Insider Threat Detection Using a Graph-Based Approach

William Eberle[*]
Department of Computer Science
Tennessee Technological University
Box 5101, Cookeville, TN 38505
weberle@tntech.edu

Lawrence Holder
School of Electrical Engineering & Computer Science
Washington State University
Box 642752, Pullman, WA 99164-2752
holder@wsu.edu

Jeffrey Graves
Department of Computer Science
Tennessee Technological University
Box 5101, Cookeville, TN 38505
jagraves21@tntech.edu

Abstract. We present the use of *graph-based* approaches to discovering anomalous instances of structural patterns in data that represent insider threat activity. The approaches presented search for activities that appear to match normal transactions, but in fact are structurally different. We show the usefulness of applying graph theoretic approaches to discovering suspicious insider activity in domains such as social network communications, business processes, and cybercrime. We present some performance results to show the effectiveness of our approaches, and then conclude with some ongoing research that combines numerical analysis with structure analysis, analyzes multiple normative patterns, and extends to dynamic graphs.

Keywords: anomaly detection, graph-based, insider threat.

[*] Corresponding author.

1. Introduction

The ability to mine structurally complex data has become the focus of many initiatives, ranging from business process analysis to cyber-security. Since September 11, 2001, there has been an increasing emphasis on applicable methods for analyzing everything from bank transactions to network traffic, as our nation scours individual communications for possible illegal or terrorist activity.

Protecting our nation's cyber infrastructure and securing sensitive information are critical challenges for both industry and homeland security. One of the primary concerns is the deliberate and intended actions associated with malicious exploitation, theft or destruction of data, or the compromise of networks, communications or other IT resources, of which the most harmful and difficult to detect threats are those perpetrated by an insider. However, current efforts to identify unauthorized access to information such as what is found in document control and management systems are limited in scope and capabilities. We propose to address these challenges by analyzing the relationships between entities in the data.

The ability to mine data for nefarious behavior is difficult due to the *mimicry* of the perpetrator. If a person or entity is attempting to participate in some sort of illegal activity, they will attempt to convey their actions as close to legitimate actions as possible. Recent reports have indicated that approximately 6% of revenues are lost due to fraud, and almost 60% of those fraud cases involve employees [AFCE 2006]. The Identity Theft Resource Center recently reported that 15.8 percent of security breaches so far in 2008 have come from insiders, up from 6 percent in 2007 [Foley 2008]. Various insider activities such as violations of system security policy by an authorized user, deliberate and intended actions such as malicious exploitation, theft, or destruction of data, the compromise of networks, communications, or other IT resources, and the difficulty in differentiating suspected malicious behavior from normal behavior, have threatened our nation’s security. Organizations responsible for the protection of their company’s valuable resources require the ability to mine and detect internal transactions for possible insider threats. Yet, most organizations spend considerable resources protecting their networks and information from the outside world, with little effort being applied to the threats from within.

Cybercrime is one of the leading threats to company confidential data and resources. A recent study by the Ponemon Institute surveyed 577 IT practitioners, who rated the issue of cybercrime as the top trend in their industry for the next few years, over such hot topics as cloud computing, mobile devices, and peer-to-peer sharing [Ponemon 2009]. The U.S. Department of Justice, in its Computer Crime & Intellectual Property Section, reported six incidences in the last month alone, ranging from trafficking in counterfeit computer programs to accessing government databases [USDOJ 2009]. News stories detail how insiders have bilked corporations out of millions due to their ability to access sensitive information – sometimes after they have resigned from a company that did not immediately remove their confidential access [Vijayan 2009]. There have even been studies that suggest that the economy has impacted, or will impact, the surge in cybercrime [Kirk 2009][Bush 2009].

For the last several years, companies have been analyzing their IT operations and processes for the purpose of uncovering insider threats and cybercrime. Most approaches have been either statistical in nature, leading to various data mining approaches, or a visualization of their resources where they can monitor for illegal access or entry. However, recently, the ability to mine relational data has become important for detecting *structural* patterns. The complex nature of heterogeneous data sets, such as network activity, e-mail, payroll and employee information, provides for a rich set of potentially interconnected and related data. Graph-based data mining approaches analyze data that can be represented as a graph (i.e., vertices and edges). While there are approaches for using graph-based data mining for intrusion detection, little work has been done in the area of *graph-based anomaly detection,* especially for detecting threats from insiders.

In this paper, we present our work in graph-based anomaly detection for identifying insider threats. After presenting some previous work in graph-based data mining, we define what a graph-based anomaly is, and then briefly present our algorithms. Sections 4 through 6 present our approach as applied to several insider threat scenarios. Next, we evaluate the performance of our algorithms on graphs of varying sizes and structure. Then we demonstrate some potential enhancements to our algorithms, including the handling of dynamic graphs. We then conclude with a brief discussion of future work.

2. Previous Work

Much of the information related to insider threats resides in the *relationships* among the various entities involved in an incident. Recently there has been an impetus towards analyzing multi-relational data using graph theoretic methods. Not to be confused with the mechanisms for analyzing "spatial" data, graph-based data mining approaches are an attempt at analyzing data that can be represented as a graph (i.e., vertices and edges). Yet, while there has been much written as it pertains to graph-based intrusion detection

[Staniford-Chen et al. 1996], very little research has been accomplished in the area of graph-based anomaly detection.

In 2003, Noble and Cook used the SUBDUE application to look at the problem of anomaly detection from both the anomalous substructure and anomalous subgraph perspective [Noble & Cook 2003]. They were able to provide measurements of anomalous behavior as it applied to graphs from two different perspectives. *Anomalous substructure* detection dealt with the unusual substructures that were found in an entire graph. In order to distinguish an anomalous substructure from the other substructures, they created a simple measurement whereby the value associated with a substructure indicated a degree of anomaly. They also presented the idea of *anomalous subgraph* detection which dealt with how anomalous a subgraph (i.e., a substructure that is part of a larger graph) was to other subgraphs. The idea was that subgraphs that contained many common substructures were generally less anomalous than subgraphs that contained few common substructures. In addition, they also explored the idea of conditional entropy and data regularity using network intrusion data as well as some artificially created data.

Several approaches employ statistical measures to identify individual node or edge anomalies. Lin and Chalupsky [Lin & Chalupsky 2003] took the approach of applying what they called rarity measurements to the discovery of unusual links within a graph. The AutoPart system presented a non-parametric approach to finding outliers in graph-based data [Chakrabarti 2004]. Part of this approach was to look for outliers by analyzing how edges that were removed from the overall structure affected the minimum descriptive length (MDL) of the graph [Rissanen 1989]. The idea of entropy was used by Shetty and Adibi [Shetty & Adibi 2005] in their analysis of the famous Enron e-mail data set. Using bipartite graphs, Sun et al. [Sun et al. 2005] presented a model for scoring the normality of nodes as they relate to other nodes. Rattigan and Jensen went after anomalous links using a statistical approach [Rattigan & Jensen 2005].

In Priebe et al.'s work, they used what are called "scan statistics" on a graph of the e-mail data that is represented as a time series [Priebe et al. 2005]. While their approach detects statistically significant events (excessive activity), without further analysis, they are unable to determine whether the events are relevant (like insider trading). Martin et al. examined what they called "behavioral features" of a particular user's network traffic in order to discover abnormal activity [Martin et al. 2005]. Through various clustering approaches, and comparisons to methods such as Support Vector Machines and Naives Bayes Classification, they group sets of users into single behavioral models. Diesner et al. applied various network analytic techniques in their exploration of the structural properties of the Enron network. They used various graph structural metrics, such as betweenness centrality, eigenvectors and total degree in order to identify key players across time [Diesner & Carley 2005]. In 2007, Kurucz et al. used hierarchical spectral clustering to evaluate weighted call graphs [Kurcz et al. 2007]. They analyzed several heuristic approaches using phone calls made over an eight-month period. However, their purpose was not to expose anomalies in phone traffic, but instead to address the issues associated with processing large graphs. In Swayne et al's work, they used graph techniques to explore AT&T phone records [Swayne et al. 2003]. While their approach was able to provide for the analysis of phone traffic, it was entirely based upon graph visualization, rather than any graph theoretic approaches. In fact, when it comes to generating graphs of information, much research has dealt with only the visual aspects of what is represented, rather than the structural aspects of the graphs themselves.

The advantage of graph-based anomaly detection is that the relationships between elements can be analyzed, as opposed to just the data values themselves, for structural oddities in what could be a complex, rich set of information. Furthermore, our approach identifies the structural context in which the anomaly occurs rather than just the particular nodes or links that are anomalous.

3. Graph-based Anomaly Detection

3.1 Definition

The idea behind the approach used in this work is to find anomalies in graph-based data where the anomalous substructure in a graph is part of (or attached to or missing from) a *normative substructure*.

Definition: *A graph substructure S' is anomalous if it is not isomorphic to the graph's normative substructure S, but is isomorphic to S within X%.*

X signifies the percentage of vertices and edges in *S'* that would need to be changed in order for *S'* to be isomorphic to *S*. The importance of this definition lies in its relationship to any deceptive practices that are intended to illegally obtain or hide information. The United Nations Office on Drugs and Crime states the first fundamental law of money laundering as "The more successful money-laundering apparatus is in imitating the patterns and behavior of legitimate transactions, the less the likelihood of it being exposed" [Hampton and Levi 2009].

There are three general *categories of anomalies*: insertions, modifications and deletions. Insertions would constitute the presence of unexpected vertices or edges. Modifications would consist of unexpected labels on vertices or edges. Deletions would constitute the unexpected absence of vertices or edges.

3.2 Approaches

Most anomaly detection methods use a supervised approach, which requires some sort of baseline of information from which comparisons or training can be performed. In general, if one has an idea what is normal behavior, deviations from that behavior could constitute an anomaly. However, the issue with those approaches is that one has to have the data in advance in order to train the system, and the data has to already be labeled (e.g., normal employee transaction versus threatening insider activity).

GBAD (Graph-based Anomaly Detection) [Eberle and Holder 2007] is an *unsupervised* approach, based upon the SUBDUE graph-based knowledge discovery method [Cook and Holder 2000]. Using a greedy beam search and Minimum Description Length (MDL) heuristic [Rissanen 1989], each of the three anomaly detection algorithms in GBAD uses SUBDUE to discover the best-compressing substructure, or normative pattern, in an input graph. In our implementation, the MDL approach is used to determine the best substructure(s) as the one that minimizes the following:

$$M(S,G) = DL(G \mid S) + DL(S)$$

where G is the entire graph, S is the substructure, $DL(G|S)$ is the description length of G after compressing it using S, and $DL(S)$ is the description length of the substructure. The description length *DL(G)* of a graph *G* is the minimum number of bits necessary to describe the graph *G*.

We have developed three separate algorithms: GBAD-MDL, GBAD-P and GBAD-MPS. Each of these approaches is intended to discover one of the three possible graph-based anomaly types as set forth earlier. The following is a brief summary of each of the algorithms, along with some simple examples to help explain their usage. The reader should refer to [Eberle and Holder 2007] for a more detailed description of the actual algorithms.

3.1.1 Information Theoretic Algorithm (GBAD-MDL)

The GBAD-MDL algorithm uses the Minimum Description Length (MDL) heuristic to discover the best-compressing (normative) substructure in a graph, and then subsequently examines all of the instances of that substructure that "look similar" to that pattern – or more precisely, are *modifications* to the normative pattern. In Noble and Cook's work on graph-based anomaly detection [Noble & Cook 2003], they presented a similarly structured example (albeit with different labels) to the one shown in Figure 1.

In this example, the normal business process involves Sales sending an order to the Dispatcher, the Dispatcher verifying the order and sending in onto the Warehouse, and the Warehouse confirming the fulfillment of the order with Sales. When applying the GBAD-MDL algorithm to this example, the circled substructure in Figure 1 is reported as being anomalous. In this case, there are three entities communicating for each order, but Accounts is handling the order instead of Sales - going outside the normal process. With Noble and Cook's approach, the "Accounts" vertex would have correctly been shown to be the anomaly, but the importance of this new approach is that a larger context is provided regarding its associated substructure. In other words, not only are we providing the anomaly, but we are also presenting the context of that anomaly within the graph (the individual anomaly within the instance is in **bold**.)

3.1.2 Probabilistic Algorithm (GBAD-P)

The GBAD-P algorithm also uses the MDL evaluation technique to discover the best-compressing (normative) substructure in a graph, but instead of examining all instances for similarity, this approach examines all extensions (or *insertions*) to the normative substructure with the lowest probability. The difference between the algorithms is that GBAD-MDL is looking at instances of substructures with the same characteristics (e.g., size), whereas GBAD-P is examining the probability of extensions to the normative pattern to determine if there is an instance that includes edges and vertices that are probabilistically less likely than other possible extensions. Taking the business process example again, Figure 2 shows the process flow between a warehouse (W), dispatcher (D), accounting (A) and the customer (C).

In this example, the normal process involves a communication chain between Sales, Warehouse and Dispatcher, with the order confirmation being conveyed by the Dispatcher to the Customer. After the first iteration of the GBAD-P algorithm, the boxed instance in Figure 2 is one of the instances of the normative substructure. Then, on the second iteration, extensions are evaluated, and the circled instance is the resulting anomalous substructure. In this example, the Dispatcher is communicating with Accounts when it should have been the Customer. Again, the edge and vertex (shown in

bold) are labeled as the actual anomaly, but the entire anomalous substructure is output to provide additional context for possible analysis.

3.1.3 Maximum Partial Substructure Algorithm (GBAD-MPS)

The GBAD-MPS algorithm again uses the MDL approach to discover the best-compressing (normative) substructure in a graph, then it examines all of the instances of parent (or ancestral) substructures that are missing various edges and vertices (i.e., *deletions*). The value associated with the parent instances represents the cost of transformation (i.e., how much change would have to take place for the instance to match the normative substructure). Thus, the instance with the lowest cost transformation is considered the anomaly, as it is closest (maximum) to the normative substructure without being included on the normative substructure's instance list. If more than one instance has the same value, the frequency of the instance's structure will be used to break the tie if possible. Consider the slightly more complex graph of a business process, involving multiple transactions that are linked together by common entities, as shown in Figure 3.

In this example, the normative pattern in the process is a Sales person communicating with the Warehouse and a Customer, and the Warehouse corresponding with a Dispatcher. Suppose we take one of the instances of the normative pattern (shown in the box), and remove an edge and its associated vertex (shown in the circle). When applying GBAD-MPS to that modified graph, an anomalous substructure, similar to the normative pattern, is discovered, where the Customer entity is missing along with the "note" link from Sales.

4. Experiments on Communication-based Social Networks

Social networks consist of nodes representing individuals and edges representing various relationships (e.g., friendship) between individuals. Discovering patterns and anomalies in social networks is of particular interest. Here, we focus on social networks where the relationship links represent communication between individuals. One communication-based social network that has garnered much interest is based on e-mail traffic. As the New York Times reported, it "... is a potential treasure trove for investigators monitoring suspected terrorists and other criminals…" [Kolata 2005]. Up until recently, researchers have struggled with being able to obtain corporate e-mail due to the obvious restrictions placed on releasing what could be sensitive information. However, with the Federal Energy Regulatory Commission publication of the e-mail associated with the infamous Enron Corporation, researchers now have access to a rich data set of correspondences between management, lawyers and traders, many of whom were directly involved in the scandal.

Another domain that has been the subject of data mining activities involves the analysis of phone calls. Organizations such as the National Security Agency (NSA) have spent the last several years collecting suspicious phone calls and storing them in a database [Cauley 2006]. The significance of being able to peruse phone call information lies in the fact that an analyst can see who called whom, when they talked, for how long they talked, and the location of both parties. In the case of cell-phone calls, one can also ascertain the specific global position of two entities. Such information has been useful to not only general data mining research, but more specifically, research in diverse areas such as marketing, terrorist monitoring, and social network analysis.

The following sections explore the detection of anomalies in the social networks that can be found in both e-mail communications and cell-phone traffic.

4.1. E-mail Communications

One of the more recent domains that have become publicly available is the data set of e-mails between employees from the Enron Corporation. The Enron e-mail dataset consists of not only messages, but also employee information such as their full name and work title. By limiting our graph to the Enron employees and their correspondences, we are able to not only create a "social network", but also discover anomalous behaviors among *classes* of individuals [Eberle & Holder 2009]. Thus, we generated graphs based upon the social aspect and company position of employees that start a "chain" of e-mails, where a chain consists of the originating e-mail and any subsequent replies or forwards to that corresponding e-mail. Each graph consists of the substructures shown in Figure 4.

In this representation, a graph consists of individual, disconnected substructures that represent the "flow" of each e-mail that originates from someone with a specified employment title (e.g., Director). An e-mail can be sent by one or more TRANSFERs, from a SENDER (and their corresponding TITLE), to a RECEIVER (with the TITLE of the e-mail),, and can either be sent back (as a reply or forward, called a STATE), with a unique message identifier (called a MID), or forwarded/replied on to other entities (via a specific METHOD). There is no limit to the number of times a message can be replied/forwarded.

There are many different employee titles within Enron (i.e., Managers, Directors, CEOs, etc.), and each of the GBAD algorithms were able to show different structural anomalies in the chains of e-mails that originated along people’s company titles. For instance, running GBAD on the graph that consists of e-mails originating from Directors, the anomalous instance shown in Figure 5, visualized using the GraphViz tool [www.graphviz.org], is discovered.

This anomalous instance consists of a message being sent from a Director to an Employee (i.e., non-management personnel), that was then forwarded to another non-management Employee. What is interesting about this anomaly is that the data set consists of many e-mails that are sent "TO" "Employee"s from "Director"s, but this is the only situation where the Employee FORWARDed the e-mail on to another “Employee”, who was not privy to the original e-mail. Specifically, the e-mail started with Hyatt (director) regarding "Oasis Dairy Farms Judgement", who sent it to Watson (employee), who then forwarded it to Blair (employee).

While applying GBAD-MPS and GBAD-P to the graph of e-mails originating from personnel with the title of “Trader” does not produce any significant anomalies, the GBAD-MDL algorithm does produce two anomalous instances. Figure 6 shows two situations where a Trader was involved in a chain of e-mails that resulted in correspondences to a CEO and a President (respectively) that were not normal.

In terms of the first anomalous instance, shown in Figure 6, from an e-mail entitled "Financial Disclosure of $1.2 Billion Equity Adjustment", there are only 4 e-mails that are sent to CEOs. But, this is the only example of an e-mail being sent TO a CEO - the other 3 e-mails are CCed to the CEO. In the case of the second anomalous instance shown in Figure 6, an e-mail entitled "Fastow Rumor", this is the only time that an e-mail is sent by a Trader to a President.

4.2. Cell-Phone Traffic

As part of the 2008 IEEE Symposium on Visual Analytics Science and Technology [VAST 2008], we decided to apply our approaches to one of the mini-challenges that deals with cell-phone traffic [Eberle & Holder 2008]. While the goal of the challenge is to target new visual analytics approaches, it is still possible to apply these graph-based anomaly detection algorithms to the same data sets. One of the data sets consists of cell-phone traffic between inhabitants of the fictitious island of Isla Del Sueño. The data consists of 9,834 cell-phone calls between 400 people over a 10-day period. The challenge is to describe the social network of a religious group headed by Ferdinando Cattalano and how it changes over 10 days. A graph of the general structure of this cell-phone traffic is represented as shown in Figure 7.

In order to answer the challenge, we decided to focus on the social interactions by creating a graph of the social network that indicated for a particular day, who called who. Based upon all of the information that was provided with the challenge, we made the following assumptions about this particular data set:

- The person with an ID of 200 is Ferdinando Catalano.
- Anyone that Ferdinando Catalano calls (or that calls him) is in his “inner circle”.

- The person with an ID of 5 is Estaban Catalano, Ferdinando's brother, as he is called the most.
- The person with an ID of 1 is David Vidro, as he talks the most to the others that Ferdinando talks to.

