# Novel Graph Based Anomaly Detection Using Background Knowledge

**Sirisha Velampalli**
Department of Computer Science & Engineering
Jawaharlal Nehru Technological University
Kakinada, AP India
sirisha.velampalli@gmail.com

**William Eberle**
Department of Computer Science
Tennessee Technological University
Cookeville, TN USA
weberle@tntech.edu

**Abstract**

The use of graph based anomaly detection has applications in a variety of diverse fields including health care, networks, finance, and insurance. Detecting anomalies using graphs has become important recently due to the interdependence of data from the web, emails, phone calls, etc. In this paper, we introduce a novel approach for graph-based anomaly detection by adding background knowledge to the evaluation metrics used in a traditional graph-mining approach, where we bias the substructure discovery process towards discovering anomalous substructures. Background knowledge is added in the form of *rule coverage*, which reports the percentage of the final graph covered by the instances of the substructure. Since one would expect that anomalies would be infrequent, it is our hypothesis that by assigning negative weights to the rule coverage, we can discover anomalous substructures. We are able to empirically evaluate that our proposed approach is comparable in accuracy to other approaches, and because the search space is reduced, do it in a fraction of the time. We test our approach on the well-known KDD Cup 99 network intrusion dataset.

## Introduction

In a highly-connected world, a huge amount of data is being generated from social networks, blog networks, telephone networks, etc. In addition to their voluminous nature, much of the generated data is inter-dependent in nature, whereby links exist or can be inferred between entities across domains. Representing these types of data as *graphs* provide for a meaningful representation that can be used for searching, analyzing, or discovering interesting patterns. Detecting anomalies from datasets represented in the form of graphs is known as *graph based anomaly detection* [Padmanabhan et al. 2014]. Graph based anomaly detection enables one to analyze datasets in a way that traditional data mining approaches cannot do easily: looking for *structure* in a network of data [Akoglu et al. 2014]. In this work, we present a novel approach for detecting *anomalous* substructures in a graph using *background knowledge.* The key to the proposed approach lies in the definition of anomaly. Our definition of anomaly is same as the one defined by [Noble and Cook 2003], i.e., anomalous substructures occur infrequently when compared to normative substructures. In their approach, they use the SUBDUE pattern discovery system which defines relevant substructures as those that compress the graph the best using the Minimum Descriptive Length (MDL) principle [Peter and Centrum 2005]. In earlier work by [Cook and Holder 1994], they use background knowledge to further refine the search process for discovering interesting normative patterns. In this work, it is our hypothesis that one could *use background knowledge in the form of "rule coverage"*, augmenting existing evaluation techniques, such as MDL and size (used in approaches like SUBDUE), *to discover anomalous substructures*.
The contributions of this work are as follows:

- A novel graph based anomaly detection approach,
- The use of a background knowledge rule, with MDL and size metrics, to aide in discovering anomalous substructures,
- A comparative analysis of MDL and size metrics,
- Calculating a prior value of rule coverage that guides the discovery of anomalous substructures, and
- Empirical evaluation on KDD Cup 99 datasets.

In the following sections, first we cover basic definitions, types of anomalies and in particular graph based anomalies. Then we give related work on graph-based anomaly detection. Next we present our proposed algorithms followed by detailed discussion of algorithms. We then present an implementation of our proposed approach followed by the data used in our experiments and the results. We evaluate our approach on the well-known

KDD Cup 99 network intrusion dataset. We then conclude this work with comparisons and potential future work.

## Background

First, we will present some definitions related to graphs that we will be using in this work, followed by a discussion of graph-based anomalies.

**Definition 1** *A graph G= (V, E) consists of a set of vertices where V= { v1, v2, v3,... } and a set of edges where E={ e1, e2, e3,...}*
**Definition 2** *A vertex, or node, in a graph represents a single entity [3].*
**Definition 3** *An edge, or link, in a graph represents a relationship or transaction between two vertices [3].*
**Definition 4** *Anomaly detection is the process of identifying data points, items or observations that do not conform to expected behaviour [1].*
**Definition 5** *Coverage refers to the fraction of a substructure in a graph that is described by the instances of the corresponding substructure.*

### Graph-Based Anomalies

An anomaly in a graph can be the result of various different types of *structural* changes [Eberle and Holder 2007]. Structural changes include the following:

- Insertions: An unexpected vertex or edge is present.
- Modifications: The type of vertex or edge is unexpected.
- Deletions: An expected vertex or edge is absent.

