order to make the system more secured against following anomalies.

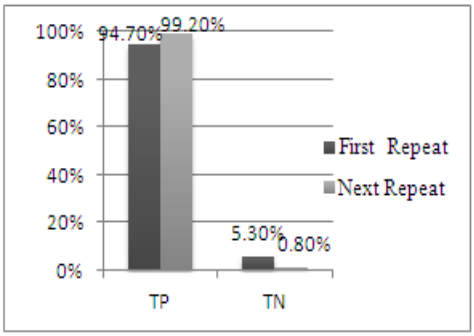

Figure 5. detection probability of TP and TN during first round of execution of 100% normal behaviors and second round of requests.

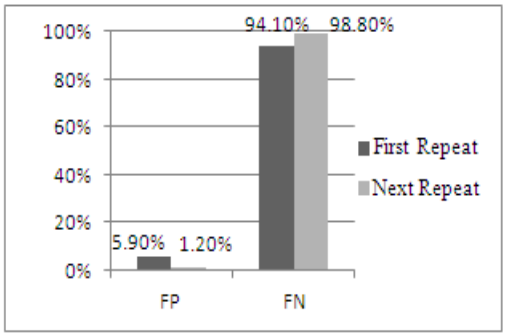

Figure 6. FP and FN detection probability in the first round of execution of requests having anomaly in one edge or more; with no node of anomaly and second iteration of requests.

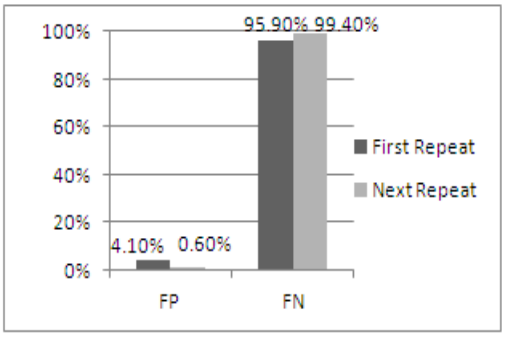

Figure 7. FP and FN detection probability during first round of execution of requests having anomaly in one node or more; with no edge of anomaly second round of the requests.

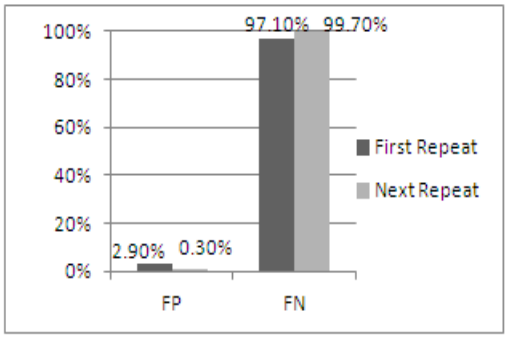

Figure 8. FP and FN detection probability during first round of execution of requests having anomaly in both nodes and edges and second round of the requests.

REFERENCES

[1] K. Deb, "*Evolutionary Algorithms for Multi-Criterion Optimization in Engineering Design*" Kanpur Genetic Algorithms Laboratory (KanGAL), Depatment of Mechanical Engineering, Indian Institute of Technology, Kanpur, India, 1999.

[2] S. Calderara, U. Heinemann, A. Prati, R. Cucchiara, N.iTishby, "*Detecting anomalies in people's trajectories using spectral graph analysis*", Computer Vision and Image Understanding, pp. 1099–1111, 2011.

[3] A. Lakhina, M. Crovella, C. Diot, "*Diagnosing network-wide traffic anomalies*", Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, ACM, USA, pp. 219–230, 2004.

[4] A. Lakhina, M. Crovella, C. Diot, "*Characterization of network-wide anomalies in traffic flows*", Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, ACM, USA, pp. 201–206, 2004.

[5] L. Huang, X. Nguyen, M. Garofalakis, M. Jordan, A. Joseph, N. Taft, "*In-network PCA and anomaly detection*", Neural Information Processing Systems, 2007.

[6] W. Eberlea, L. Holder, "*Mining for Structural Anomalies in Graph-Based Data*", International Conference on Data Mining. June, 2007.

[7] W. Eberlea, L. Holder, "*Insider Threat Detection Using Graph-Based Approaches*", Cyber Security Applications & Technology Conference For Homeland Security, IEEE, pp. 237-241, 2009.

[8] W. Eberlea, L. Holder, "*Applying Graph-Based Anomaly Detecti on Approaches to the Discovery of Insider Threats*", IEE, pp. 206-208, USA, 2009.

[9] W. Eberlea, L. Holder, "*Anomaly detection in data represented as graphs*", Intelligent Data Analysis, pp. 663–689, 2007.

[10] C.B. Necib, J. Freytag, "*Ontology based query processing in database management systems*", Proceeding on the 6th international on ODBASE, pp. 37-99, 2003.

[11] A.Mazak, M. Lanzenberger, B. Schandl, "*iweightings: Enhancing Structure-based Ontology Alignment by Enriching Models with Importance Weighting*", International Conference on Complex, Intelligent and Software Intensive Systems, pp. 992-997, 2010.

[12] R. Asgari, M. Moghaddam, S. Sahraei, R. A. Ebrahimi, "*An Approach to Ontologies Alignment Based Upon Pareto Front*", 5th International Computer and Instructional Technologies Symposium, Turkey, 2011.

[13] L.Juanzi, J. Tang, Y. Li, Q. Luo, "*RiMOM: A Dynamic Multistrategy Ontology Alignment Framework*", IEEE transactions on knowledge and data, Vol. 21, pp. 1218- 1232, 2009.

[14] C. Xie, "*Semantic Similarity-Based Ontology Alignment for Enterprise Ontologies*", 6th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 386-390, 2009.

[15] http://oaei.ontologymatching.org/2010/benchmarks.

Authors

Reza Asgari

Reza Asgari was born in Iran, Ghazvin. He received his BSc degree from university of Guilan, Iran in 2011. He is now MSc student at university of Guilan, Iran. His research interests in operating system and database security.

Ali Ahmadian Ramaki

He was born in Iran on August 10, 1989. He received his BSc degree from University of Guilan, Iran in 2011. He is now MSc student at university of Guilan, Iran. His research interests in computer security, network security and intelligent intrusion detection.

Reza Ebrahimi Atani

Reza EbrahimiAtani received his BSc degree from university of Guilan, Rasht, Iran in 2002. He also received MSc and PhD degrees all fromIran University of Science and Technology, Tehran, Iran in 2004 and 2010 respectively. Currently, he is the faculty member and assistant professor at faculty of engineering, University of Guilan. His research interests in cryptography, computer security, network security, information hiding and VLSI design.