Starting with these simple assumptions, and a graph that consisted of vertices for each unique ID with links between the vertices if there was a conversation on that day, we were able to create a simple visualization of Ferdinando's "inner circle" social network structure (or Catalano/Vidro social structure) over the 10 days that data was generated

Figure 8 was rendered using AT&T's graph visualization program GraphViz. This visualization shows the graph structure of interactions between people in 200's (i.e., Ferdinando Catalano's) inner circle (i.e., 200, 1, 2, 3, 5, 97 and 137), the normative pattern within the graph, and the anomalous patterns in terms of the normative pattern.

In Figure 8, the substructure in the upper-right shows the normative pattern that was discovered in this graph. The substructure consisting of an edge to the vertex labeled "96" is the anomaly that was discovered by the GBAD-P algorithm, which analyzes a graph for anomalous extensions to the normative pattern. In this case, the fact that 5 called 97 was anomalous, when compared to other instances of what was the normal social structure. The third substructure from the right across the top is the anomaly that was discovered by the GBAD-MPS algorithm, which analyzes a graph for missing structure. In this case, the fact that 200 did not talk to 3 on that day is considered anomalous.

Looking at the visualization shown in Figure 8 of the Catalano/Vidro calling-history, we are able to make several interesting observations about his social network:

- There are only 9 substructures in the graph. This is due to the fact that on Day 8, nobody in 200's inner circle talked to each other. In other words, there were no calls between 1, 2, 3, 5, 97, 137 or 200 on that day.
- Catalano/Vidro's "normative" social pattern only occurs on Days 1, 2, 4, 5 and 7.
- Nobody from the "normative inner circle" (i.e., 1, 2, 3, 5 and 200), communicates with anyone else in the normative circle after Day 7. Could it be that Ferdinando sent them to the United States at this point?
- 200 communicates with both 97 and 137 on Day 9, and just 97 on Day 10.
- 200 is involved in an "inner-circle" conversation on every day (except Day 8).

We also played with several other variants of the graph, including the "directedness" of the graph. While we chose an undirected graph for all of the results shown above (because we considered a conversation between two people to be a two-way communication), we also looked at a directed version of the graph, where the edge between two vertices was directed going from the person who called to the person who was being called. When we did that, we noticed that 97 and 137 are never called by 1, 2, 3 and 5 – and they only call 5 and 200.

5. Experiments on Business Processes

For years, companies have been analyzing their business processes for the purposes of streamlining operations, discovering wasteful overhead, overcoming inefficiencies in production, etc. However, there have also been several efforts applied towards analyzing business processes for fraud detection, which has led to an increase in pertinent data mining activity. Most of these approaches have dealt with the visualization of business processes, such as VisImpact [Hao 2006]. Some approaches have used data/audit logs that are collected by a company, in order to generate fraud alerts in near real-time. While there are approaches for using graph-based data mining on domains like intrusion detection [Staniford-Chen 1996], little work has been done in the area of *graph-based anomaly detection*, especially for application to business processes, such as in document control and management systems.

In order to perform a systematic evaluation of the Graph-Based Anomaly Detection (GBAD) approach for identifying anomalies, or insider threats, in business transactions or processes, we used the OMNeT++ discrete event simulator [OMNeT] to model transactions and processes, generate transaction and process data, represent the data in graph form, and then analyze the graphs using GBAD [Eberle & Holder CIDM 2009]. This process has two main benefits. First, we can model many different types of transactions with known structure and known anomalies, which allows us to easily verify GBAD's ability to detect these anomalies. Second, the OMNeT++ framework can be used to model real business processes to further evaluate the real-world applicability of the GBAD approach. Here we give a brief introduction of this process on a simple business transaction example, followed by a more complex example representing a known business process.

5.1. Order Processing

Consider the order-fulfillment process depicted in Figure 9. The process is initiated by the Customer placing an Order, which is sent to the Sales department. The Sales department sends an Order Acknowledgement back to the Customer and sends an Internal Order to the Warehouse. Once the Warehouse ships the order, they send a Delivery Note to the Customer. One possible anomaly in this process is when someone in the Sales department copies the Order to an Unknown entity, perhaps to leak insider information to a competitor about the order.

We first define the structure of this network using OMNeT++'s Network Description Language and then define the flow of information within C++ modules for each node. After receiving an Order message, the Sales module waits 10-60 seconds and then sends an Order Acknowledgement message to the Customer module, sends an Internal Order message to the Warehouse module, and with a Bernoulli probability of 0.001 (as defined in the omnetpp.ini file) sends an Order message to the Unknown module.

Figure 10 shows a portion of the output from the order fulfillment simulation. In addition to the logging information produced by OMNeT++, the figure also shows the GBAD-related messages printed from each module describing order-related messages as they are sent and received by the modules. It is this information we use to construct graphs of the ordering process.

For the experiment depicted in Figure 9, representing the processing flow of 1,000 orders, we generated a graph of approximately 3,000 vertices and 4,000 edges. From this graph, GBAD is able to successfully discover, with no false-positives, the known anomaly shown with dotted lines and a larger font in Figure 11, alongside two other non-anomalous instances of the normative pattern.

5.2. Passport Applications

Another type of process for which we have applied our approaches, motivated by two real world sources of information, deals with document processing [Eberle & Holder CIDM 2009]. One source is the incidents reported in the CERT Insider Threat documents [Randazzo 2004][Kowalski 2008] that involve privacy violations in a government identification card processing organization and fraud in an insurance claim processing organization. Another source, for which our model directly simulates, is based on the process flow associated with a passport application [Chun 2008]. The outline of this process flow, depicted in Figure 12, is as follows:

1. The applicant submits a request to the frontline staff of the organization.
2. The frontline staff creates a case in the organization's database and then submits the case to the approval officer.
3. The approval officer reviews the case in the database and then assigns the case to one of the case officers. By default, there are three case officers in the organization.
4. The assigned case officer reviews the case. The assigned case officer may request additional information from the applicant, which is submitted to the frontline staff and then forwarded to the assigned case officer. The assigned case officer updates the case in the database based on this new information. The assigned case officer may also discuss the case with one or more of the other case officers, who may review the case in the database in order to comment on the case. Ultimately, the

assigned case officer will recommend to accept or reject the case. This recommendation is recorded in the database and sent to the approval officer.

5. Upon receiving the recommendation from the assigned case officer, the approval officer will make a final decision to accept or reject the case. This decision is recorded in the database and sent to both the frontline staff and the applicant.
6. Finally, upon receiving the final decision, the frontline staff archives the case in the database.

There are several scenarios where potential insider threat anomalies might occur, including:

1. Frontline staff performing a Review Case on the database (e.g., invasion of privacy).
2. Frontline staff submits case directly to a case officer (bypassing the approval officer).
3. Frontline staff recommends or decides case.
4. Approval officer overrides accept/reject recommendation from assigned case officer.
5. Unassigned case officer updates or recommends case.
6. Applicant communicates with the approval officer or a case officer.
7. Unassigned case officer communicates with applicant.
8. Database access from an external source or after hours.

Representing the processing of 1,000 passport applications, we generated a graph of approximately 5,000 vertices and 13,000 edges, and proceeded to replicate some of the insider threat scenarios described above.

First, we randomly inserted an example that represents Scenario 1. While the GBAD-MDL and GBAD-MPS algorithms do not discover any anomalous structures, GBAD-P is able to successfully discover the only anomalous case out of 1,000 where a frontline staffer was performing a review of a case – a clear violation of their duties. Figure 13 shows the normative pattern and the anomalous edge “ReviewCase” between the “FrontlineStaff” node and the “Database” node.

The actual anomaly in Figure 13 is shown with a bolded edge and larger label font. Also, while not shown here, this same structural anomaly can be found in scenarios 3 and 6. Scenario 3 consists of an extra edge (“RecommendAcceptCase”) going from the “FrontlineStaff” node to the “Database” node, and as such is only different from Scenario 1 by the label on the edge. Scenario 6 consists of an extra edge between the “Applicant” node and the “ApprovalOfficer” (or “CaseOfficer”) node, which is structurally identical to the other two scenarios – an unexpected edge between two expected vertices.

For Scenario 2, we randomly inserted three examples where a frontline staffer submitted a case directly to a case officer, instead of sending it to the approval officer. In this case, GBAD-P and GBAD-MDL do not uncover any anomalous structures, whereas GBAD-MPS is able to successfully discover all three instances where the frontline staffer did not submit the case to the approval officer. Figure 14 shows the normative pattern and the missing “SubmitCase” edge between “FrontlineStaff” and “ApprovalOfficer”, the missing second “ReviewCase” edge between “ApprovalOfficer” and “Database”, and the missing “AssignCase” edge between “ApprovalOfficer” and “CaseOfficer”.

The actual anomalies in Figure 14 are shown with a larger label font and a dashed edge, indicating their absence from the graph.

For Scenario 4, we randomly modified three examples by changing the recommendation that the "CaseOfficer" sends to the "ApprovalOfficer". In one example, the "CaseOfficer" recommends to accept the case, and the recommendation from the "ApprovalOfficer" is changed to rejecting the case, and in the other two examples the reverse is implemented. For this example, GBAD-MDL and GBAD-MPS do not find any anomalies, and GBAD-P only discovers one of the anomalous examples (where the "CaseOfficer" recommends to reject the case but the "ApprovalOfficer" decides to accept the case. Figure 15 shows the normative pattern and the anomalous structures from this example.

When we have GBAD report on the top two most anomalous substructures, instances of that type (reject changed to accept) are discovered, but we are still missing the first anomalous example (accept changed to reject). The issue is that we are dealing with multiple normative patterns (i.e., multiple substructures that can be considered normative in the entire graph.) In this case, there are two basic normative patterns – one where the "ApprovalOfficer" and "CaseOfficer" both accept a case, and one where the "ApprovalOfficer" and "CaseOfficer" both reject a case. However, when we modified the GBAD-P algorithm to analyze the top *N* normative patterns, both of the examples where the "CaseOfficer" recommends rejecting the case but the "ApprovalOfficer" accepts the case, are reported as the most anomalous examples, and the next most anomalous instance reported is the other anomalous example. Also, no other substructures were reported as anomalous along with these top three anomalies (i.e., no false positives). This highlights a general issue with GBAD in regards to finding anomalies when multiple normative patterns exist in the graph. We discuss this issue in more detail in Section 8.

For Scenario 5, we randomly inserted into two examples the situation where a “CaseOfficer” recommends to accept a case for which they were not assigned. In this scenario, GBAD-MDL does not report any anomalies, while both GBAD-MPS and GBAD-P each discover both anomalous instances. GBAD-MPS discovers the anomalies because the “CaseOfficer” has assigned himself to the case without any corresponding recommendation back to the “ApprovalOfficer” or “Database”, while GBAD-P uncovers the extra “CaseOfficer” and his unauthorized assignment to the case. Figure 16 shows the normative pattern and the anomalous structures from one of these examples. Also, while not shown, this same structural anomaly can be found in Scenario 7. Scenario 7 consists of an extra edge going from the unauthorized “CaseOfficer” node to the “Customer” node, and as such is only different from Scenario 5 by the label on the edge and the targeted node.

The added aspect of time, found in Scenario 8, involves the analysis of numerical attributes and how to incorporate them into the graph structure. This will be discussed further in Section 8, when we incorporate numerical analysis with the structural analysis.

6. Experiments on Cybercrime

An example of cybercrime is the leaking of information by employees with access to confidential and sensitive information. As part of the 2009 IEEE Symposium on Visual

Analytics Science and Technology [VAST 2009], we again applied our approaches to one of their mini challenges [Eberle Holder Graves 2009]. Whereas the 2008 challenge mentioned earlier focused on cell-phone traffic, each of the 2009 mini-challenges consists of various aspects of a fictional insider threat, based upon the leaking of information. The goal of these challenges is to allow contestants to apply various visual analysis techniques so as to discover the spy and their associated actions.

Again, while our GBAD approaches are not "visually based", we chose to apply our algorithms to the mini-challenge that consists of badge and network IP traffic. The data set is comprised of employee "badge swipes" during the month of January in 2008, and the IP log consists of all network activity to and from the facility. One of the goals of this mini-challenge was to determine which computers the "spy" used to send the sensitive information.

We can separate the cybercrime discovery process into three separate tasks:

1. Discover the anomalous network activity,
2. Create *targeted* graphs for just those days and people that might be involved in the anomalous activity, and
3. Use GBAD to discover which employees participate in anomalous activity.

The first stage of this process is to discover the network activity that is unusual – or the source of illegal transmissions. Rather than apply a graph-based approach to the discovery of what would be numerical/statistical anomalies (i.e., non-structural anomalies), we can do a simple analysis of the actual records. Sorting the IP logs by amount of traffic, one discovers that the top five transmissions are all to the same destination IP, 100.59.151.133 on port 8080:

```
…
Synthetic Data   37.170.100.31    2008-01-15T17:03  100.59.151.133   8080    9513313     14324
Synthetic Data   37.170.100.20    2008-01-24T17:07  100.59.151.133   8080    9732417     42347
Synthetic Data   37.170.100.13    2008-01-22T08:50  100.59.151.133   8080    9984318     42231
Synthetic Data   37.170.100.56    2008-01-29T15:41  100.59.151.133   8080   10024754     29565
Synthetic Data   37.170.100.8     2008-01-31T16:02  100.59.151.133   8080   13687307    485421
```

In the IP log file, the first column is the type of data, the second column is the source IP, the third column is the date and time, the fourth column is the destination IP, the fifth column is the destination port, the sixth column is the size of the transmission, and the last column is the size of the response record. In fact, 17 of the 32 highest-transmission records have this same destination IP - clearly an unusual volume of traffic to a single, external destination. In addition, with our graph-based approach, we can *verify* the anomalousness of the traffic based upon the relationship of the activity within the graph. For example, knowing that employee 31's computer is one of the computers that sent the supposedly illegal transmissions (see the top record above), we can analyze the subgraph of that employee's activity on that day.

In order to discover an insider committing this form of cybercrime, we make two reasonable assumptions:

1. The insider never uses their own computer (for fear of their actions being traced back to them), and
2. The insider only uses someone else's computer when they are in the classified area (as that is the only time we know that they are not in their office).

Using these two assumptions, we can then focus on the generation of graphs that (1) exclude people whose computer was compromised from being considered as suspects, and (2) reduce the graph search space to only those days where the illicit transmissions took place. In this data set, 10 employees are removed from being considered as suspects, and only the activity of other employees during the anomalous network activity are represented in the graph. This will enable us to analyze abnormal structure in the graph during the times of the crimes.

So first we create graphs consisting of subgraphs that represent employee movements for *each* targeted day (i.e., the days when the illicit transmissions took place), as well as graphs that represent the movements for each employee over *all* of the targeted days. Each subgraph will contain a “backbone” of movement vertices. Attached to the movement vertices will be two vertices representing where the person was before entering the current location and the current location (i.e., outside, building, classified). The edges will be labeled start and end, respectively. Then, if network traffic is sent before the person moves again, a network vertex will be created and linked to the movement vertex via a sends edge. The network vertex will also be linked to a vertex with a numerical label, representing how many messages are sent before the next movement occurs. The result is a graph topological representation as shown in Figure 17.

In the partial example shown in Figure 18, a person enters from the outside, transfers some data across the network, and then moves into the classified area.

We created a tool to process the comma-delimited proxy log and IP log files and output graph files for use with GBAD. Once the graph files are created, GBAD can then be used to obtain (1) the normative pattern discovered in the specified graph input file and (2) the top-N most anomalous patterns.

Using this graph representation, GBAD discovers the normative pattern shown in Figure 19.

After uncovering the normative pattern, GBAD can then use its three algorithms to discover all of the possible structural changes that can exist in a graph (i.e., modification, deletions, and insertions).

The VAST data set consists of the activities of 60 employees at an embassy over the month of January in 2008. As stated earlier, there are 17 transmissions to the suspect IP. Based upon our first assumption, we can remove 10 employees from the list of suspects (some employees' computers were compromised more than once). We can also reduce our data set down to just the days where the anomalous transmissions took place, which consists of 8 of the 31 available days worth of information. This subset of the data is then the baseline for our GBAD analysis.

Using these targeted graphs (8 day graphs and 50 people graphs), we ran the GBAD algorithms using default parameter settings, where it would report only the most anomalous instances, rather than the top-K instances. On the graphs that represent individual people and their movements and network activities across all targeted days, the GBAD-MDL algorithm discovers 12 employees as having anomalous movements and activities, and the GBAD-MPS algorithm reports 8 employees as anomalous. On the graphs that represent all movements and activities for each targeted day, GBAD-MDL reports 6 employees as anomalous while GBAD-MPS reports 2 employees. However, there is an interesting commonality across all four experiments. If you take the overlap (intersection) between them, in other words which employees are reported in ALL of the experiments, one discovers that there are only 2 employees that are very suspicious:

employee 49 and employee 30.

We can further distinguish a difference between these two employees by analyzing the graphs and GBAD results. From the GBAD results, employee 30 is reported as the most anomalous (score-wise) on 6 of the 8 days, with employee 49 being the most anomalous on the other 2. Also, employee 30 is the only employee with the structural anomaly shown in Figure 20.

In Figure 20 (only the parts of the graph necessary for this observation are shown), one will notice that the employee initially moves from the outside into the building. However, their next move is from the classified area into the building – with no movement into the classified area before that. This is called "piggybacking", where an employee does not use their badge, but instead follows on the heels of another employee. Employee 30 is not only the only employee to piggyback into the classified area, but they do it several times. Perhaps their intent is to gather classified information without a trace of ever entering the area. Unfortunately (for them), they had to badge-out of the area – resulting in a structural anomaly in their movement.

It should also be noted that the GBAD-P algorithm does not report any significant movement or activities as anomalous, but does report the differences in network traffic sizes. In addition, it is interesting to note that all of the anomalous activity takes place on Tuesdays or Thursdays. Future work in anomaly detection could detect structural patterns in the anomalies themselves.

7. Performance

Of course, the ability to discover anomalies is critical to the success of any anomaly detection system. However, in order to be useful as a real-world application, the performance of the algorithms must be viable for real-time analysis.

In previous work [Eberle and Holder 2007], we demonstrated the accuracy of GBAD on real-world and synthetic graphs of varying sizes and density (number of vertices and edges). Recently, we have conducted various experiments with GBAD to identify areas for further performance enhancements in regards to scalability by examining the relationship between accuracy and efficiency as it relates to the graph topology. These efforts have focused on identifying the performance of GBAD in terms of:

1 Measuring the effects on accuracy and efficiency as a function of the size of the input graph.
2 Measuring the effects on accuracy and efficiency as a function of the size of the normative pattern.

For the first set of experiments, we tested various graph input sizes (number of vertices). As an example of our results, Figure 21 shows a log-log plot of the running time of GBAD as a function of the graph size where the normative pattern has 10 vertices and 9 edges. Here we see that all three versions of GBAD run in time polynomial (degree just below 2.0) in the graph size. This particular plot is for disconnected graphs, where each instance of the normative pattern appears in a separate graph and some instances are randomly selected for the introduction of an anomaly (insertion, deletion or modification). In these experiments, all anomalies are found with no false positives.

Figure 22 shows a similar plot for connected graphs; that is, all instances of the normative pattern appear in one connected graph. Here again we see that all three versions of GBAD run in time polynomial in the number of vertices in the graph. While all anomalies are found, there were some false positives found in the smaller graphs (3 at size 100, 2 at size 200, and 2 at size 400). The main reason for this is that fewer random edges are needed to connect the smaller graphs, and therefore these edges appear anomalous. As the graph grows in size, the random edges appear with more frequency than the anomalous edges, and therefore appear less anomalous. Overall, these results show that GBAD scales well with graph size in both efficiency and accuracy, as GBAD's running time is low-degree polynomial in the size of the input graph.

We then evaluated GBAD's performance as the size of the normative pattern increases. We initially identified performance bottlenecks. For example, when GBAD looked for anomalies that were small modifications to the normative pattern, after finding the normative pattern, GBAD had to find close matches to the pattern (e.g., a graph with a vertex or edge label changed). As the normative pattern's size increased, finding close matches became increasingly costly in both memory and running time. Basically, the bottleneck was due to the graph isomorphism test being performed on subgraphs of the normative pattern as GBAD expanded these subgraphs to become the same size as the normative pattern - yet were potentially anomalous. In response to this, we enhanced the

GBAD algorithms with a more constrained matching technique that takes advantage of the fact that a large portion of the normative pattern and potential anomaly will match exactly, and that the graph matching process should not try to undo this portion of the match. Adding this constraint alone results in a ~40% speedup in running time.