Figure 1 demonstrates each of the structural changes.

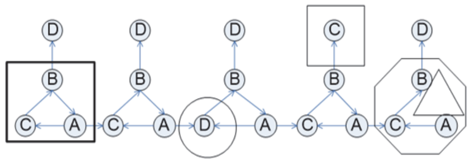

*Figure 1. Example Graph Showing Different Types of Anomalies*

In Figure 1, the vertex encircled represents an anomalous modification (i.e., the vertex is labelled with a "D" rather than an expected "C"); the vertex blocked with a square represents an anomalous extension (i.e., an extension to a vertex labelled "C", rather than "D"); and the edge outlined with a triangle represents an anomalous deletion (i.e., missing an edge).

In this work, we will adopt the definition of a *graph-based anomaly* as the one defined by [Noble and Cook 2003]. Generally, anomalous substructures occupy only a small portion of the entire graph. Frequent substructures are those which generally consist of repeatable substructures, whereas anomalous substructures are less repeatable. Take the example shown in Figure 2.

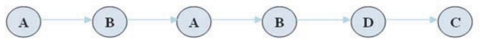

*Figure 2. Sample Graph*

In this example, the substructure "A-B" occurs twice, while the substructure "D-C" only appears in the graph once. Thus, "D-C" is considered anomalous.

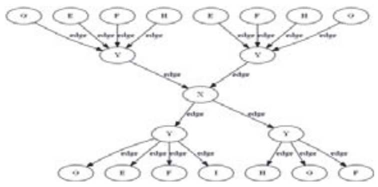

*Figure 3. Sample Graph*

Figure 3 provides another example of a typical graph structure. In this work, our hypothesis for anomaly detection is based on the number of instances of each substructure. Substructures which have a fewer number of instances are considered as more anomalous. In this particular example, substructures "X" and "I" are more anomalous as they have only one instance.

## Related Work

[Noble and Cook 2003] identify both anomalous substructures and subgraphs using a variant of the Minimum Description Length (MDL) principle. They address (1) the problem of finding unusual substructures in a given graph and (2) the problem of finding unusual subgraphs. The main insight into solving these problems is to look for structures that occur infrequently, which are roughly opposite to what are called the "best substructures". However, their methods work with only categorical attributes, whereas many datasets often contain both numerical and categorical attributes. [Chakrabarti 2004] uses the Minimum Description Length principle to detect outliers. His AUTOPART system is based on the notion that nodes with similar neighbors are clustered together, and the edges that do not belong to any structure constitute anomalies. Similarly, nodes that have many cross-connections to multiple different communities are considered not to belong to any particular cluster and thus also constitute anomalies. [Moonesinghe and Tan 2006] propose a random walk approach for outlier detection. In their approach, they calculate what they call the *normality scores* of vertices, where an anomalous vertex is the one with the lowest normality score. However, the performance of random walk technique is highly dependent on the choice of similarity measure, and does not scale well to large graphs.

The approach by [Sun and Faloutsos 2005] detects anomalies in bipartite graphs. The two main problems they address are (1) how to find the community of a given node, which is also referred as the "neighborhood" of a node, and (2) how to quantify the level of the given node to be a bridge node. [Qiu at al. 2016] use weighted bipartite graphs to detect abnormal activities in vehicle inspection stations. Their idea is to detect abnormal behaviors based on the distribution of normality scores. [Chen et al. 2010] perform community-based anomaly detection using the concept of graph representatives and community representatives. Their approach is applicable to evolutionary networks even with overlapping communities. [Eberle and Holder 2007] developed the Graph Based Anomaly (GBAD) system to detect the three structural anomalies discussed earlier. However, in their GBAD system, if the type of anomaly is not known beforehand, the user may need to run all three GBAD algorithms. For more detailed information, the reader can refer to a survey on graph based anomaly detection [Akoglu et al. 2014].

## Proposed Approach

What we are proposing in this work is a novel way to detect graph-based anomalies that improve upon existing approaches by reducing the search space, false positive rates, and time complexity. Our first proposed approach, which we call Instances-based MDL Anomaly Detection (IMAD), uses the MDL evaluation metric (which will be discussed in more detail later). Algorithm 1 presents the steps of our IMAD approach.

**Algorithm 1:** ***IMAD*** (Instances-based MDL Anomaly Detection) **Proc IMAD**

```
1: Discover the normative substructures S from Graph G which minimize $DL(S)+DL(G|S)$, DL= Description Length
2: Identify substructures S in G having least number of instances, $I_k$
3: Return all substructures having the least number of Instances.
```

Our second proposed approach, which we call Instances-based Size Anomaly Detection (ISAD), uses a size evaluation metric (again, we will discuss in more detail later). Algorithm 2 presents the steps of our ISAD approach.

**Algorithm 2:** ***ISAD*** (Instance-based Size Anomaly Detection) **Proc ISAD**

```
1: Discover the normative substructures S from Graph G which minimize size(S)+size(G|S), size= #vertices+#edges
2: Identify substructures S in G having least number of instances, $I_k$
3: Return all substructures having the least number of Instances.
```

The IMAD approach uses the MDL evaluation metric, whereas the ISAD approach uses a size evaluation metric for a graph G. Both algorithms first discover a normative substructure $S_i$ where a normative substructure S is a subgraph that has an associated description and a set of instances in the input graph, G. After which, the number of instances of the normative substructures, where an instance is an occurrence of a substructure S in a graph G, is returned. In the end, substructures having the fewest number of instances are reported.

The difference between these two algorithms lies in the evaluation metric. While we can discover anomalous substructures using either algorithm, there are pros and cons to each approach. For instance, the ISAD approach is faster because it uses a simple size evaluation metric, whereas calculating compression is slightly costlier. However, the structure of the graph may affect the discovery process. For example, if there are many overlapping substructures in a graph, the size metric may discover anomalous substructures that the MDL metric may not, and vice-versa. In addition, the MDL metric is more widely used in the literature, and has many applications in various domains. Thus, we will present results using both evaluation metrics.

## Evaluation Metrics

The hypothesis of this work is that we can use the background knowledge of evaluation metrics in order to guide the graph-based anomaly detection process. Our proposed algorithm IMAD uses the MDL evaluation metric, whereas our proposed ISAD approach uses the size evaluation metric.

### MDL Encoding of Graphs

The concept of MDL [Rissanen.J 1984] was first introduced by Jorma Rissanen. The MDL principle involves the relation between the regularity in data and the compression of data. The principle implies that whenever we are able to compress the data well, there is much regularity in the data. The concept of MDL is employed in many data mining tasks, including outlier detection, clustering, and feature selection [Charu et al. 2014].

In order to implement our approach, we will use the publicly available SUBDUE system. SUBDUE uses a model evaluation method called "Minimum Encoding", a technique derived from the MDL principle. MDL states that the best description of a data set is the one that minimizes the description length of the entire data set. In SUBDUE, the best description of the dataset is the one that minimizes DL(S)+DL(G|S), where S is the substructure, DL(S) is the number of bits (i.e., description length, or DL) required to encode S, and DL(G|S) is the length of the encoding of G after being compressed using S.

In SUBDUE, graph connectivity is represented using an adjacency matrix. So, for an adjacency matrix A:

A[i, j]=1, when there is edge between vertex i and vertex j, andA[i, j]=0, when there is no edge between vertex i and j.

- For encoding vertex labels, *vbits* are needed.
  $vbits = \lg V + V \lg l_u$ (1)

where, V is the number of vertices and $l_u$ is the number of unique labels.

- *rbits* are needed to encode the rows in the adjacency matrix.

$rbits = \lg(b+1) + \sum_{i=1}^{V} \lg(b+1) + \sum_{i=1}^{V} lg \binom{V}{K_i}$ (2)

where, $K_i$ is the number of 1's in the $i_{th}$ row and $b = \max_i K_i$.

- For encoding edges, *ebits* are needed.

$ebits = e(1 + lgl_u) + (k+1)\lg m$ (3)

where, K is the number of 1's in adjacency matrix, $m = \max_{i,j} e(i,j)$, and e(i ,j)= number of edges that are present in between i and j.