For the second set of experiments, we analyzed GBAD's performance as it relates to both the size of the input graph and the size of the normative pattern. Before the enhancement, we noted a sharp increase in runtime as the size of the normative pattern reached a certain percentage size of the input graph. However, by implementing the enhancement described above, we are able to alleviate the increased runtime, in many cases achieving two orders of magnitude speedup over the previous version, without sacrificing accuracy. For example, the Table 1 shows results for increasing sizes of normative patterns for an input graph of size 1700 vertices. In this table, we see significant improvement at higher normative pattern sizes for modification-based anomalies (anomalies that are the same size as the normative pattern, but with a small difference). In the case of the 40 vertex normative pattern, the enhancement allowed GBAD to complete where it had not been able to previously. This enhancement has also allowed processing of graphs up to 60,000 vertices in less than 5 minutes.

However, there is still room for improvement. Figure 23 shows the running time of GBAD with increasing input graph size for sparse graphs. The log-log plot indicates an approximately quadratic order of growth. Experiments on dense graphs show an even higher order of growth. Most of this time is spent finding the normative patterns, which is currently done using the SUBDUE graph-based knowledge discovery approach [Cook and Holder 1998]. SUBDUE has several parameters that limit the amount of computation (search) that is performed, and we are exploring better settings for these parameters that reduce running time, but still find the correct normative pattern. Another approach would be to employ a frequent subgraph discovery algorithm, like FSG [Kuramochi and Karypis 2004] or gSpan [Yan and Han 2002], which tend to be more efficient than SUBDUE since they search for frequent subgraphs; whereas, SUBDUE searches for highly-compressing subgraphs.

On a more positive note, results of GBAD on a fixed-size sparse graph with an increasing-sized normative pattern show fairly constant running times. Figure 24 shows the running time of GBAD on a sparse graph of 60,000 vertices as the size of the normative pattern increases. In sparse graphs, GBAD's running time is less influenced by the size of the normative pattern, and most domains of interest to the insider threat problem are sparse. However, when the graph is dense, the size of the normative pattern does affect the running time of GBAD. Figure 25 shows GBAD's running time on a denser graph with 53,900 vertices as the size of the normative pattern increases. Here we see that larger normative patterns do increase the running time in a dense graph. These results again indicate that a reduction in the time to find the normative pattern can result in a significant reduction in GBAD's overall running time. We are continuing to investigate approaches to reduce the complexity of this phase of GBAD's processing.

8. Discovering Anomalies to Multiple Normative Patterns in Structural and Numeric Data

One of the primary issues with traditional anomaly detection approaches is their inability to handle complex, structural data. The advantage of graph-based anomaly detection is that the relationships between elements can be analyzed, as opposed to just the data values themselves, for structural oddities in what could be a complex, rich set of information. However, until now, attempts at applying graph-based approaches to anomaly detection have encountered two issues: (1) Numeric values found in the data are not incorporated into the analysis of the structure, which could augment and improve the discovery of anomalies; and (2) The anomalous substructure may not be a deviation of the most normative pattern, but deviates from one of many normative patterns. The following proposes enhancements to existing graph-based anomaly detection techniques that address these two issues and shows experimental results validating the usefulness of these enhancements.

8.1. Multiple Normative Patterns

One of the issues with this approach is that many data sets, when represented as a graph, consist of multiple normative patterns. For example, a graph of telephone calls across multiple customers or service providers would contain different calling patterns. The normative "behavior" of one customer would not be representative of another customer's calling pattern. For this reason, most telecommunications fraud detection systems use a "profiling" system to distinguish between different customer calling patterns [Cortes and Pregibon 2001]. However, the issue with these sorts of traditional systems is that they are a type of supervised approach because they require a profile of the customer before they can detect anomalies.

The GBAD approach is unsupervised, discovering substructures that are the smallest deviations from the normative pattern (i.e., the substructure that best compresses the graph). However, if we extend GBAD to consider the top *N* normative substructures, we can then discover other deviations that are potentially more anomalous. This results in the following change to the first step of each of the GBAD algorithms:

Find the N normative substructures S_i that have the N smallest values for $DL(S_i)+DL(G|S_i)$.

where *N* normative patterns are initially discovered, against which potentially anomalous instances are analyzed.

For example, suppose we have the graph in Figure 26.

In Figure 26, the best normative pattern consists of the substructure outlined in the big box. Then, using that normative pattern, GBAD would report the two anomalous substructures shown in the small boxes. However there is another normative pattern which is the second best substructure in the graph, shown outlined with an ellipse (in **bold)**. From that normative pattern, a more anomalous substructure is discovered (shown in a smaller ellipse, also in **bold**), as the probability of an extension to an *A* vertex is rarer than the previously reported anomalous extensions (*Y*) associated with the first normative pattern.

In order to test this in a real-world scenario, we simulated a passport application document processing scenario based upon the process flow depicted in Figure 27. We generated a graph representing the processing of 1,000 passport applications, consisting of approximately 5,000 vertices and 13,000 edges. Potentially, there are two types of prevalent patterns in this type of data: (1) The ApprovalOfficer and CaseOfficer both accept a passport application, and (2) The ApprovalOfficer and CaseOfficer both reject an application. Therefore, potentially anomalous scenarios could exist where the ApprovalOfficer overrides the accept/reject recommendation from the assigned CaseOfficer.

For our testing, we used the OMNeT++ tool to generate a graph consisting of these two normative patterns, although these patterns were not the top-ranked most normative substructures. We then had the tool randomly insert an anomalous instance of the first type (case officer accepts, approval officer rejects) and two anomalous instances of the second type (case officer rejects, approval officer accepts). Applying the GBAD algorithms to this graph allowing only one normative pattern results in only one of the anomalous instances to be discovered. However, when we modify the GBAD-P algorithm (which was the only algorithm to discover an anomalous instance) to analyze the top *N* normative patterns, where *N* is set arbitrarily to 20, all three anomalous instances are reported as the most anomalous. Other experiments showed that the size of *N* was not important. For instance, in this example, when we increase *N* to 100, the top three anomalies reported are still the same ones. In addition, no other substructures are reported as anomalous along with these top three anomalies (i.e., no false positives).

8.2. Numerical Analysis

While GBAD provides for the *structural* analysis of complex data sets, another one of the issues with this approach is the lack of analysis regarding the numeric values that are present in certain data. GBAD has had success discovering anomalies regarding the *relationships* between data entities [Eberle and Holder 2007], including differences between node and link labels, but sometimes the distances between actual entity *values* needs to be considered. Take for instance the simple example shown in Figure 28.

In Figure 28, each person has a name and an age. Running GBAD on this simple graph results in the reporting of the four age vertices as equally anomalous. While each person has an age, because their ages have different values, they are each viewed as being structurally different.

Currently, GBAD-P calculates the probability of the existence of an edge and/or vertex as:

P(attribute=value) = P(attribute exists)

where *P(attribute exists)* is in terms of the probability that it exists as an extension of the normative pattern. However, when we implement the following change to the GBAD-P algorithm:

*P'(attribute=value) = P(attribute=value | attribute exists) **
P(attribute exists)

where the probability of the data is calculated as the probability of the value, given that the attribute even exists, times the probability that it exists. Calculating the mean and standard deviation for all attribute values, we can generate *P(attribute=value | attribute exists)* by using a Gaussian distribution:

$$\rho(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

where σ is the standard deviation and μ is the mean.

Using the same simple example shown in Figure 28, the probability P(x) that each *age* edge exists is 0.25. The mean of the *age* value is 37.75 and the standard deviation is 4.03. When applying this revised probability *P'*, GBAD-P is able to correctly identify that while the structures are the same, with edges labeled "age", the associated vertex with a labeled age of "32", results in the lowest probability, *P'(x),* and thus the greater "anomalousness" (i.e., anomaly value closer to zero):

P'(age=41) = 0.017876	P'(age=38) = 0.024694
P'(age=40) = 0.021173	P'(age=32) = 0.008946

In addition, further experimentation with using a Gaussian probability metric along with the structural anomalous metric indicates that any numeric value less than one standard deviation results in the anomaly not being reported as anomalous. For example, Figure 29 shows how the anomalousness lessens (anomalous score increases) as the numeric value gets closer to the mean, where eventually the originally anomalous vertex is just as anomalous as another vertex. The bottom line in this chart (i.e., the values with the lowest, and thus most anomalous, score) shows that as the age value is increased, closing the gap between its value and the mean value of all of the structures (i.e., the other lines in the chart), the anomalousness of this subgraph lessens (i.e., its score increases).

In order to demonstrate the potential effectiveness of this approach, take the example of a slightly more complex graph that consists of a bank transactions scenario. In this case, the graph consists of 10 bank accounts, where each account exhibits two deposits and two withdrawals. Then one extra deposit was inserted into 3 different accounts, with 2 of the deposits being closer to the mean than the other deposit. The graph consists of vertices labeled "account", "deposit", and "withdrawal", edges labeled "transaction" and "amount", and vertices with dollar values (e.g., "2000.0"), similar to what is shown in Figure 30.

Again, in order to calculate the probability of the normal distribution, first the mean and standard deviation of all of the amount values are calculated. Applying the GBAD-P algorithm, it first discovers the structural differences inherent in the 3 accounts that contain the extra deposits, and then it applies the new Gaussian probability metric to correctly identify the account that contains the deposit with the largest deviation in amount. Also, as was shown in the earlier example, further experimentation with using a Gaussian probability metric on the transaction amount, along with the structural anomaly metric indicates that any value less than one standard deviation results in the anomaly not being reported as anomalous.

What makes this significant from a practical perspective, is that while the value of the anomalous deposit was high ($5000 for this transaction, and $1000 and $2000 for the other two extra deposits), there were actually 11 transactions of this same amount (i.e., out of 43 transactions, over 1/4 of the transactions were at the $5000 level) within this graph. If one were to perform a traditional numerical analysis of this value in terms of all of the deposits (and withdrawals) that were made, the value of $5000 would not have been interesting. However, when combined with the anomaly of the extra structure (i.e., an extra deposit transaction), then it becomes significant.

In addition, earlier we presented a passport application scenario which included the situation where an employee accesses the system after hours. Since this scenario also includes some numerical analysis (hours), we represented time in the graph as the number of hours since midnight, and used this enhanced statistical analysis of numerical attributes as part of its evaluation of the graph structure. For this scenario, we randomly inserted two anomalies into the graph, and the GBAD-P algorithm was able to successfully discover both anomalies where access to the company database was during unexpected hours, with no false positives reported. While the structure was the same, the time information (represented as a number), provides extra information that aides in the insider threat detection.

9. Dynamic Graphs

So far, GBAD only detects anomalies in static graphs. However, many domains in which we desire to detect anomalies are dynamic; that is, the information is changing over time. One solution to this scenario is to collect data over a time window, build a graph from this data that may or may not explicitly represent time, and then apply GBAD to the graph. While this solution will find anomalies to patterns within the time window, any dynamic component to the patterns and anomalies will rely on a proper representation of time and a sufficiently long time window in which to observe the patterns' regularity.

One approach to detecting patterns of structural change in a dynamic graph, which has been successfully applied to the domain of biological networks [You et al. 2008], is called DynGRL [You et al. 2008]. In this approach, DynGRL first learns how one graph is structurally transformed into another using graph rewriting rules, and then abstracts these rules into patterns that represent the dynamics of a sequence of graphs. The goal of DynGRL is to describe how the graphs change over time, not merely whether they change or by how much.

Graph rewriting rules represent topological changes between two sequential versions of the graph, and transformation rules abstract the graph rewriting rules into the repeated patterns that represent the dynamics of the graph. Figure 31 shows the framework of this approach. The dynamic graph contains a sequence of graphs that are generated from sampling snapshots of the graph from a continuously-changing graph. First, the approach learns graph rewriting rules including removals (R_i) and additions (A_{i+1}) between two sequential graphs G_i and G_{i+1} (Figure 31 (B)), and generates a list of all graph rewriting rules (Figure 31 (C)). The final step is to learn the transformation rules to abstract the structural change of the dynamic graph based on the repeated patterns in the graph rewriting rules. If some structural changes are repeated in the dynamic graph, there exist common subgraphs in the Rs and As.

In order to detect anomalies in the change of dynamic graphs, we must first learn how one graph is structurally transformed into another, and then abstract patterns that represent the dynamics of a sequence of graphs. In order to detect anomalies, the goal is to describe how the graphs change over time, and discover those changes that are structurally anomalous. Specifically, we want to (1) look for structural modifications, insertions and deletions to nearby instances of the transformation rules as potential anomalies to the normative pattern, and (2) detect anomalies in the temporal application of the transformation rules, e.g., when in some cases the structure does not appear exactly four times after it was last removed. Evaluation of our approach will involve graphs that represent data dynamically changing over time.

Using this approach, we coupled DynGRL with GBAD to produce a system for discovering anomalies in dynamic graphs, which we call DynGBAD. First DynGBAD produces a sequence of difference graphs for each pair of graphs in the time-slice sequence, searching for recurring patterns in these difference graphs. DynGBAD then analyzes these difference graphs using the normative recurring patterns discovered by the relational learner (DynGRL) and identifying anomalies to these patterns (GBAD). A dynamic anomaly may be a change in the dynamic pattern at some point in time (similar to what GBAD already does), but also may consist of a change in the period of recurrence of the pattern. Our hypothesis is that a representation that links the difference graphs together will allow DynGBAD to detect such anomalies.

To further explain how DynGBAD works, take the simple example shown in Figure 32. The general idea is that the subgraph consisting of vertices F, G and H appears and disappears regularly over time. Specifically, the subgraph appears and disappears in a 10100 pattern, where 1 means presence of the subgraph at that time, and 0 means its absence. However, at one point in the graph, the subgraph appears as 11000 – an anomaly in the regularity of the dynamic graph. Figure 32(a) shows the input dynamic graph for this problem, and Figure 32(b) shows the normative pattern found by GBAD in this dynamic graph. Figure 32(c) shows the anomaly found by GBAD. While this anomaly does not make obvious the change in regularity of the F-G-H subgraph, it is in fact the instance of the subgraph that occurs at the time of the anomaly. So, GBAD does identify the anomalous event, just not in the form we would like to see. However, when we expand the size of the dynamic graph to include more occurrences of the 10100 pattern in the likelihood that GBAD can discover the entire 10100 sequence as the normative pattern, the discovery of the 11000 anomaly is more apparent. Figure 33 shows the normative pattern found by DynGBAD after extending the size of the dynamic graph to include more instances of the normative pattern. In this case, DynGBAD finds the complete normative pattern consisting of two copies of the FGH subgraph separated by one time slice. Figure 34 shows the anomaly found by DynGBAD in the extended dynamic graph. The blue vertices and edges (located inside the squiggly shape) represent the discovered normative pattern, the orange (located inside of the box) represents the complete anomalous instance, and the red (located inside of the oval) indicates the specific anomaly that triggered the discovery of the entire anomalous instance. Here we see that DynGBAD finds the anomaly within a larger context; namely, two FGH subgraphs occur in two consecutive time slices. So, in order to find the proper context, we must be sure that the normative pattern occurs with sufficient frequency to be fully captured by DynGBAD; thus, allowing anomalies to these patterns to be discovered with sufficient context for a user to determine where they occur in the dynamic graph and to what extent they differ from the normative pattern.

We have also done some initial real-world testing of our DynGBAD approach using the Enron email dataset. Previously, we have used GBAD to analyze this dataset by looking for anomalies in graphs representing the email traffic of an employee, but we have yet to consider how this traffic changed over time, or what anomalies to these time-changing patterns may occur. First, we changed the static representation of the Enron data to that of a dynamic graph. Each dynamic graph consists of a sequence of graphs, each representing one day of email activity for a particular employee (e.g., Kenneth Lay). Figure 35 shows a portion of the graph of emails involving Kenneth Lay on October 10, 2001. This is one graph in the sequence comprising the dynamic graph. Initially, we applied DynGBAD on dynamic graphs constructed from each day in October 2001 and on each Wednesday (an arbitrary day of the week, where perhaps there is the most activity) in 2001. However, the normative pattern tends to be the message infrastructure (“message”, “sender”, “original” nodes) and the resulting anomalies are uninteresting.

Figure 36 shows an alternative representation, where the message information is removed and only the senders and receivers are included in the graph as nodes. Our desire in analyzing this type of representation is that we will discover patterns in how these graphs change over time (e.g., Kenneth Lay emails Joannie Williamson every other week) and then anomalies to these dynamic patterns (e.g., Kenneth Lay did not email Joannie Williamson on the second Wednesday of the month, as predicted by the normative pattern). Such anomalous behavior may prompt further investigation into an employee’s activities.

The ability of DynGBAD to detect anomalies to patterns of change in dynamic graphs will open up a new dimension in the analysis of processes for detecting anomalies and insider threats. Since an insider may exhibit behavior that is structurally similar over time, detecting anomalies in the regularity of this behavior may be necessary to detect nefarious activities.

10. Conclusions and Future Work

Using the MDL principle and probabilistic approaches, we have been able to successfully discover anomalies in graphs and patterns of varying sizes with minimal to no false positives. Results from running the GBAD algorithms on e-mail, cell-phone traffic, business processes and cybercrime, show how these graph-theoretic approaches can be used to identify insider threats. While we have been able to achieve some minimal successes when applying graph-theoretic algorithms to dynamic graphs that change over time, clearly we have only begun to scratch the surface. In addition, we are going to continuously explore the incorporation of traditional data mining approaches as additional quantifiers to determining anomalousness. Using the OMNeT++ example, we can create limitless numbers and varieties of simulations modeling business processes, network traffic, email flows, etc. These can then be used to evaluate GBAD systematically and on models of real-world processes.

Two of the issues with current graph-based anomaly detection approaches are their inability to use numeric values along with their structural analysis to aid in the discovery of anomalies, and their inability to discover anomalous substructures that are not part of the normative pattern. In the future, we are going to continue researching other numeric analysis approaches that can be incorporated into the structural analysis so as to further delineate “anomalousness”. In addition, we will analyze our ability to discover an anomaly involving two different numeric attributes that individually are not anomalous, but together are rare. We will also investigate the limitations involved with analyzing multiple normative patterns, including how well this approach scales with the size of the graph, number of normative patterns, and size of the normative patterns.

In addition, we have only just begun to research the effectiveness of applying a graph-based approach to dynamic data. Further research into not only the graph representation of dynamic data, but also the techniques for analyzing graphs that represent data that is changing over time, will be valuable for providing a more comprehensive graph-based anomaly detection approach to discovering insider threats.

Acknowledgements

This material is based upon work supported by the Department of Homeland Security under Contract No. N66001-08-C-2030. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of Homeland Security.

References

AFCE (2006). 2006 AFCE Report to the Nation on Occupational Fraud & Abuse. *Association of Certified Fraud Examiners*.

Bush, J. (2009). Survey suggests economy could lead to cybercrime increase. *Purdue University News Service*. March 19, 2009.

Cauley, L. (2006). NSA has massive database of Americans' phone calls. *USA Today*. May 11, 2006.

Chakrabarti, D. (2004). AutoPart: Parameter-Free Graph Partitioning and Outlier Detection. *PKDD 2004, 8th European Conference on Principles/Practices of KDD*. pp. 112-124.

Chun, A. (2008). An AI framework for the automatic assessment of e-government forms. *AI Magazine*. Volume 29.

Cook, D. & Holder, L. (1998). Graph-based data mining. *IEEE Intelligent Systems.* 15(2), 32-41.

Cortes, C. & Pregibon, D. (2001). Signature-based methods for data streams. *Data Mining and Knowledge Discovery 2001*. 5(3): 167-182.

Diesner, J. & Carley, K. (2005). Exploration of Communication Networks from the Enron Email Corpus. *Computational and Mathematical Organization Theory*. Volume 11, Issue 3, pp. 201-228.

Eberle, W. & Holder, L. (2007). Anomaly Detection in Data Represented as Graphs. *Intelligent Data Analysis, An International Journal*.   Volume 11(6).

Eberle, W. & Holder, L. (2008). Analyzing Catalano/Vidro Social Structure Using GBAD. *VAST 2008 Challenge Track, VisWeek*. October, 2008.

Eberle, W. & Holder, L. (2009). Mining for Insider Threats in Business Transactions and Processes. *Computational Intelligence in Data Mining, IEEE Symposium Series on Computational Intelligence*. April 2009.

Eberle, W. & Holder, L. (2009). Applying Graph-based Anomaly Detection Approaches to the Discovery of Insider Threats. *IEEE International Conference on Intelligence and Security Informatics*. June 2009.

Eberle, W., Holder, L. & Graves, J. (2009). Detecting Employee Leaks Using Badge and Network IP Traffic. *IEEE Symposium on Visual Analytics Science and Technology*. October 2009.

Foley, L. (2008). ITRC Beach Meter Reaches 342, to Date. *Reuters*, June 30, 2008.

Hampton, M. & Levi, M. (1999). Fast spinning into oblivion? Recent developments in money-laundering policies and offshore finance centres. *Third World Quarterly*. Volume 20, Number 3, pp. 645-656.

Hao, M. et al. (2006). Business process impact visualization and anomaly detection. *Information Visualization.* Volume 5, Issue 1.

Kirk, J. (2009). In poor economy, IT pros could turn to e-crime. *IDG News Service*. March 25, 2009.

Kolata, G. (2005). Enron Offers an Unlikely Boost to E-Mail Surveillance. *www.nytimes.com*. May 22, 2005.

Kowalski, E. (2008). Insider Threat Study: Illicit Cyber Activity in the Information Technology and Telecommunications Sector. *http://www.cert.org/insider_threat/.*

Kowalski, E. (2008). Insider Threat Study: Illicit Cyber Activity in the Government Sector. *http://www.cert.org/insider_threat.*

Kurcz, M. et al. (2007). Spectral Clustering in Telephone Graphs. *Joint 9th WEBKDD and 1st SNA-KDD Workshop '07*. August 12, 2007.

Kuramochi, M. & Karypis, G. (2004) An efficient algorithm for discovering frequent subgraphs. *IEEE Trans. Knowl. Data Eng*., 16(9):1038–1051.

Lin, S. & Chalupsky, H. (2003). Unsupervised Link Discovery in Multi-relational Data via Rarity Analysis. *Proceedings of 3rd IEEE ICDM Intl. Conf. on Data Mining*. pp. 171-178.

Martin, S. et al. (2005). Analyzing Behaviorial Features for Email Classification. *CEAS 2005: Conference on Email and Anti-Spam*. July 21-22, 2005.

Noble, C. & Cook, D. (2003). Graph-Based Anomaly Detection. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 631-636, 2003.

OMNeT++. http://www.omnetpp.org/.

Ponemon, L. (2009). Cyber Crime: The 2009 MegaTrend. *CSO*.

Priebe, C. et al. (2005). Scan Statistics on Enron Graphs. *Computational and Mathematics Organization Theory*. Volume 11, Number 3, p229 - 247.

Randazzo, M. et al. (2004). Insider Threat Study: Illicit Cyber Activity in the Banking and Finance Sector. *http://www.cert.org/insider_threat/*.

Rattigan, M. & Jensen, D. (2005). The case for anomalous link discovery. *ACM SIGKDD Exploration News*. 7(2):41--47.

Rissanen, J. (1989). Stochastic Complexity in Statistical Inquiry. *World Scientific Publishing Company*.

Shetty, J. & Adibi, J. (2005). Discovering Important Nodes through Graph Entropy: The Case of Enron Email Database. *KDD, Proceedings of the 3rd international workshop on link discovery*. 74-81.

Staniford-Chen, S. et al. (1996). GrIDS – A Graph Based Intrusion Detection System for Large Network. *Proceedings of the 19th National Information Systems Security Conference.*

Sun, J et al. (2005). Relevance search and anomaly detection in bipartite graphs. *SIGKDD Explorations* 7(2). 48-55.

Swayne, D. et al. (2003). Exploratory Visual Analysis of Graphs in GGobi. *Proceedings of the 3rd International Workshop on Distributed Statistical Computing*, DSC 2003, March 20-22, 2003.

United States Department of Justice. (2009). Computer Crime & Intellectual Property Section. *http://www.justice.gov/criminal/cybercrime/*.

Vijayan, J. (2009). Insider at Cal Water steals $9M and runs. *Computerworld Security*. May 22, 2009.

VAST. (2009). Challenge: *www.cs.umd.edu/hcil/VASTchallenge09*.

VAST. (2008). Challenge: *www.cs.umd.edu/hcil/VASTchallenge08*.

Yan, X. & Han, J. (2002). gSpan: Graph-based substructure pattern mining. *In ICDM*, pages 721–724.

You C., Holder, L. & Cook, D. (2008) Graph-based Temporal Mining of Metabolic Pathways with Microarray Data. *ACM SIGKDD Workshop on Data Mining in Bioinformatics (BIOKDD)*, August 2008.

You, C., Holder, L. & Cook, D. (2008) Graph-based Data Mining in Dynamic Networks: Empirical Comparison of Compression-based and Frequency-based Subgraph Mining. *IEEE International Conference on Data Mining (ICDM) Workshop on Analysis of Dynamic Networks*, December 2008

Table 1 Results for different normative pattern sizes for each anomaly type on an input graph of 1700 vertices.

Size of Normative Pattern	**Anomalous Type**	**Found All Anomalies?**	**False Positives?**	**CPU Seconds (before enhancement)**	**CPU Seconds (after enhancement)**
3v/2e	Deletion	Yes	No	0.05	
	Insertion	Yes	No	0.05	
	Modification	Yes	No	0.05	
5v/4e	Deletion	Yes	No	0.12	
	Insertion	Yes	No	0.13	
	Modification	Yes	No	0.11	
10v/9e	Deletion	Yes	No	0.17	
	Insertion	Yes	No	0.27	
	Modification	Yes	No	0.57	0.22
20v/19e	Deletion	Yes	No	0.36	
	Insertion	Yes	No	0.64	
	Modification	Yes	No	1346.26	0.37
40v/39e	Deletion	Yes	No	6.27	
	Insertion	Yes	No	1.45	
	Modification	Yes	No	(process killed after a few days)	6.78

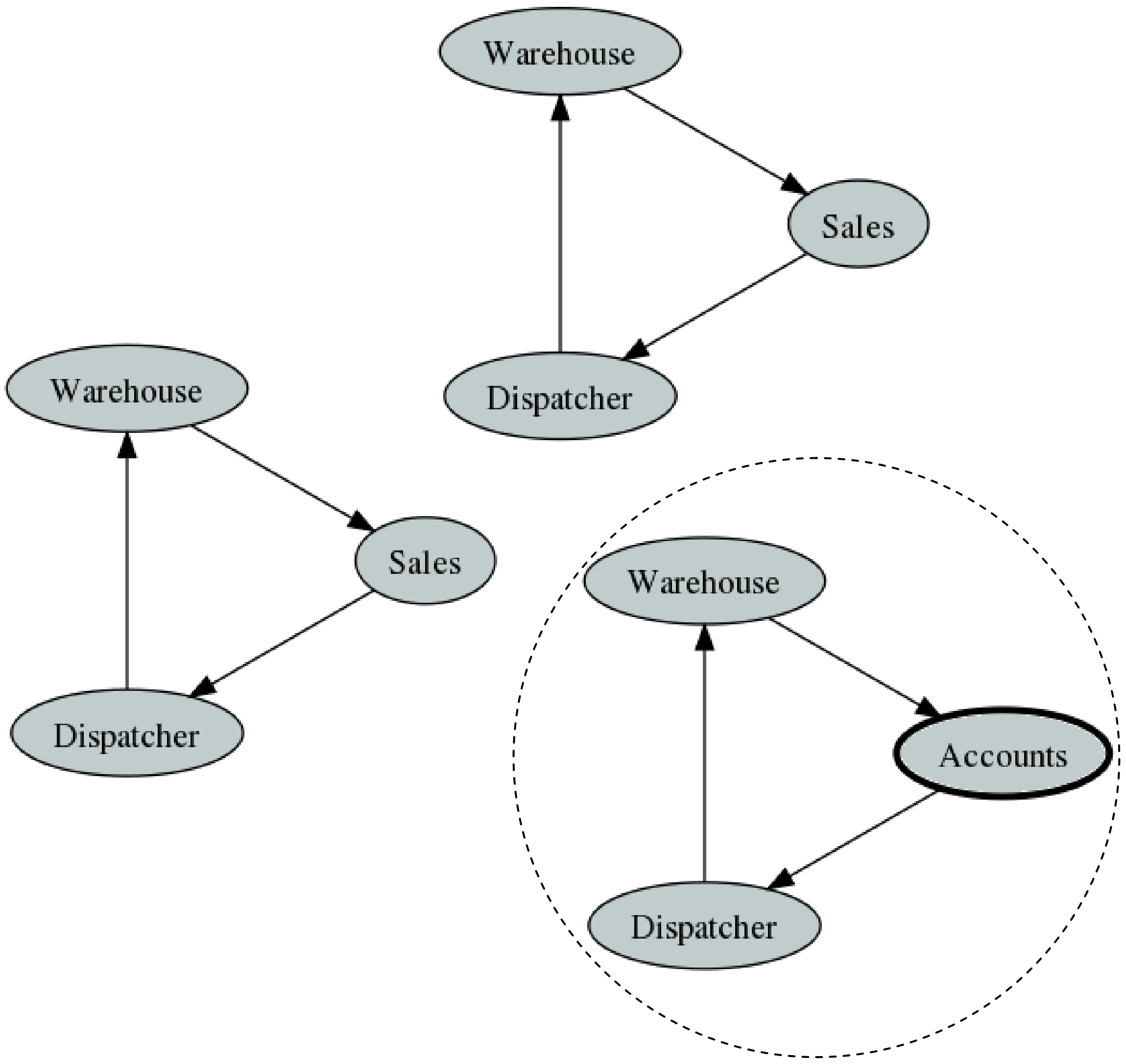

Fig. 1 GBAD-MDL example with anomalous instance circled.

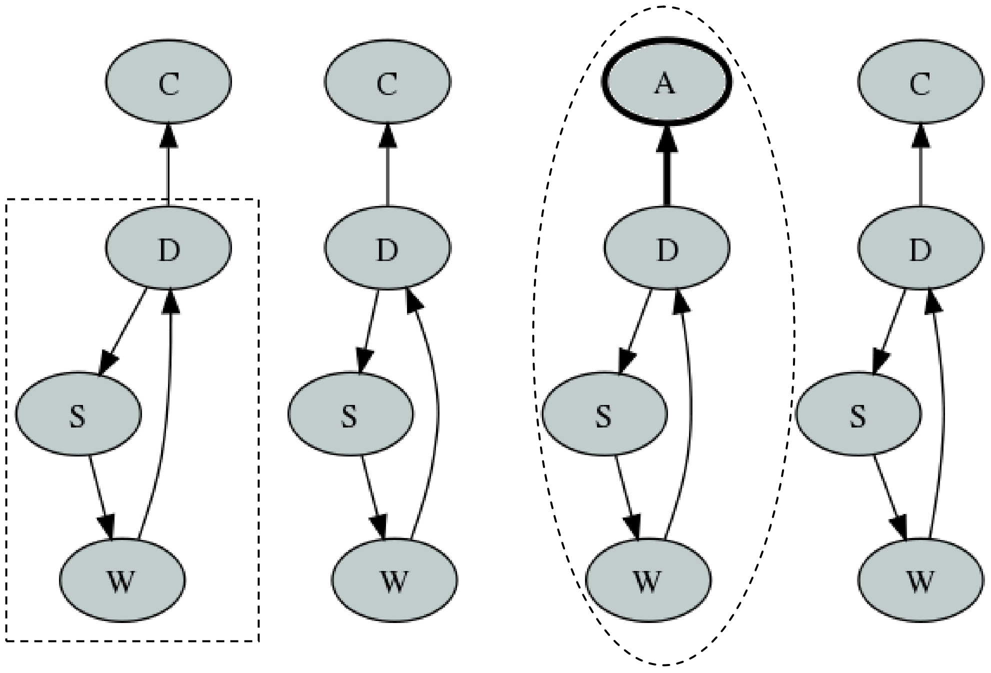

Fig. 2 GBAD-P example with instance of normative pattern boxed and anomaly circled.

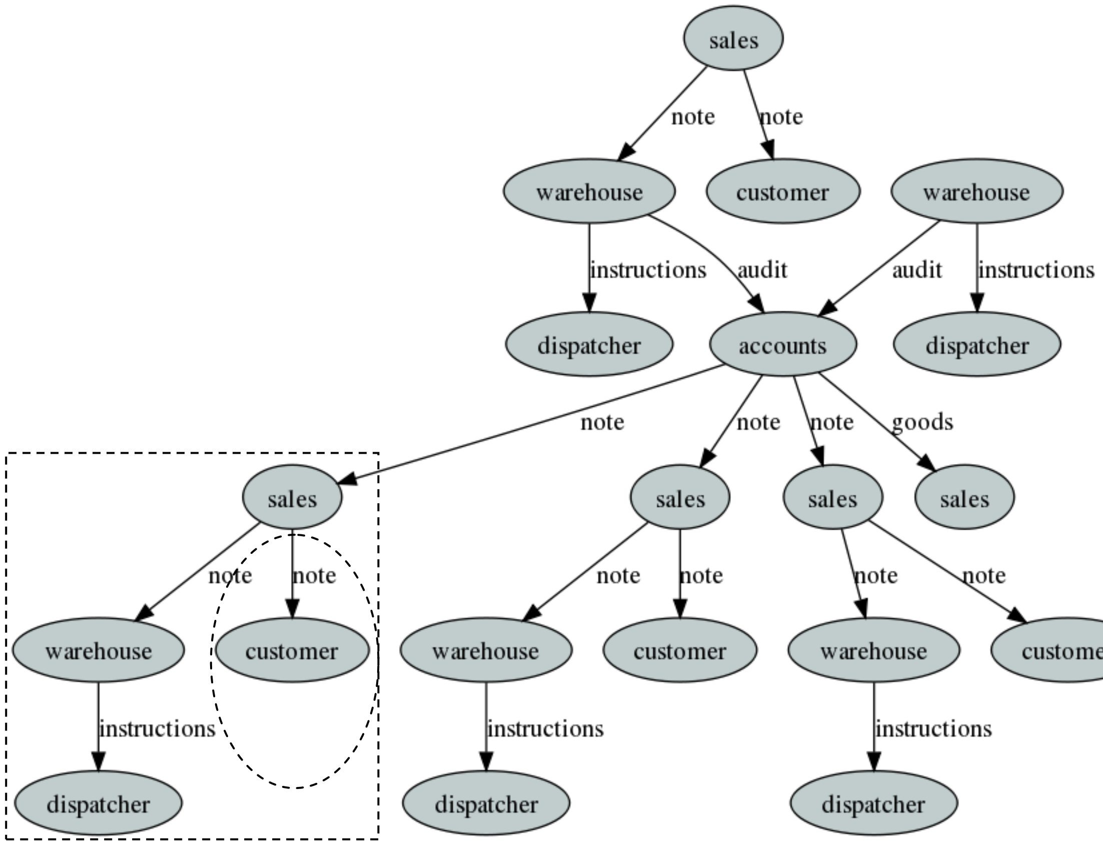

Fig. 3 GBAD-MPS example with instance of normative pattern boxed and anomaly circled.

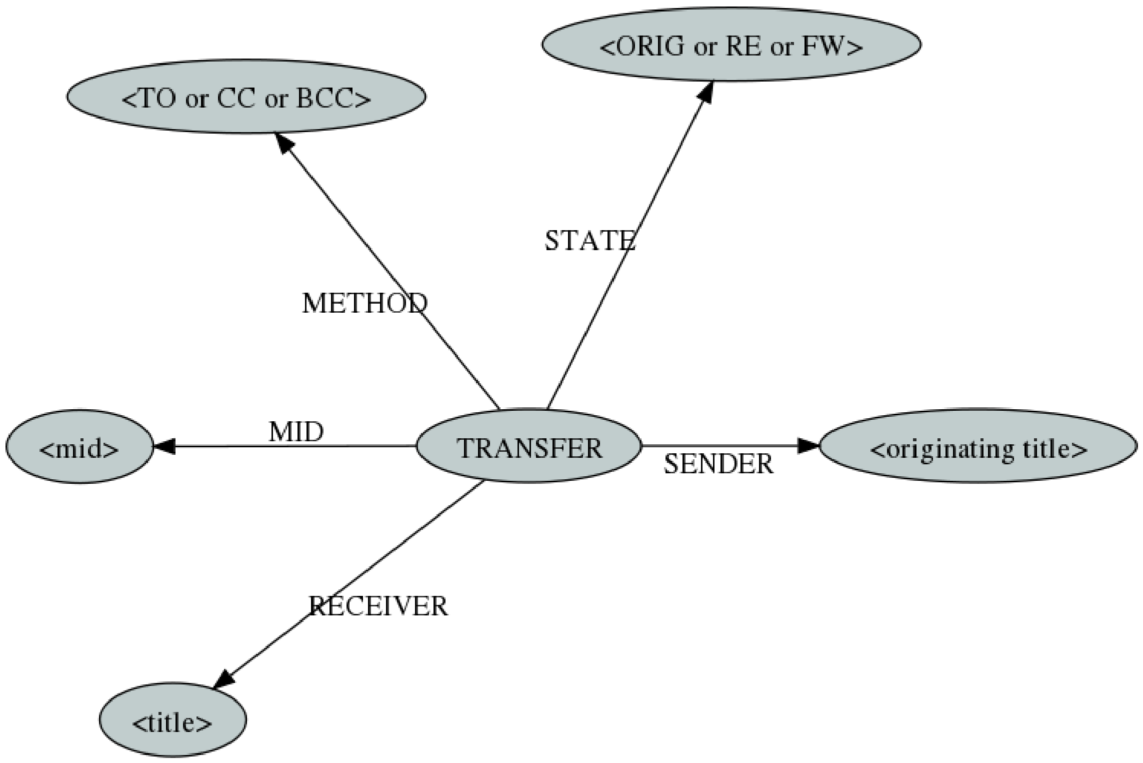

Fig. 4 Graph substructure of e-mail data set.

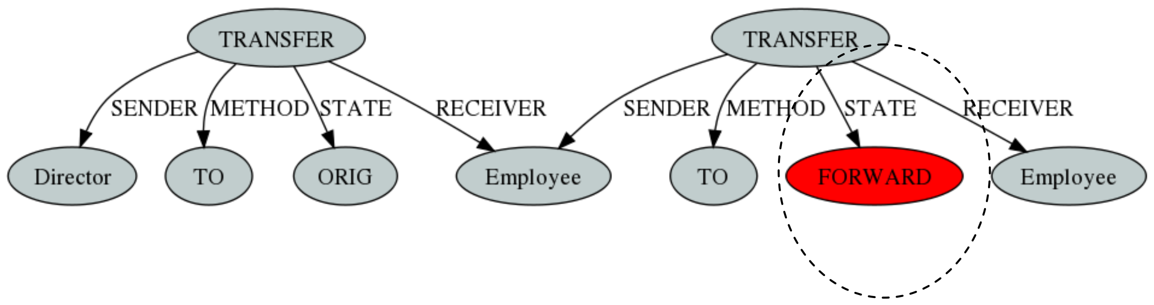

Fig. 5 Anomalous instance (portion) of e-mail being forwarded.