- Total number of bits for the entire graph

$Total\ bits = vbits + rbits + ebits$ (4)

As an example of MDL encoding, take the simple graph shown in Figure 4.

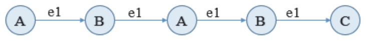

*Figure 4: Sample Graph*

From Equation (1)

$vbits = \lg V + V \lg l_u$ (1)

$= \lg 5 + 5 \lg 4$

$= 12.32192$

From Equation (2)

$$rbits = \lg(b+1) + \sum_{i=1}^{V} \lg(b+1) + \sum_{i=1}^{V} lg \binom{V}{K_i}$$

$rbits = 6 \lg 2 + \sum_{i=1}^{5} lg \binom{5}{K_i}$ = 16.28768

From Equation (3) ebits=4(1+lg 4)+(4+1) lg 1=12

Therefore, the total number of encoding bits (from Equation (4)) is 40.6096 (12.32192+16.28768+12).

## Size Metric

Size is also another evaluation metric used by SUBDUE. However, in this case, the size of an object is not computed from the description length, but the sum of the number of nodes and the number of edges:

$size(G) = \#vertices(G) + \#edges(G)$ (5)

Taking the same example shown in Figure 4, the number of vertices is 5 and the number of edges is 4, so *size(G)* = 9

## Background Knowledge to Evaluation Metrics: Rule Coverage

The hypothesis of this work is that we can use evaluation metrics as *background knowledge* in order to guide the graph-based anomaly detection process. Specifically, we propose that the use of *rule coverage* as background knowledge will improve upon our ability to discover anomalous substructures, where coverage is the percentage of the final graph to be covered by the instances of the substructure [Holder et al. 1992]

$$Coverage = 1 + \frac{1}{size(G)} \sum_{i=1}^{I} weight(i) * unique\_structure \quad (6)$$

where, "I" is the set of instances of substructure S, "unique_structure(i)" is the amount of structure in the original graph G covered by instance 'i',

and

$$weight = 1 - \frac{matchcost(S,i)}{size(i)}$$

where size(i)=#vertices(i)+#edges(i).

**Coverage: Example**

Calculation of coverage is explained by taking the same sample graph shown in Figure 4, where the number of vertices is 5 and the number of edges is 4 for a total size of 9. Using Equation (6), the values for the substructures in Figure 3 are shown in Table 1.

| **Substructure** | A | B | C | A-B | B-A | B-C |
|---|---|---|---|---|---|---|
| **Coverage Value** | 1.222 | 1.222 | **1.111** | ***1.667*** | 1.333 | 1.333 |

*Table 1 Coverage Values*

Among the substructures shown in Table 1 we can say that substructure "A-B" (value highlighted in ***bold italic***) is the one with the highest coverage value, and substructure "C" (value highlighted in **bold**) is the one with the lowest coverage value. It should be noted that for this example, the list is not exhaustive, and only substructures up to two vertices and one edge are shown, even though the largest substructure would consist of 5 vertices.

Also, "matchcost(S,i)" is the cost required to match an instance to a substructure. Specifically, it is the number of vertices and edges that would need to be changed in order to derive a matching substructure.

## Implementation

We conceptually present our proposed approach as shown in Figure 5. When the input graph is fed in to the SUBDUE system, it outputs the normative, or best, substructures. With the addition of background knowledge in the form of rule coverage, we are able to discover anomalous substructures.

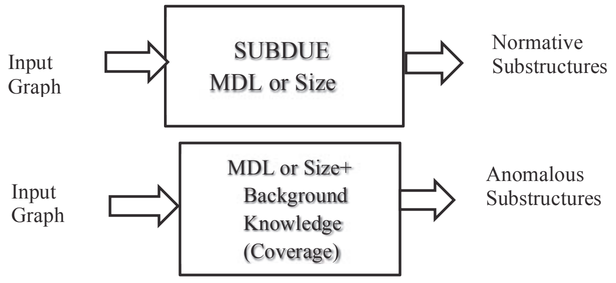

*Figure 5 Anomaly Detection- Proposed Approach*

Implementation of IMAD and ISAD in SUBDUE's framework involve biasing the substructure discovery towards anomalous substructures. The following are the steps involved in anomalous substructure discovery:

1. First we discover the evaluation values of substructures which minimize DL(S)+DL(G|S) in IMAD and size(S)+size(G|S) in ISAD.
2. We discover the coverage value of each substructure using equation (6).
3. Evaluation values are then raised to the power of coverage value with negative weight which leads to anomalous substructure discovery.

$$Value = Evaluation\ Value * ((Coverage\ Value)^{-1}) \quad (7)$$

Raising the power of coverage value with positive weights leads to substructures with more instances, but we are interested in discovering substructures with fewer instances - hence we assign a negative weight.