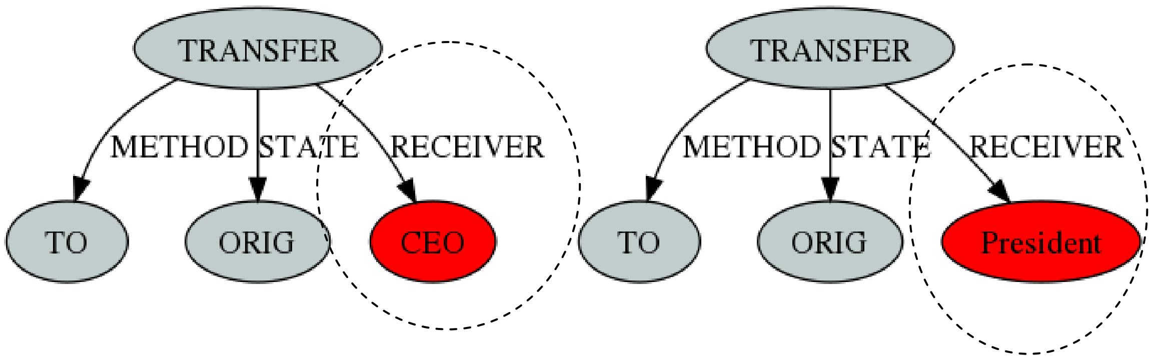

Fig. 6 Anomalous instances of e-mails to a CEO and to a President.

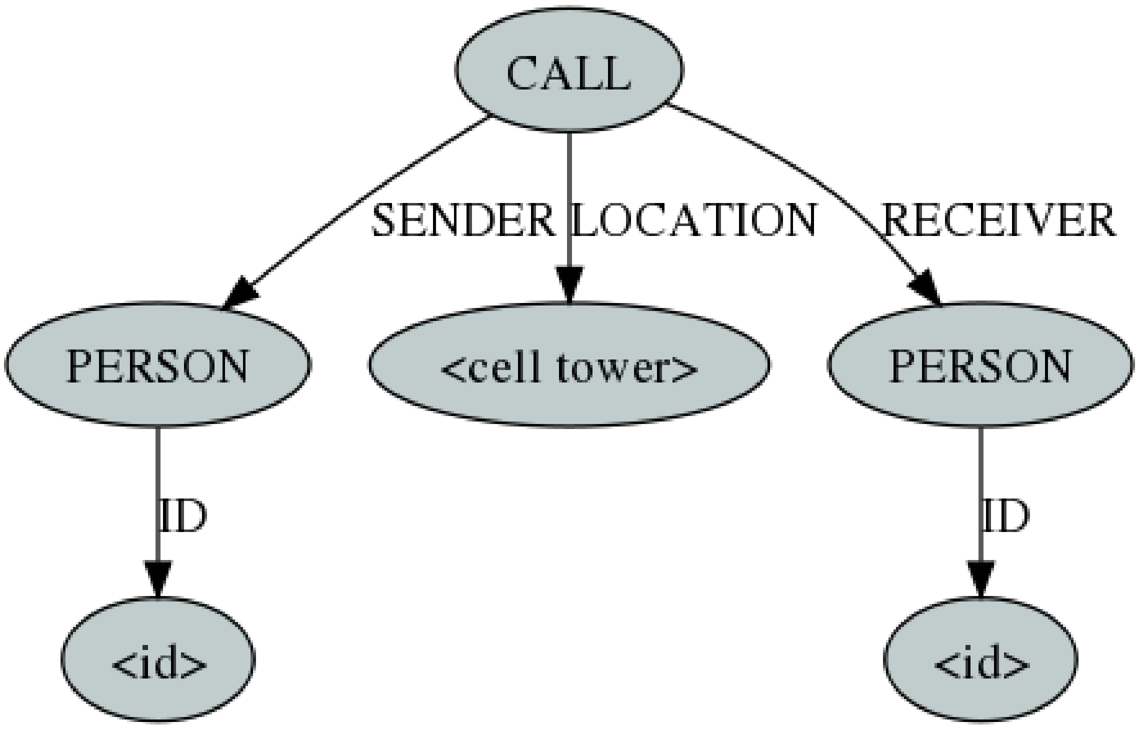

Fig. 7 Graph of a cell-phone call from the VAST dataset.

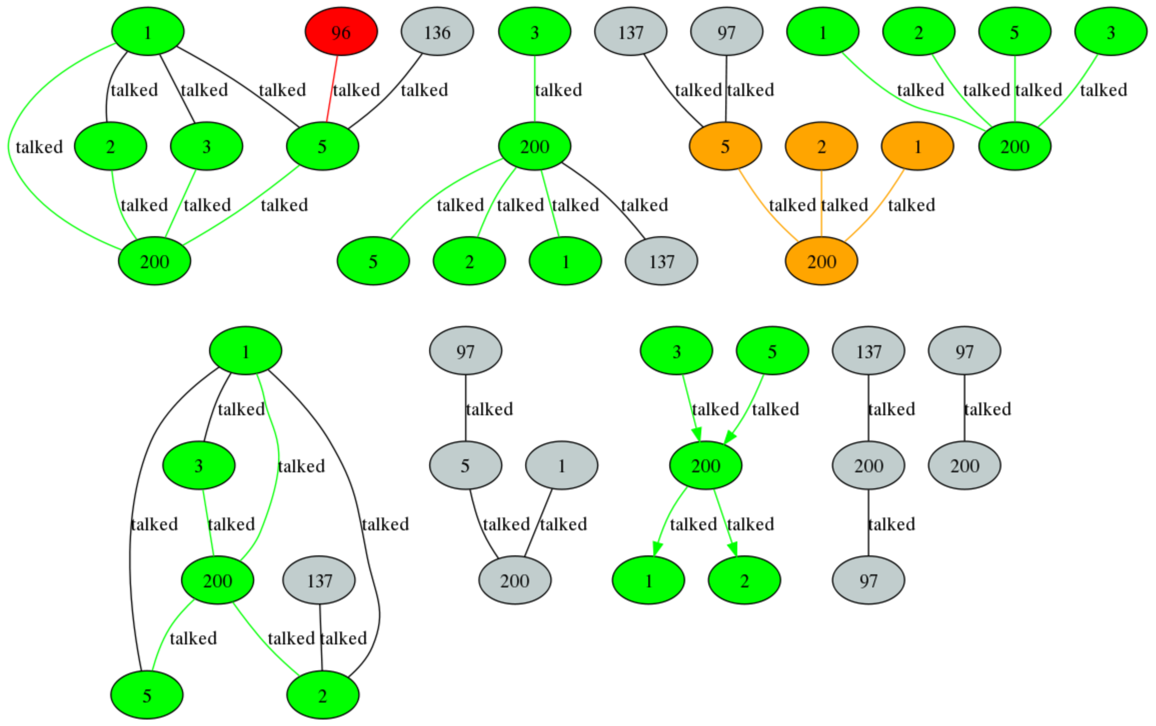

Fig. 8 Ferdinando/Catalano's social structure with associated normative pattern and anomalies.

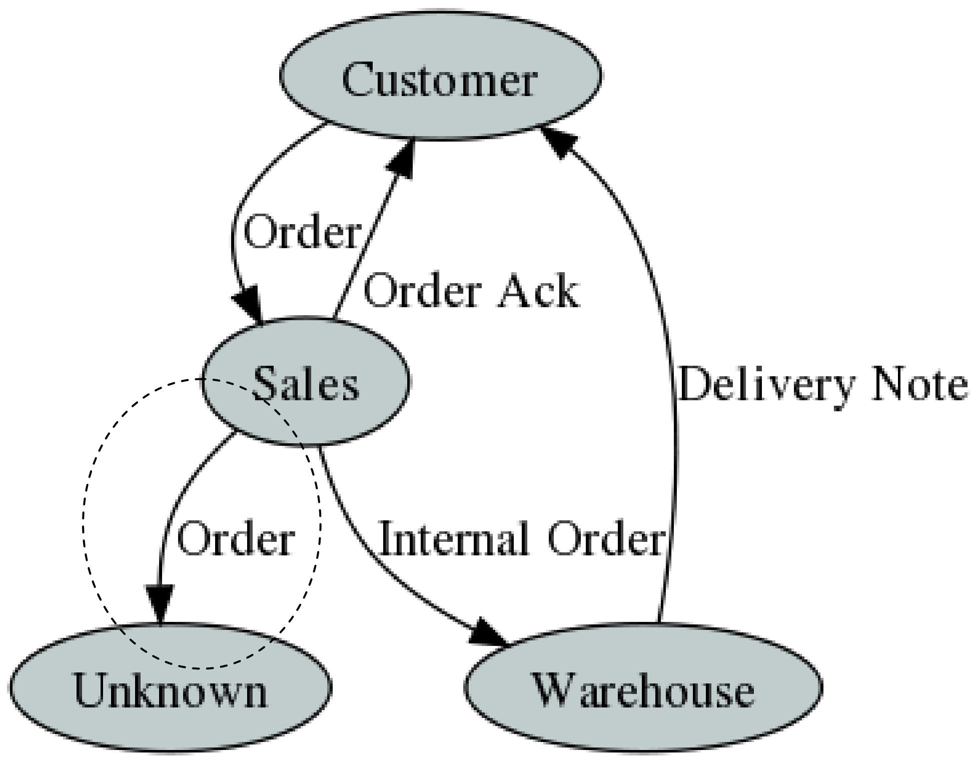

Fig. 9 Depiction of an order fulfillment process; circled edge indicates a low-probability anomaly.

```
OMNeT++/OMNEST Discrete Event Simulation  (C) 1992-2005 Andras Varga
Release: 3.3, edition: Academic Public License.
See the license for distribution terms and warranty disclaimer
Setting up Cmdenv...

Preparing for Run #1...
Setting up network `orderprocess'...
Initializing...

Running simulation...
** Event #0 T=0.0000000 (0.00s).  (Customer) orderprocess.customer (id=2)
[GBAD] 1 Order: Customer -> Sales (0)
** Event #1 T=0.0000000 (0.00s).  (Sales) orderprocess.sales (id=3)
** Event #2 T= 30.8511 (30.85s).  (Sales) orderprocess.sales (id=3)
[GBAD] 1 InternalOrder: Sales -> Warehouse (30.8511)
[GBAD] 1 OrderAcknowledgement: Sales -> Customer (30.8511)
** Event #3 T= 30.8511 (30.85s).  (Warehouse) orderprocess.warehouse (id=4)
** Event #4 T= 30.8511 (30.85s).  (Customer) orderprocess.customer (id=2)
** Event #5 T=80603.875 (22h 23m).  (Customer) orderprocess.customer (id=2)
[GBAD] 2 Order: Customer -> Sales (80603.9)
** Event #6 T=80603.875 (22h 23m).  (Sales) orderprocess.sales (id=3)
** Event #7 T=80613.88 (22h 23m).  (Sales)
```

Fig. 10 Partial OMNeT++ simulation output.

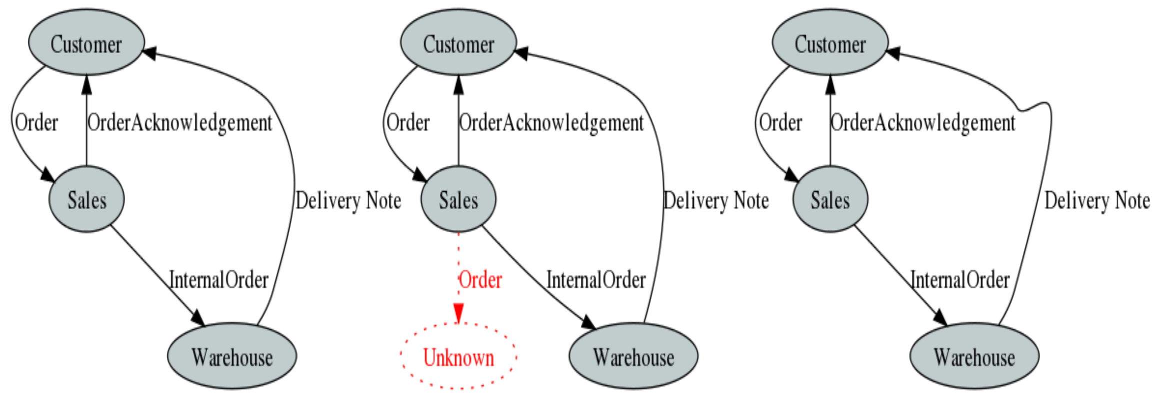

Fig. 11 Subdue-formatted (partial) graph produced from GBAD-enhanced OMNeT++ simulation output.

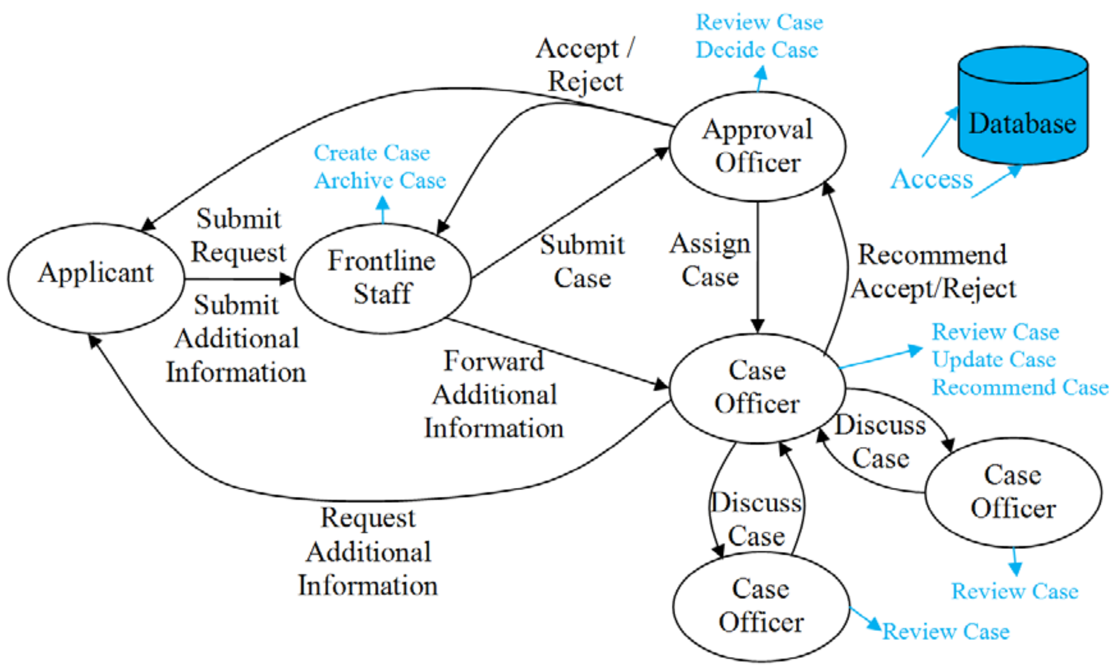

Fig. 12 Passport application process.

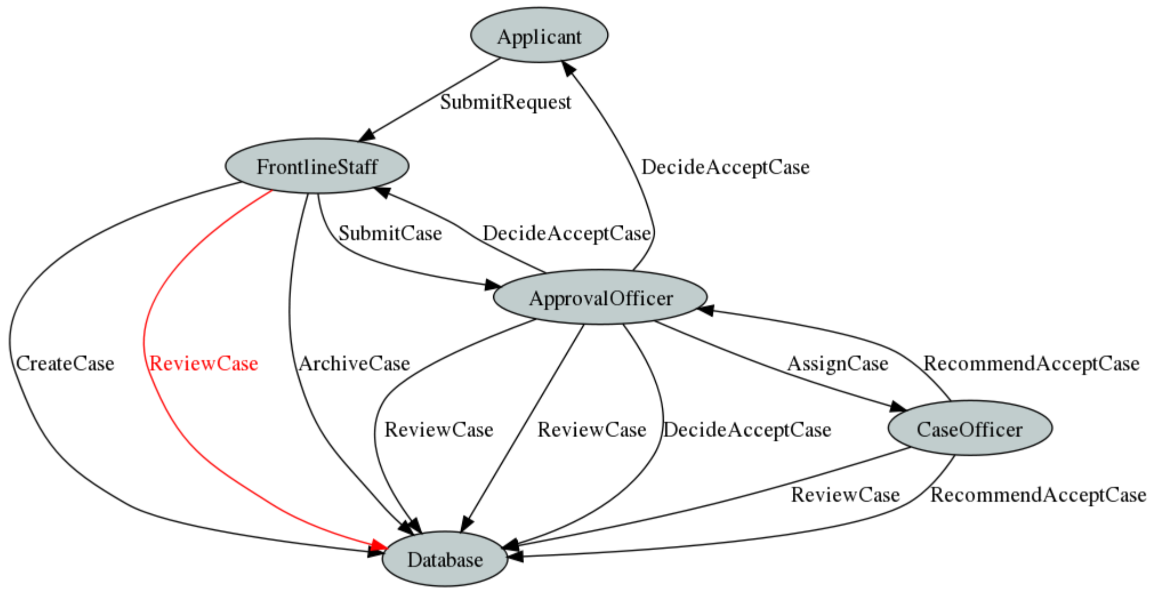

Fig. 13 Scenario 1 normative pattern and anomaly.

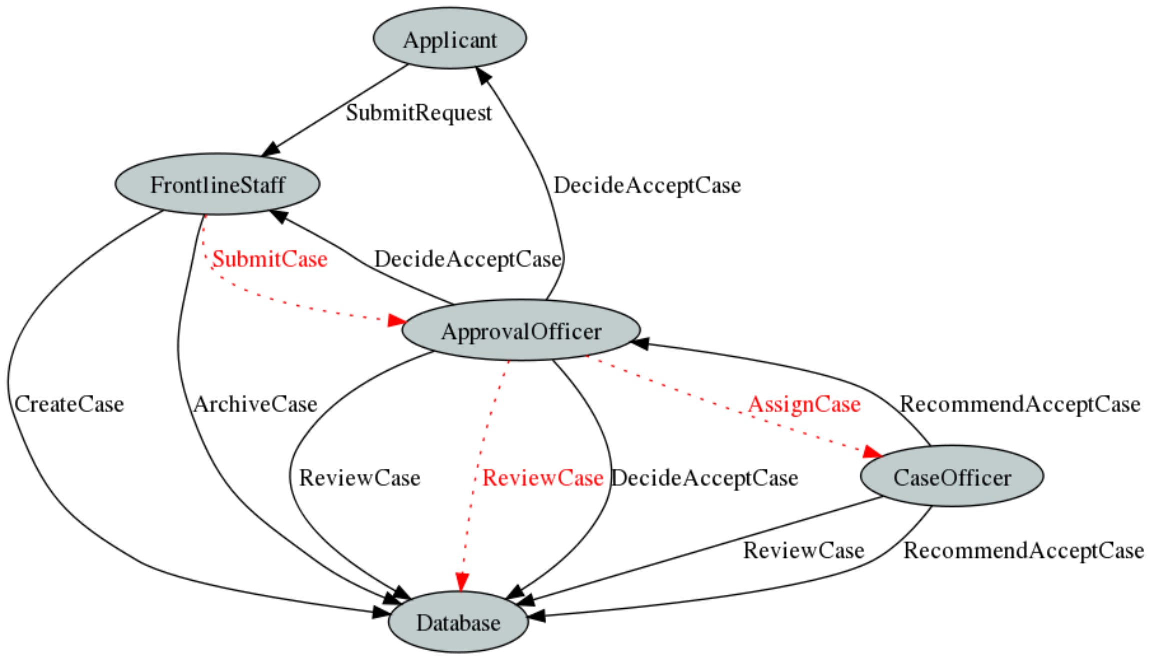

Fig. 14 Graph of Scenario 2, showing the normative pattern and missing edges.

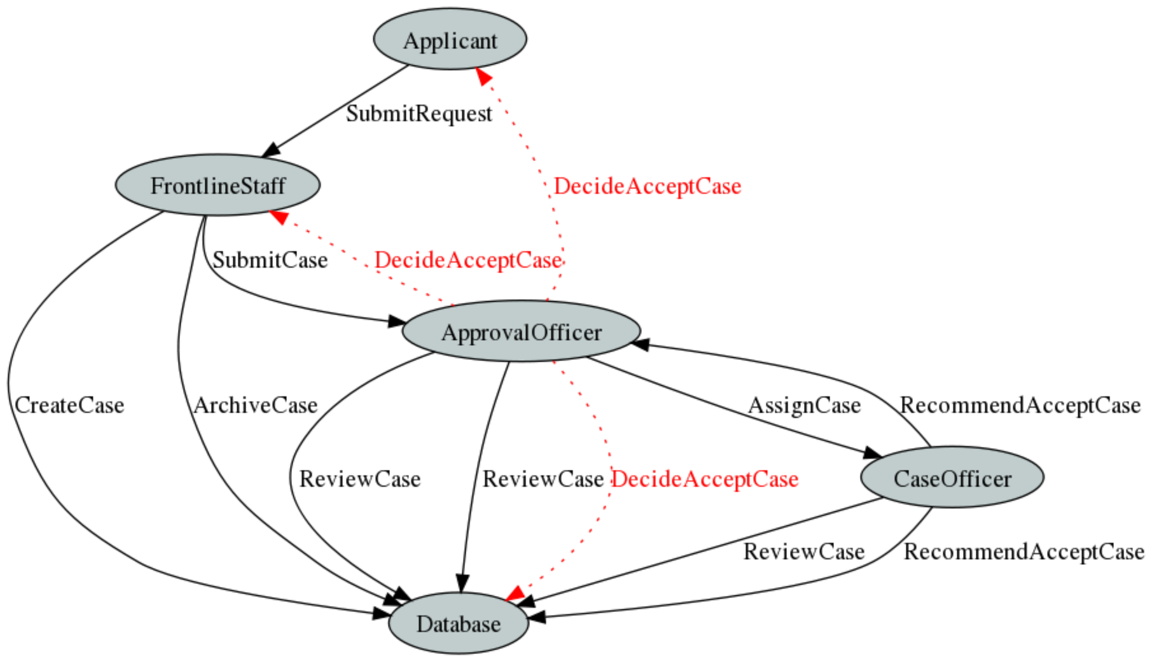

Fig. 15 Graph of Scenario 4, showing the normative pattern and unexpected edge labels.

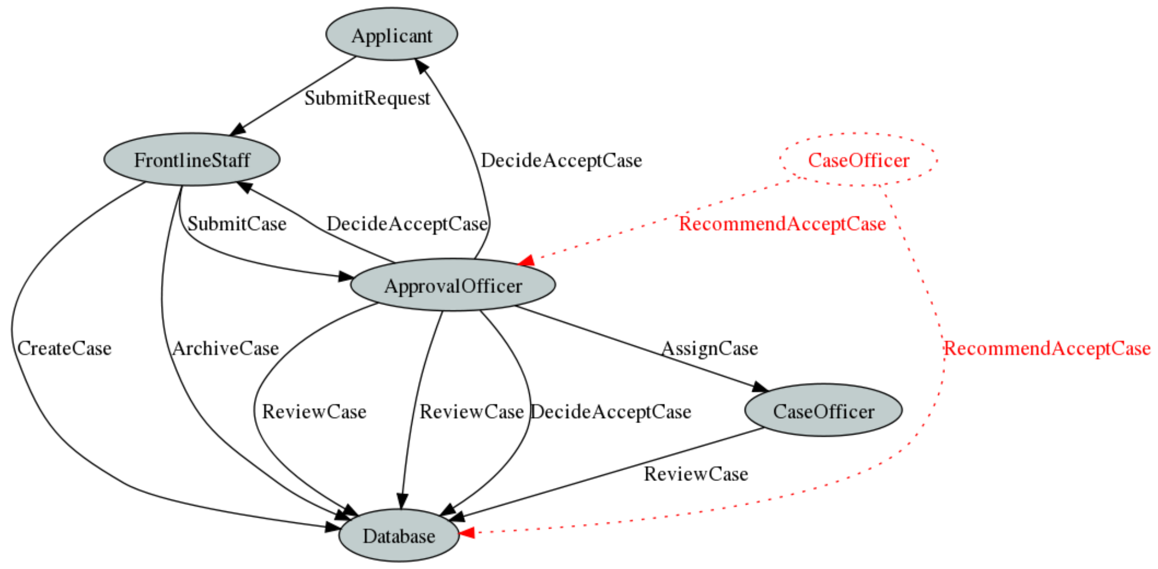

Fig. 16 Scenario 5, unauthorized handling of a case.

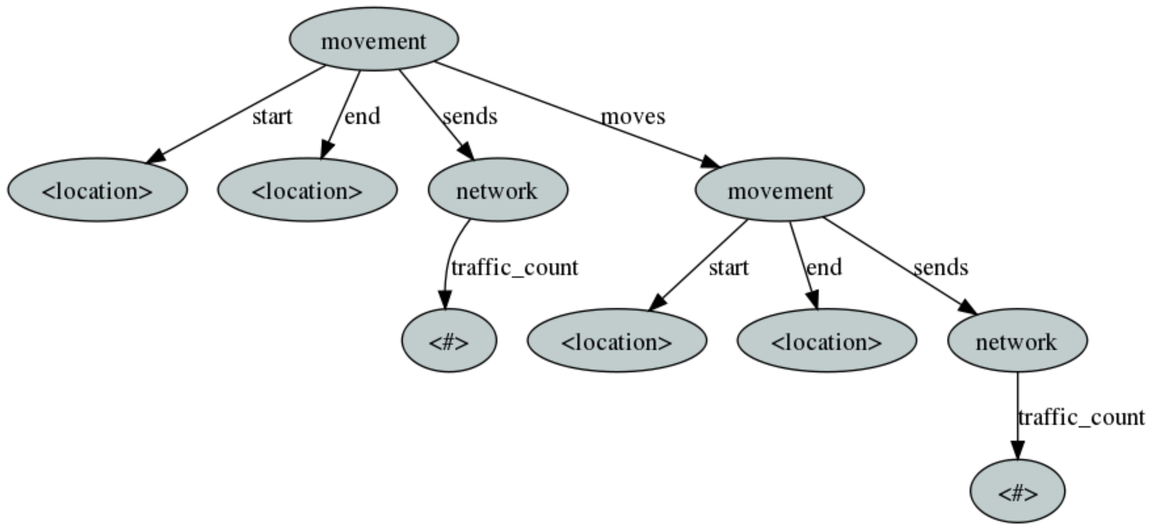

Fig. 17 Graph topological representation.

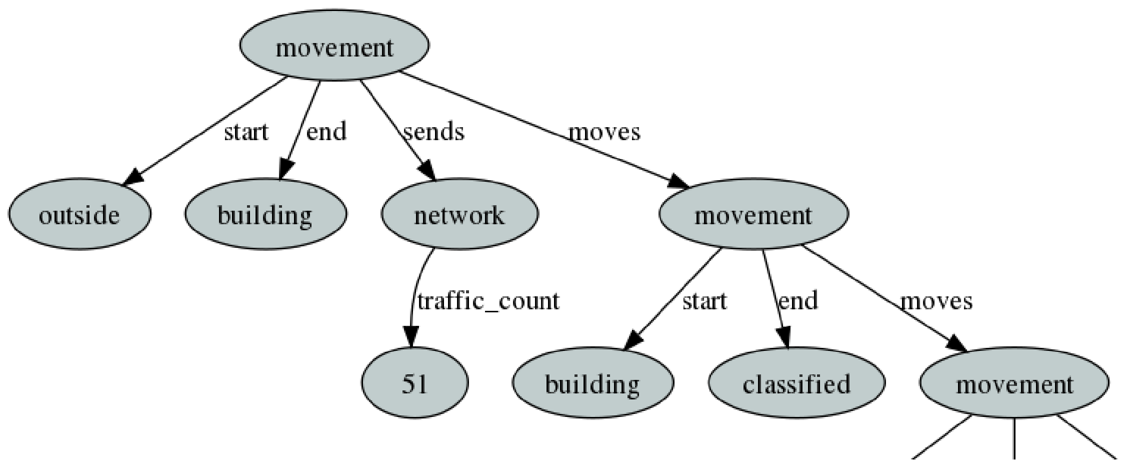

Fig. 18 Example movement and activity (partial graph shown).

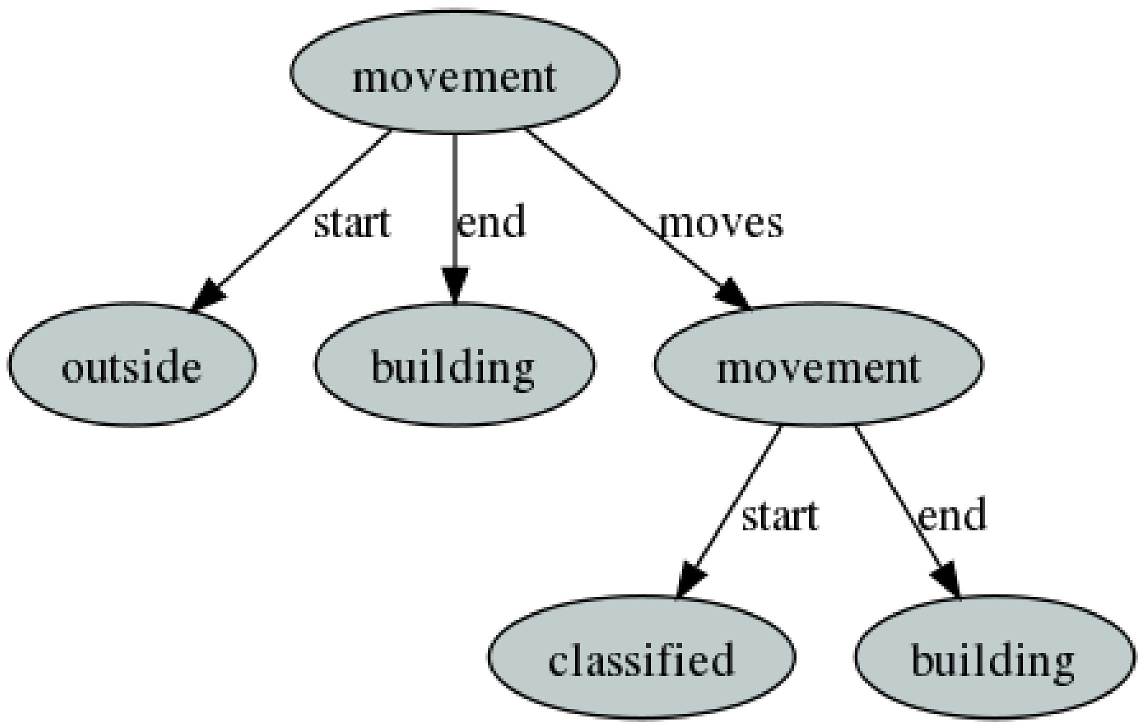

Fig. 19 Normative pattern.

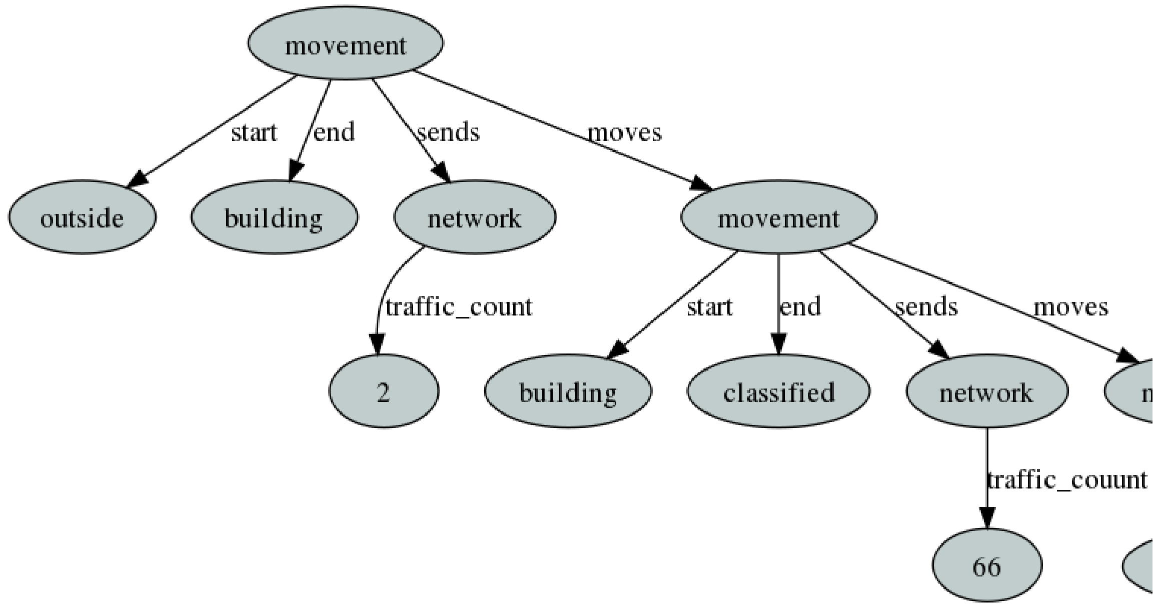

Fig. 20 Anomalous structure (in the graph).

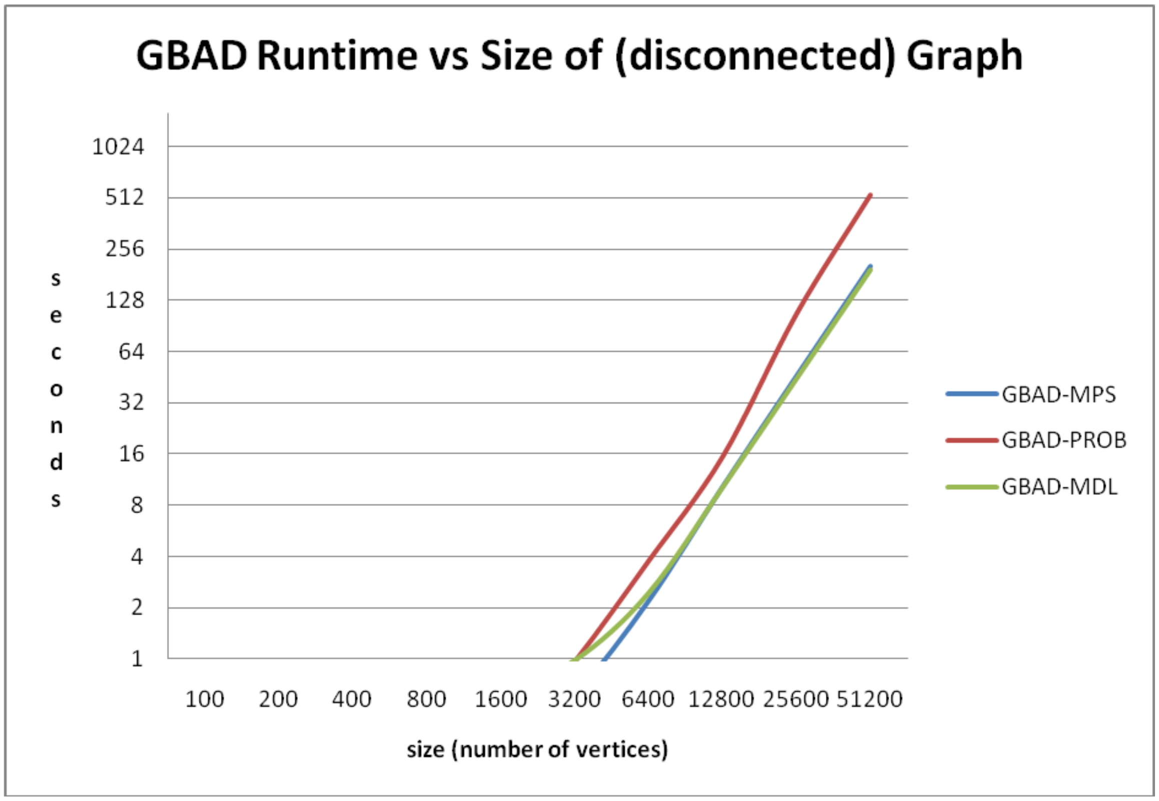

Fig. 21 GBAD running time as a function of graph size (number of vertices) for <u>disconnected</u> graphs. This log-log plot shows that GBAD scales polynomial with graph size.

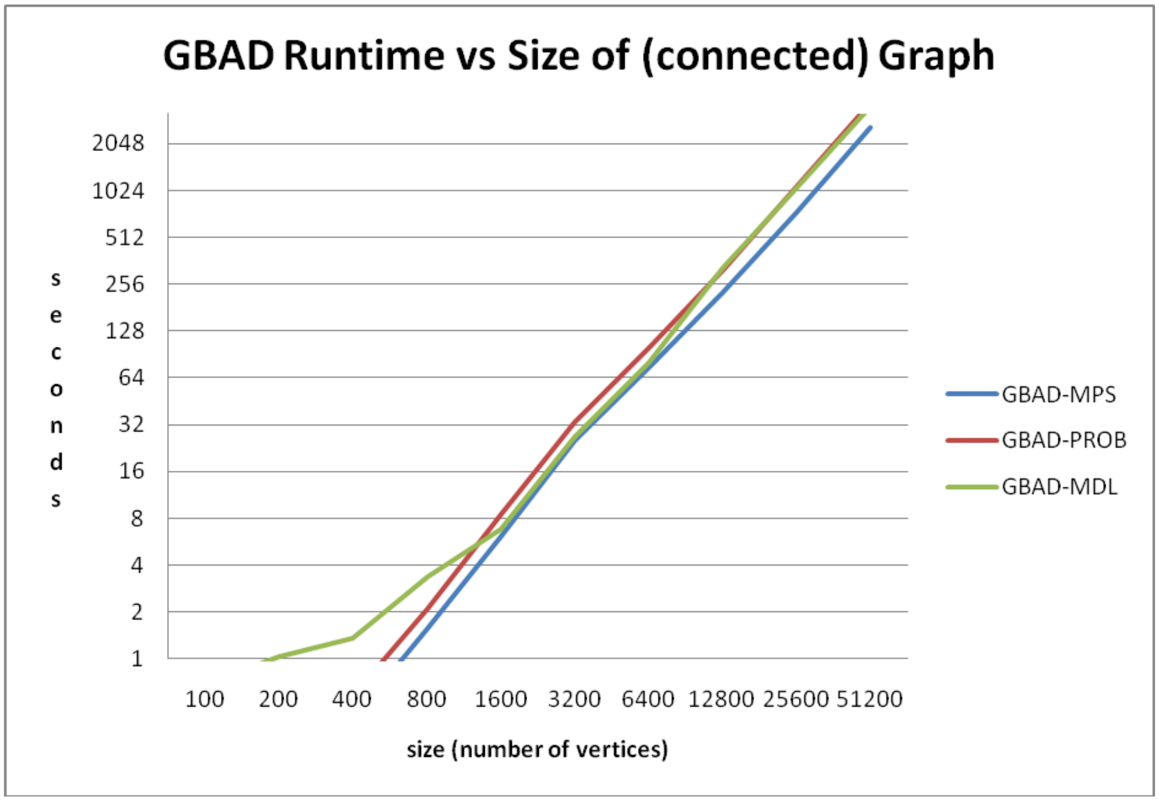

Fig. 22 GBAD running time as a function of graph size (number of vertices) for <u>connected</u> graphs. This log-log plot shows that GBAD scales polynomial with graph size.