SUBDUE implements evaluation criteria as a means to decide which patterns are going to be chosen as normative, or best, substructures. To its available evaluation metrics, we add background knowledge in the form of rule coverage (as explaining in the previous section). By assigning negative weights to substructures that are covered, we bias the substructure discovery. Our intuition is that anomalous substructures will have the fewest number of instances that cover the entire graph – i.e., substructures with the fewest number of instances are considered anomalous. Through our proposed approach, high evaluation values will be assigned to less descriptive substructures, i.e., with fewer numbers of instances, and thus the discovery of anomalous substructures.

## Experimental Evaluations: Synthetic Datasets

We used the *subgen* tool [Eberle and Holder 2011] for our experiments. *subgen* is a synthetic generator that generates graphs using the following user-specified parameters:

- Size of the graph
- Names of vertex and edge labels
- Substructure pattern
- Connectivity value
- Overlap value

For all our experiments, we assigned a value of one to connectivity in order to keep the embedded substructure connected to the rest of the graph. We assigned a value of zero to overlap parameter. In order to test our approach, we generated anomalies of varying sizes by randomly modifying vertices and edges.

Hardware specifications for all our experiments are as follows:

- Processor Intel(R) Core(TM) i3-5005U CPU @2.00GHz 2.00 GHz, 2 Core(s), 4 Logical Processor(s)
- RAM 4.00GB
- Operating system: xubuntu 16.04

For example, we experiment using a graph of 1000 vertices and 1000 edges, with a normative pattern of 10 vertices and 10 edges, and tracked runtimes (in seconds) as well as memory (bytes) needed to detect anomalies using IMAD as well as ISAD. We observe that IMAD is ~four times faster and ISAD is ~five times faster than the existing GBAD approach. Memory consumed by IMAD and ISAD is ~three times less than the existing approach. In these experiments, we are able to detect all anomalies with 100% accuracy and zero false positives.

## Comparison of Proposed Approach with Existing Approach

In order to compare our approach against a known graph-based anomaly detection approach (in this case, GBAD), we attempt to replicate the synthetic graph input files as they are described in the work by [Eberle and Holder 2007]. We note the running times as well as the required memory of the three different approaches on each of the different types of normative patterns (Triangle, Strand, Star, Cycle) with graphs of varying sizes. We observe that IMAD is ~three times faster and ISAD is ~seven times faster than GBAD. Memory consumed by IMAD is ~two times less and ISAD is ~three times less than GBAD.

Another limitation of the GBAD system is that in order to run in polynomial time, it is not guaranteed that all of the candidate normative patterns will be generated. However, in order for GBAD to discover anomalies, it requires all candidate normative substructures to determine which substructures are closest to the normative pattern. So, there is the potential for GBAD to not discover all anomalies.

## Experimental Evaluation: KDD Cup 99 Dataset

We test our hypothesis on the KDD Cup 99 network intrusion dataset. The KDD training dataset consist of 10%

of the original dataset that has 41 features including the classification label, i.e., either normal or an attack.

| Protocol | Attack Name |
|---|---|
| TCP | neptune, guess_passwd, land, portsweep, buffer_overflow, phf, warezmaster, ipsweep, multihop, perl, back, ftp_write, satan, spy, imap, rootkit |
| UDP | teardrop, satan, nmap, rootkit |
| ICMP | normal, portsweep, ipsweep, smurf, satan, pod, nmap |

*Table 2. Attacks grouped by Protocol*

The protocols that are considered in the KDD dataset are TCP, UDP, and ICMP. In Table 2 we show the types of attack grouped by protocol. Testing our approach across all data sets, our approach is able to detect 100% of the attacks with no false positives. In Figures 6 and 7, we show the runtime and memory statistics of our approach compared against the existing GBAD approach.