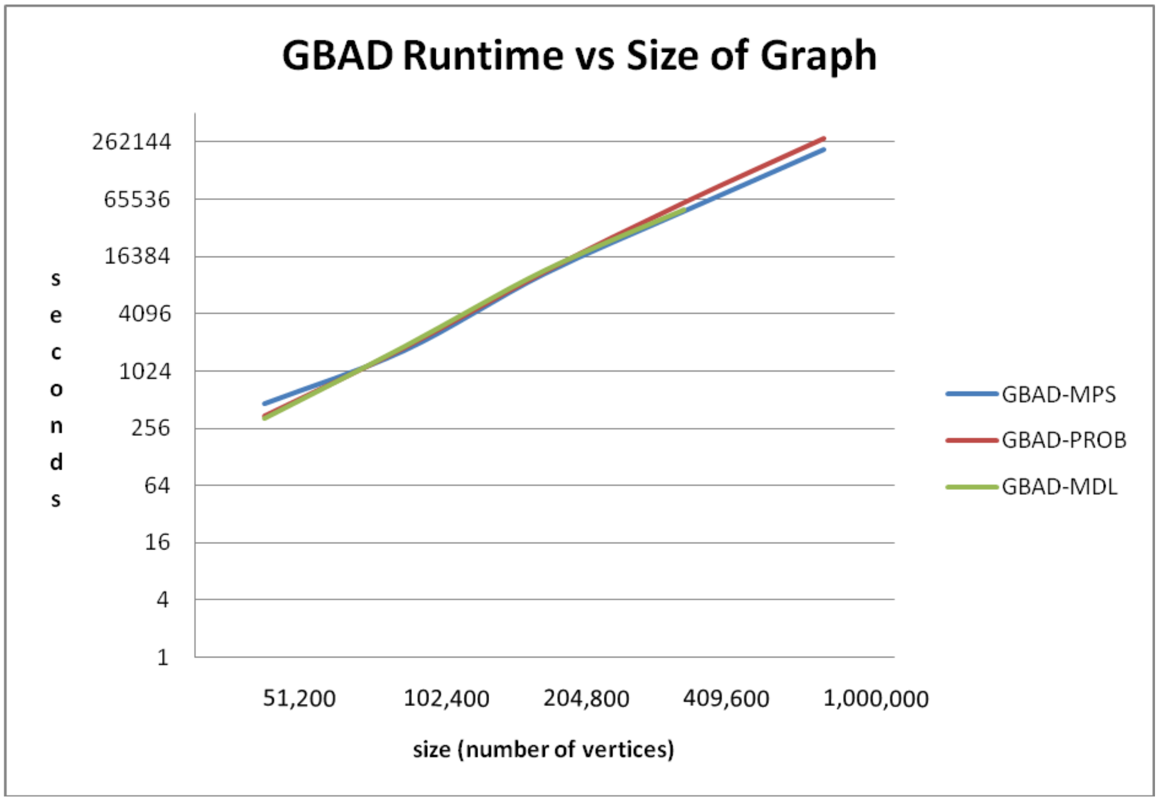

Fig. 23 GBAD running time with increasing input graph size.

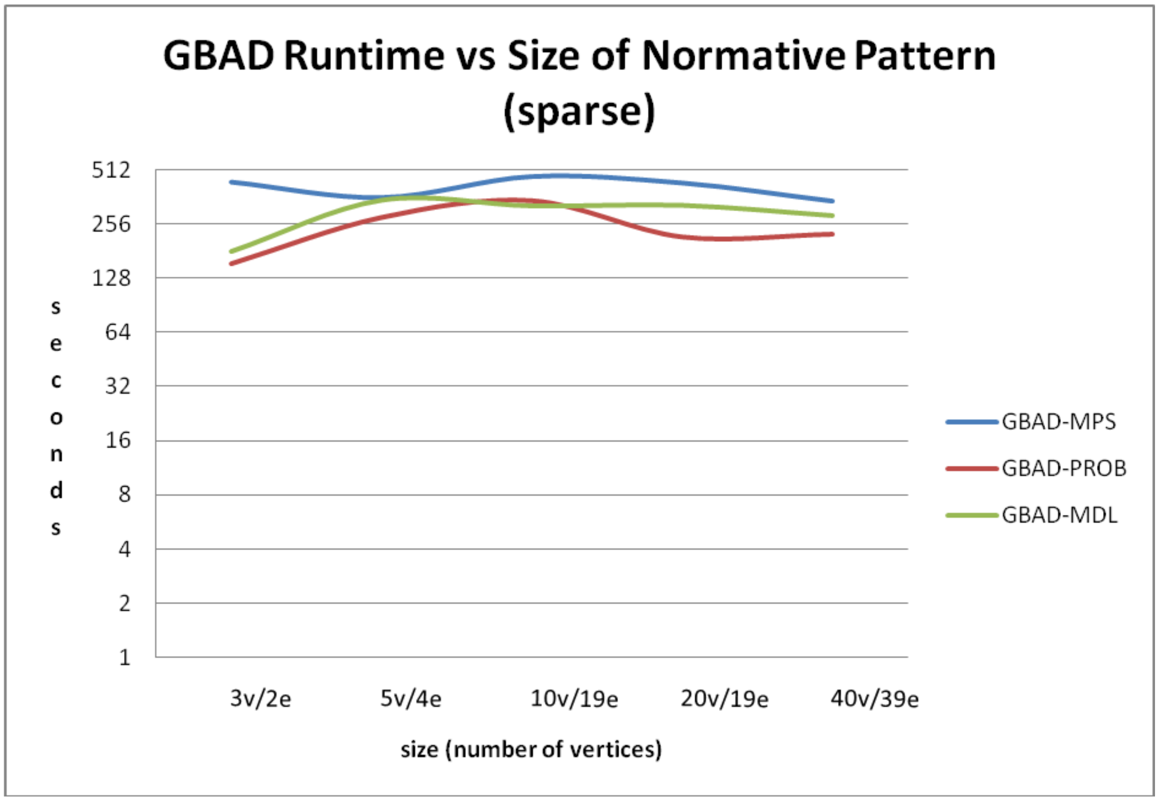

Fig. 24 GBAD running time with increasing normative pattern size on a *sparse* graph with 60,000 vertices.

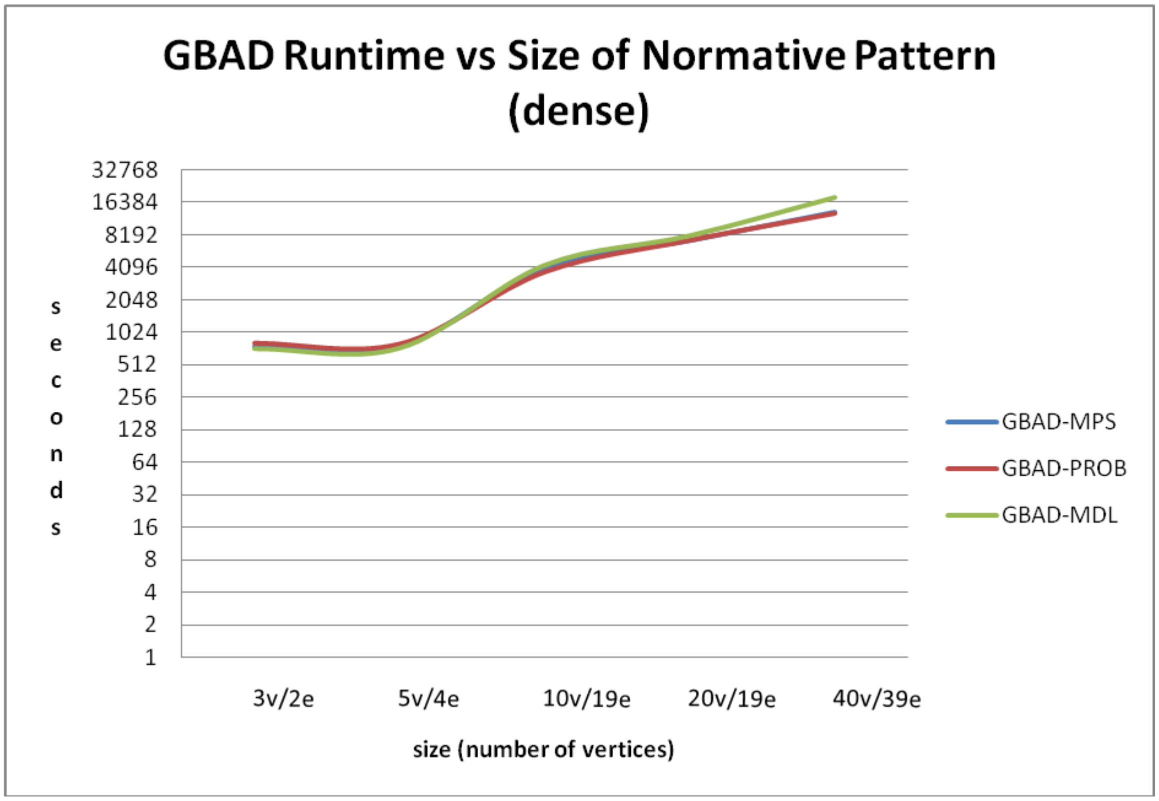

Fig. 25 GBAD running time with increasing normative pattern size on a *dense* graph with 53,900 vertices.

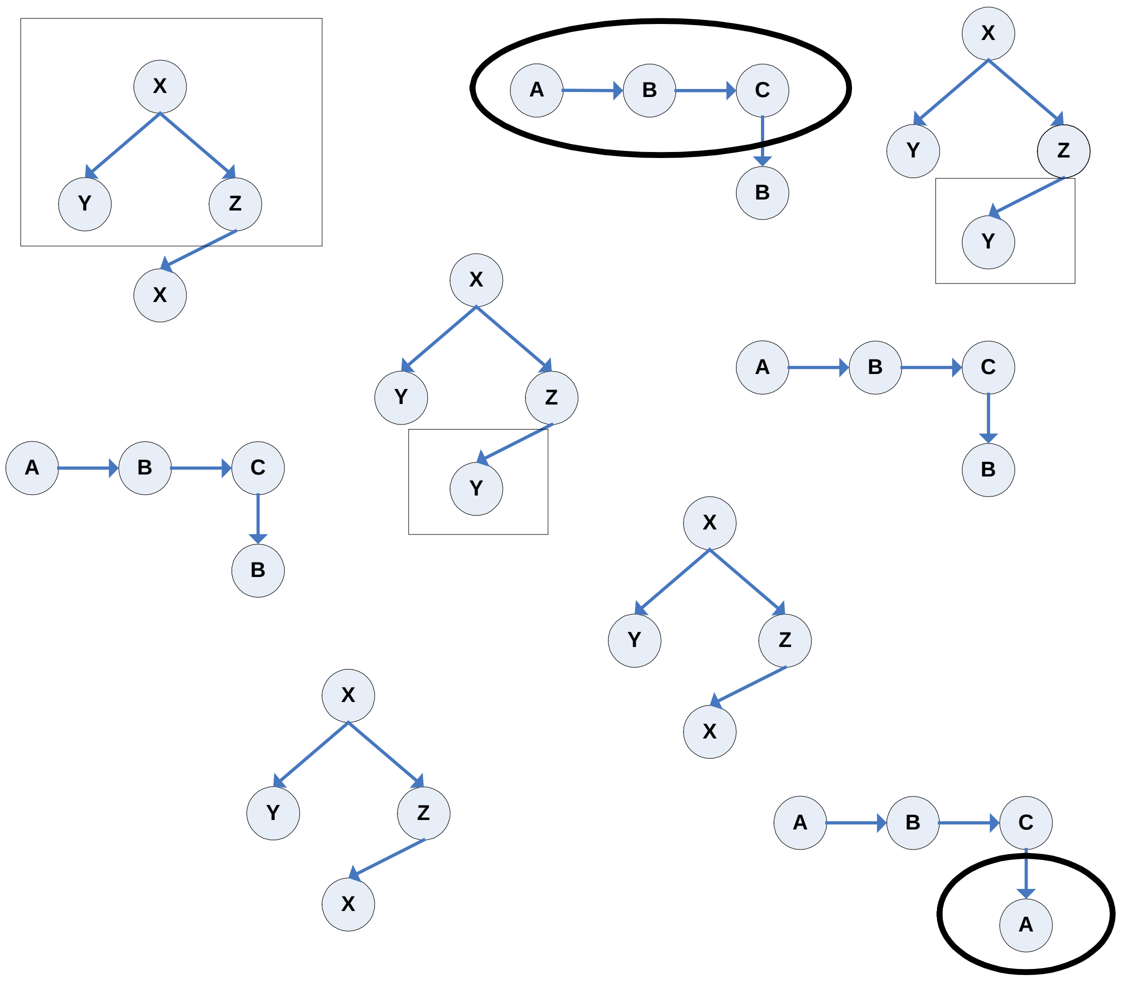

Fig. 26 Example of multiple normative patterns.

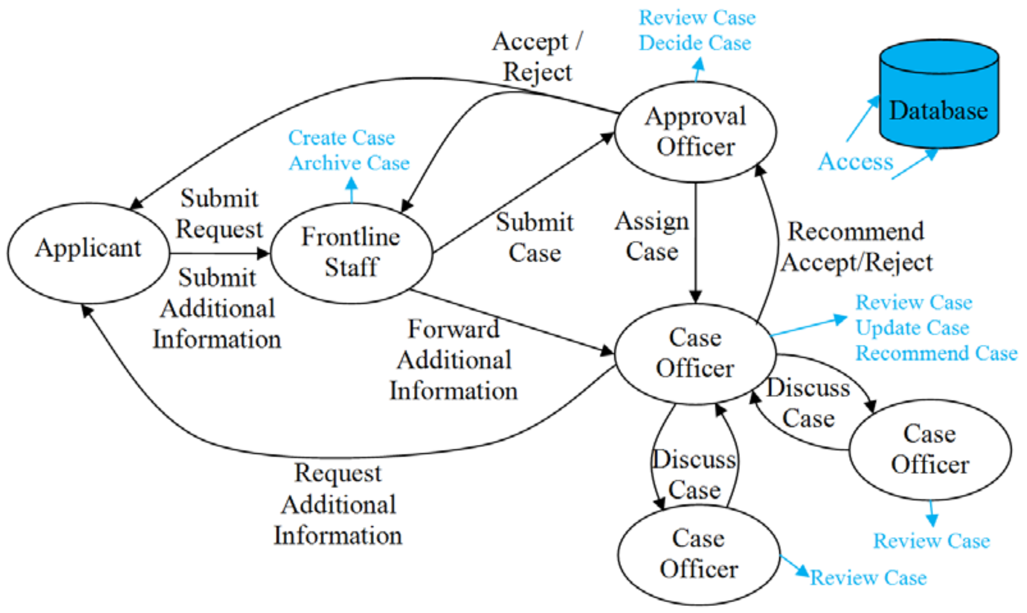

Fig. 27 Depiction of Application Processing.

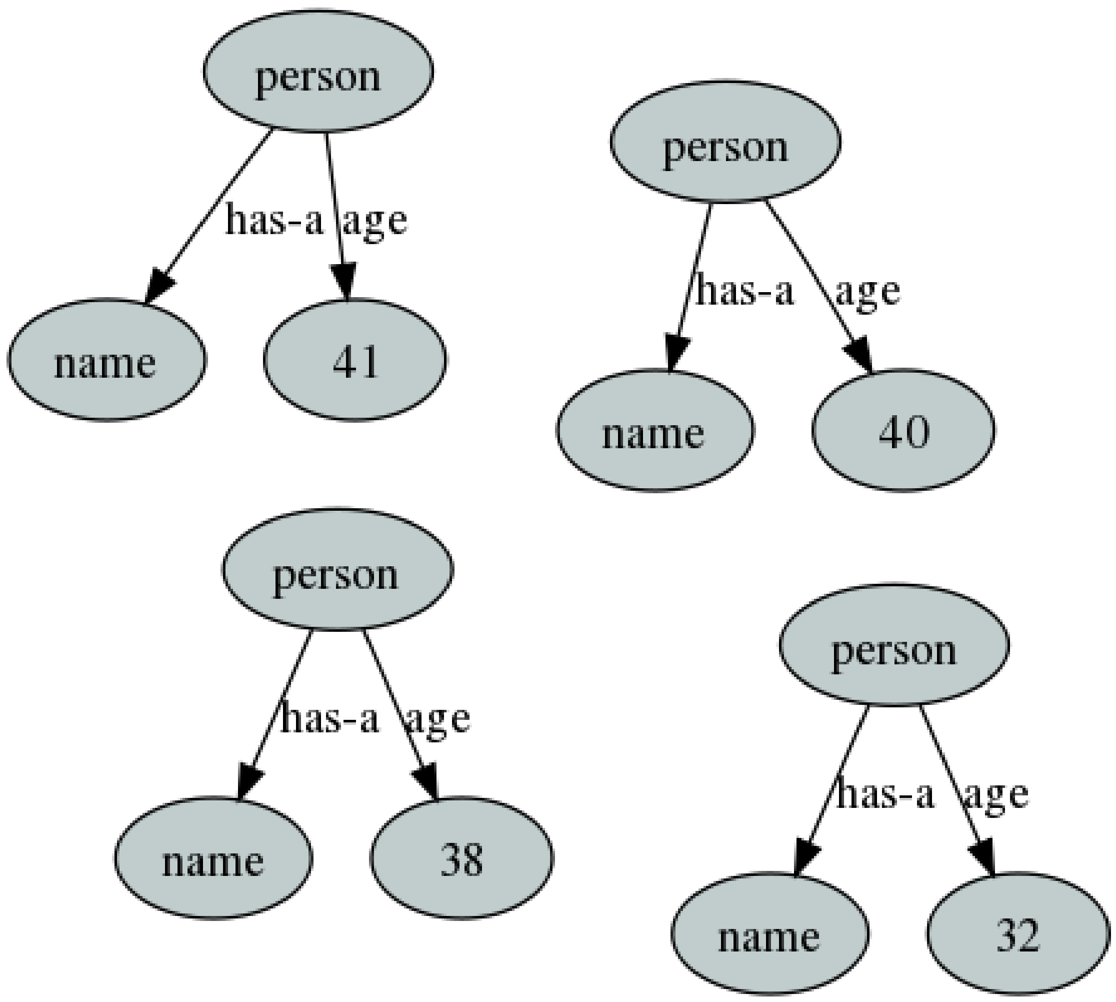

Fig. 28 Example of vertices labeled with numeric values.

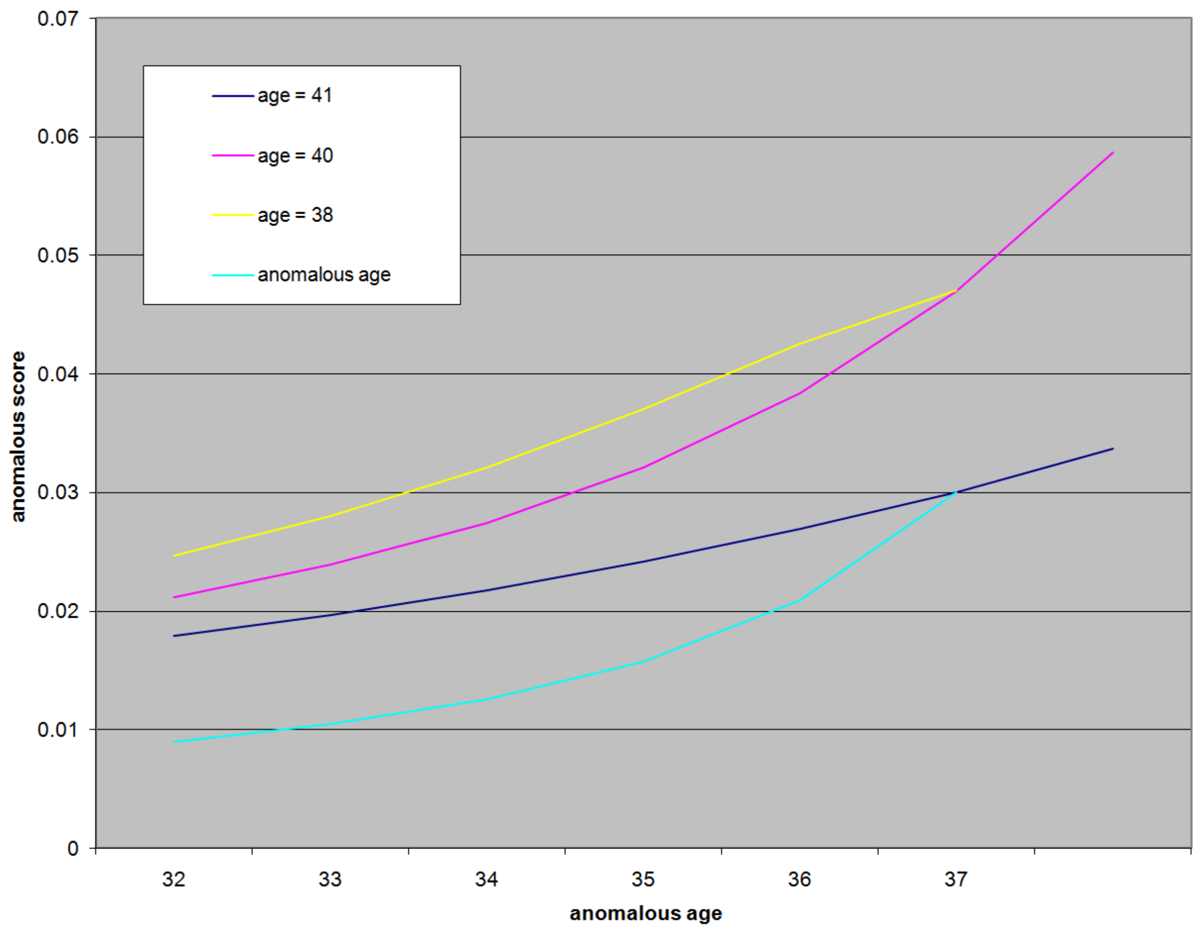

Fig. 29 Numeric deviation effecting anomalousness.