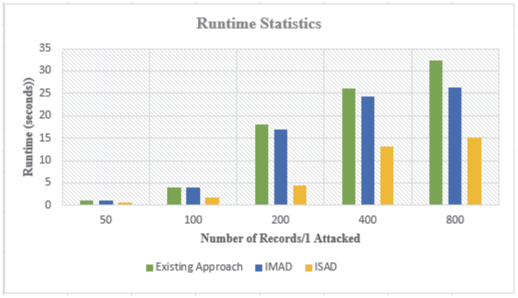

*Figure 6. Runtime Statistics*

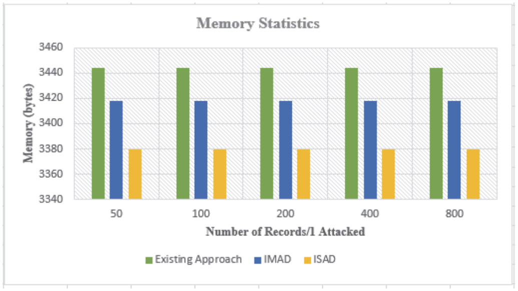

*Figure 7. Memory Statistics*

## Conclusion and Future Work

In this paper, a novel approach for graph based anomaly detection is proposed using evaluation metrics with background knowledge. We demonstrate the effectiveness of our approach on various synthetic datasets through the comparison of execution times and the memory required for our proposed approach against an existing approach. Making our proposed approach scalable to large datasets is one of our future research directions. In this paper, background knowledge in the form of rule coverage with negative weights is added to MDL and size metrics in order to improve the discovery of anomalous substructures. However, we are planning on investigating additional rules that could be added to discover other interesting substructures that are specific to the domain.

## References

Akoglu.L,Tong.H, Koutra.D 2014, *Graph-based Anomaly Detection and Description: A Survey.*

Chakrabarti.D. 2004 *AutoPart: Parameter-free graph partitioning and outlier detection.* In Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, New York, NY, USA, Springer-Verlag New York,Inc.112-124

Chen.Z, Wilson.K.A, and et al.2010, *Detecting and tracking community dynamics in evolutionary net-works.* In IEEE ICDM Workshop on Social Interactions Analysis and Services Providers, Washington, DC, USA, IEEE Computer Society, 318-327

Charu C.Aggarwal. Jiawei Han. 2014 *Mining and Using Sets of Patterns through Compression*, in Frequent Pattern Mining-Springer US

Eberle.W and Holder.L 2007, *Discovering Structural anomalies in graph based data.* Proceedings of the Seventh IEEE International Conference on Data Mining Workshops, 393-398

Eberle W and Holder.L. 2011, *Graph-Based Knowledge Discovery:Compression versus Frequency*, International Conference of the Florida AI Research Society (FLAIRS)

Eberle.W and Holder.L. 2007. *Anomaly Detection in Data Represented as Graphs*, Intelligent Data Analysis: An International Journal. Volume 11, 663-689.

Holder L.B.,Cook D.J, Bunke. 1992 H. *Fuzzy substructure discovery.* In Proceedings of the Ninth International Machine Learning Conference,218-223

Moonesinghe.H.D.K. and Tan.P. 2006 *Outlier detection using random walks.* In 18th IEEE International Conference on Tools with Artificial Intelligence, 532-539

Noble.C and Cook.D.J. 2003 *Graph-based Anomaly detection.* In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, NewYork, NY, USA, ACM, 631-636

Padmanabhan.K, Chen.Z, Sriram.L, Ramaswamy.S and Bryan Thomas.R. 2014. *Graph Based Anomaly Detection in:Practical Graph Mining with R*, CRC Press, Taylor and Francis Group,312-367

Qiu.C, Xu.H, Liu.W. 2016. *Supervision on abnormal activities in vehicle inspection service by anomaly detection in bipartite graph* In Proceedings of IEEE International Conference on Intelligent Transportation Engineering (ICITE),68-71

Rissanen.J 1984 *Universal coding, information, prediction, and estimation* IEEE Trans. Inform. Theory

Sun.J, Qu.H, Chakrabarti. .D, and Faloutsos.C.2005. *Neighborhood formation and anomaly detection in bipartite graphs.* In Proceedings of the Fifth IEEE International Conference on Data Mining, 418-425