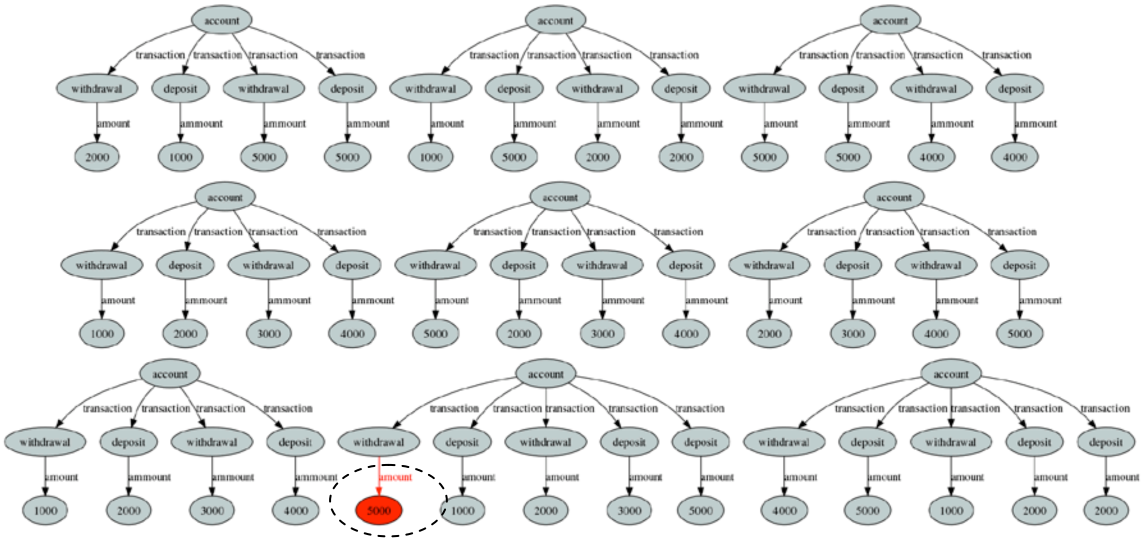
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
2000 1000 5000 5000
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
1000 5000 2000 2000
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
5000 5000 4000 4000
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
1000 2000 3000 4000
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
5000 2000 3000 4000
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
2000 3000 4000 5000
account
transaction transaction transaction transaction
withdrawal deposit withdrawal deposit
amount ammount ammount ammount
1000 2000 3000 4000
account
transaction transaction transaction transaction transaction
withdrawal deposit withdrawal deposit deposit
amount amount amount amount amount
5000 1000 2000 3000 5000
account
transaction transaction transaction transaction transaction
withdrawal deposit withdrawal deposit deposit
amount amount amount amount amount
4000 5000 1000 2000 2000

Fig. 30 Example of anomalous bank transactions.

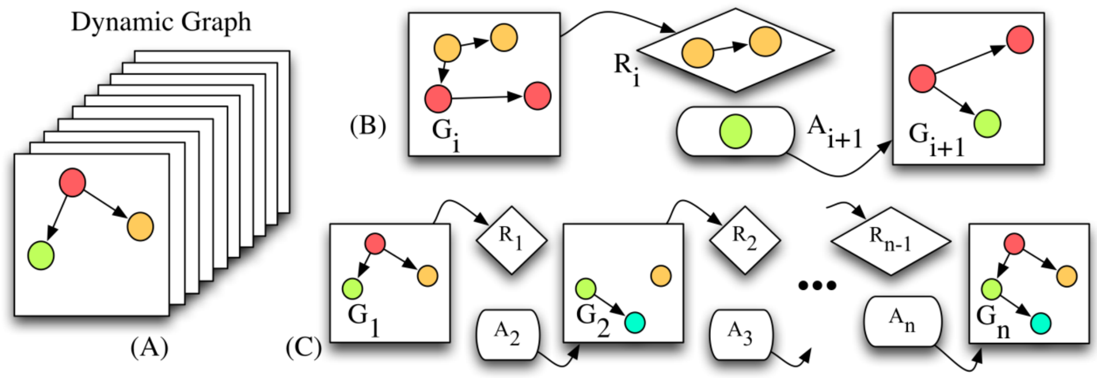

Fig. 31 Framework of dynamic graph analysis. (A) Dynamic graph. (B) Learning graph rewriting rules from two sequential graphs. (C) Learning entire set of graph rewriting rules.

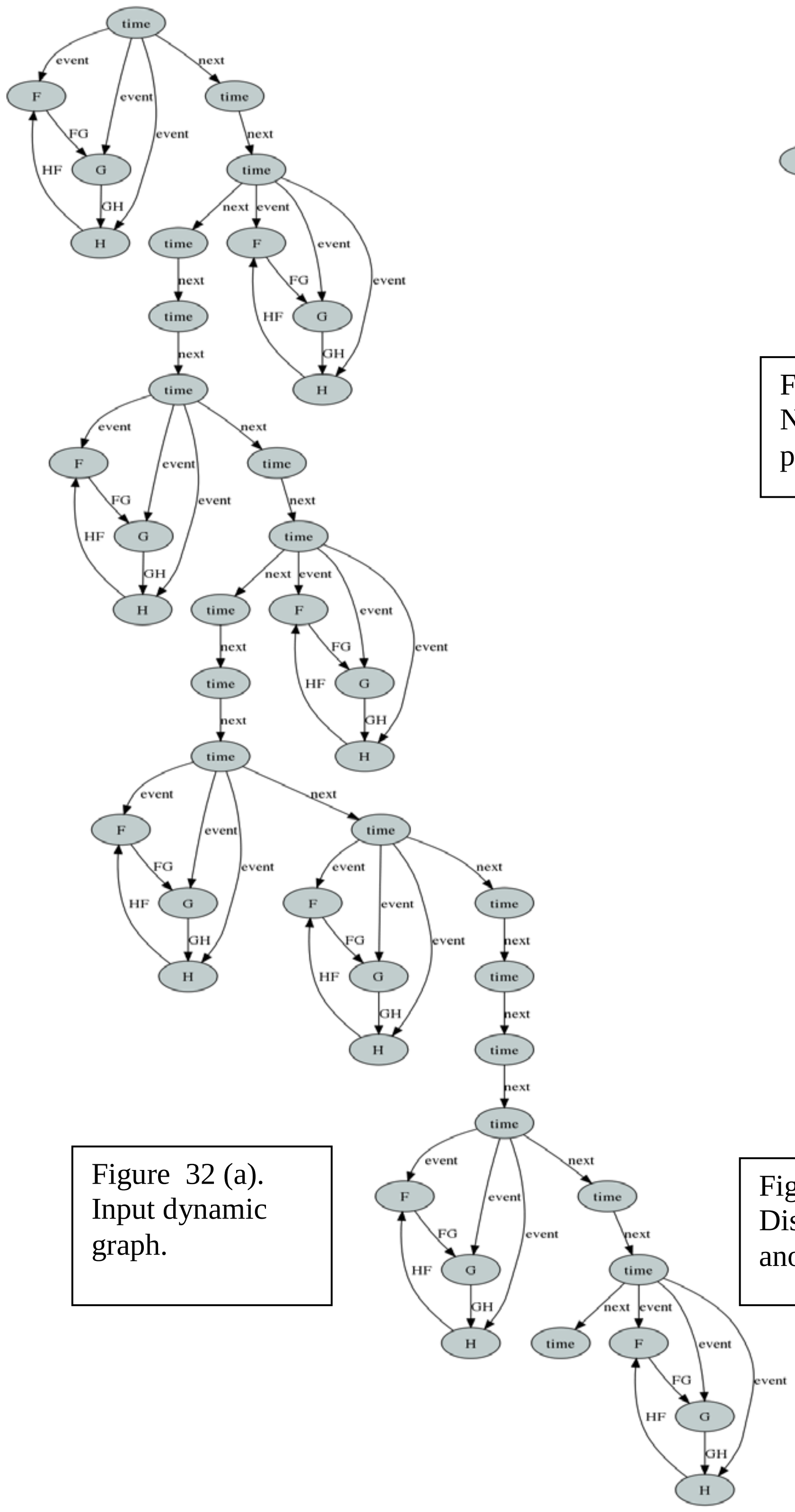

Figure 32 (a). Input dynamic graph.

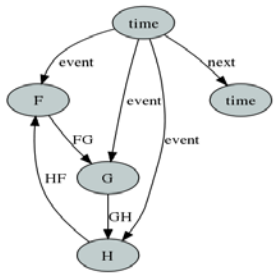

Figure 32 (b). Normative pattern.

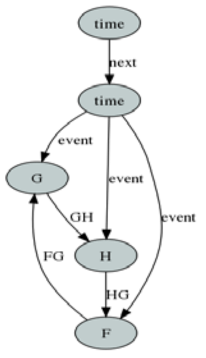

Figure 32 (c). Discovered anomaly.

Fig. 32 Simple example of DynGBAD.

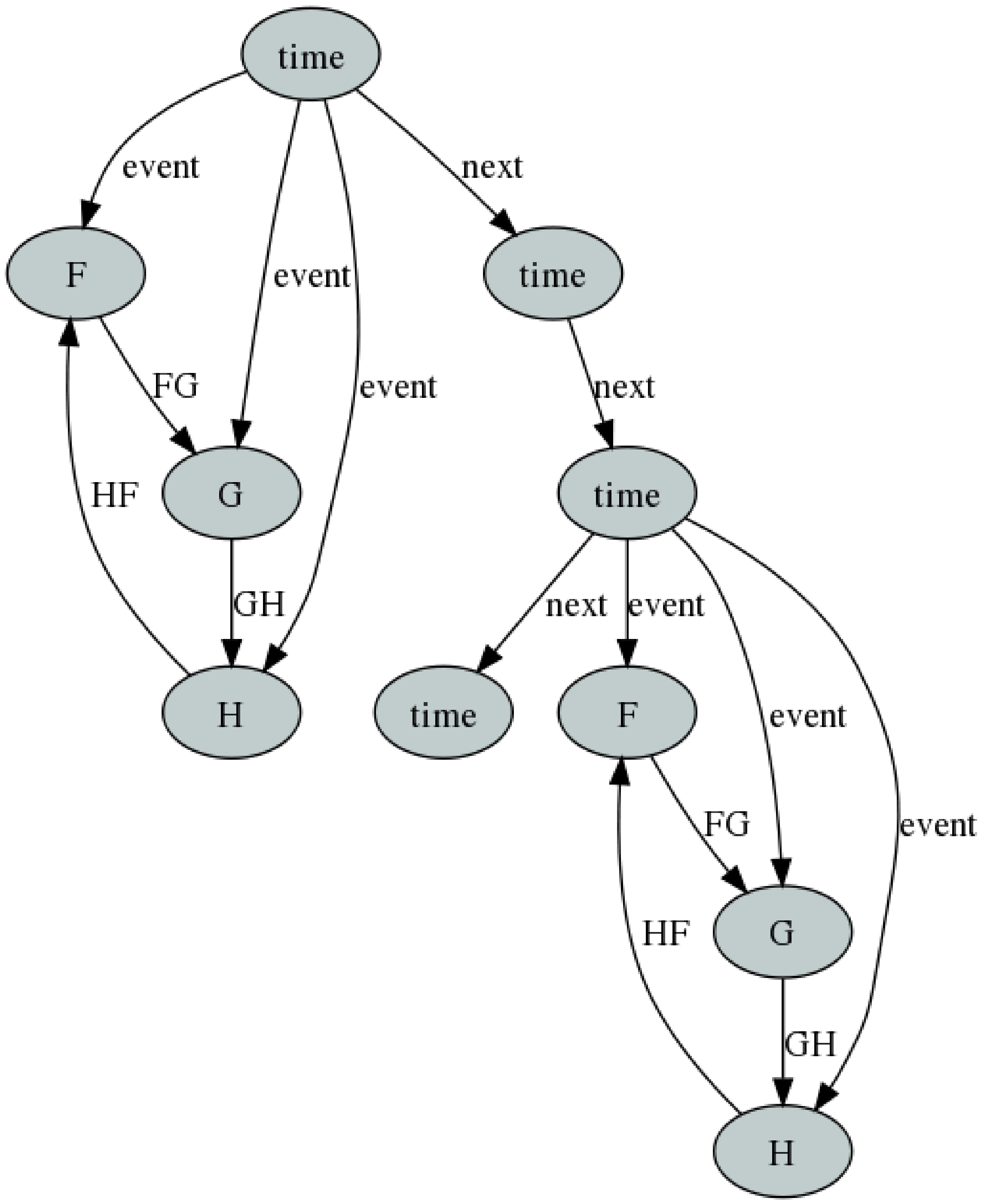

Fig. 33 Normative pattern found by DynGBAD on extended dynamic graph.

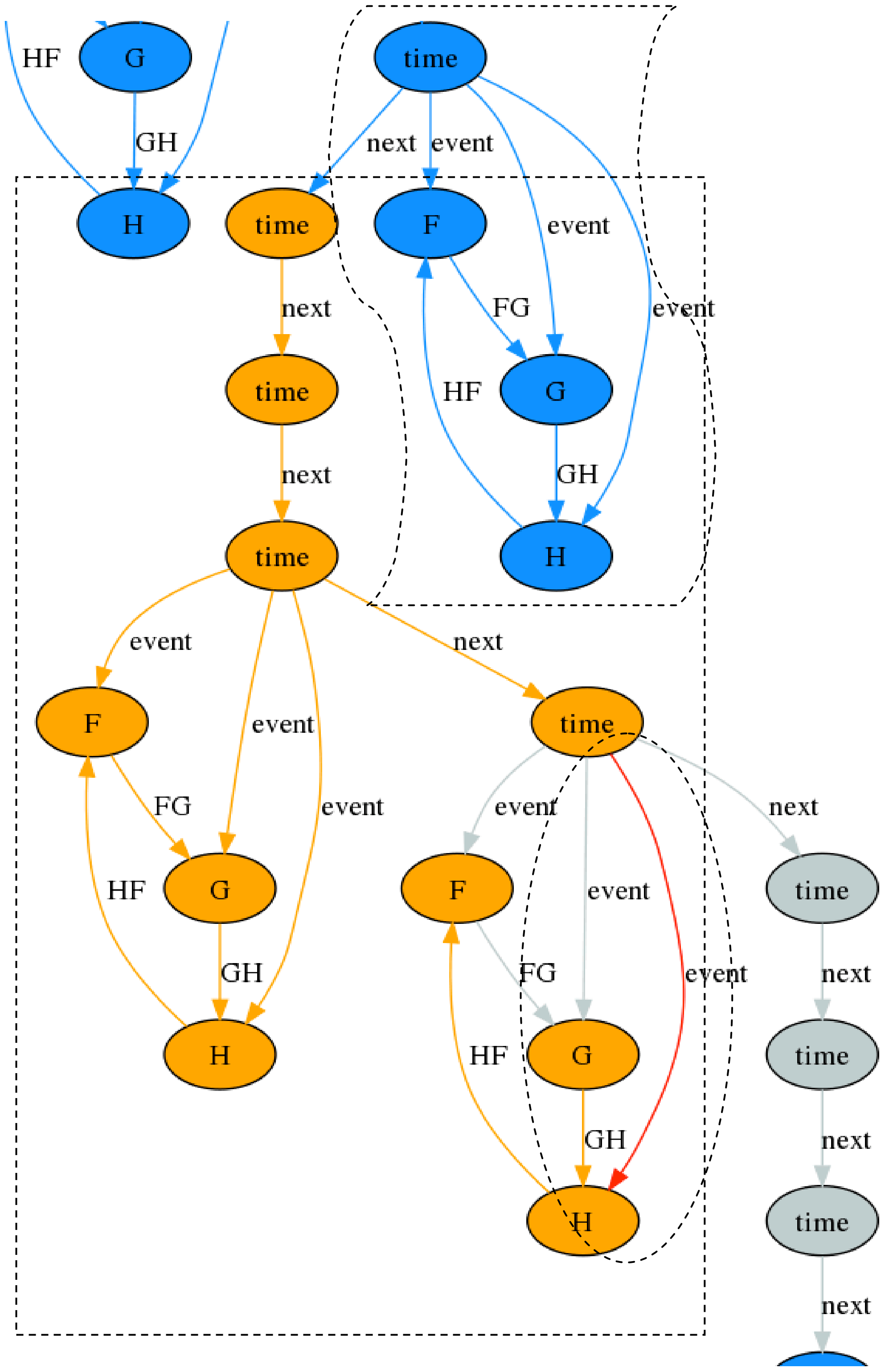

Fig. 34 Anomaly found by DynGBAD on the extended dynamic graph. The structure in orange (inside the dashed-line-box) shows the occurrence of two FGH sugbraphs in consecutive time slices, which is the correct anomaly.

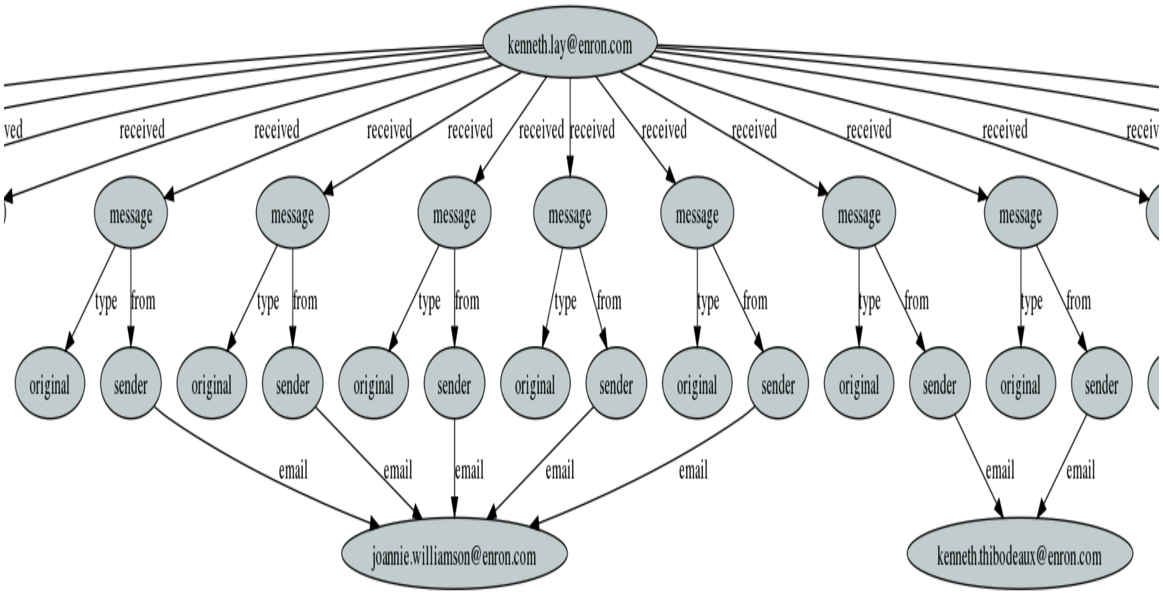

Fig. 35 Graph of emails involving Kenneth Lay on October 10, 2001.

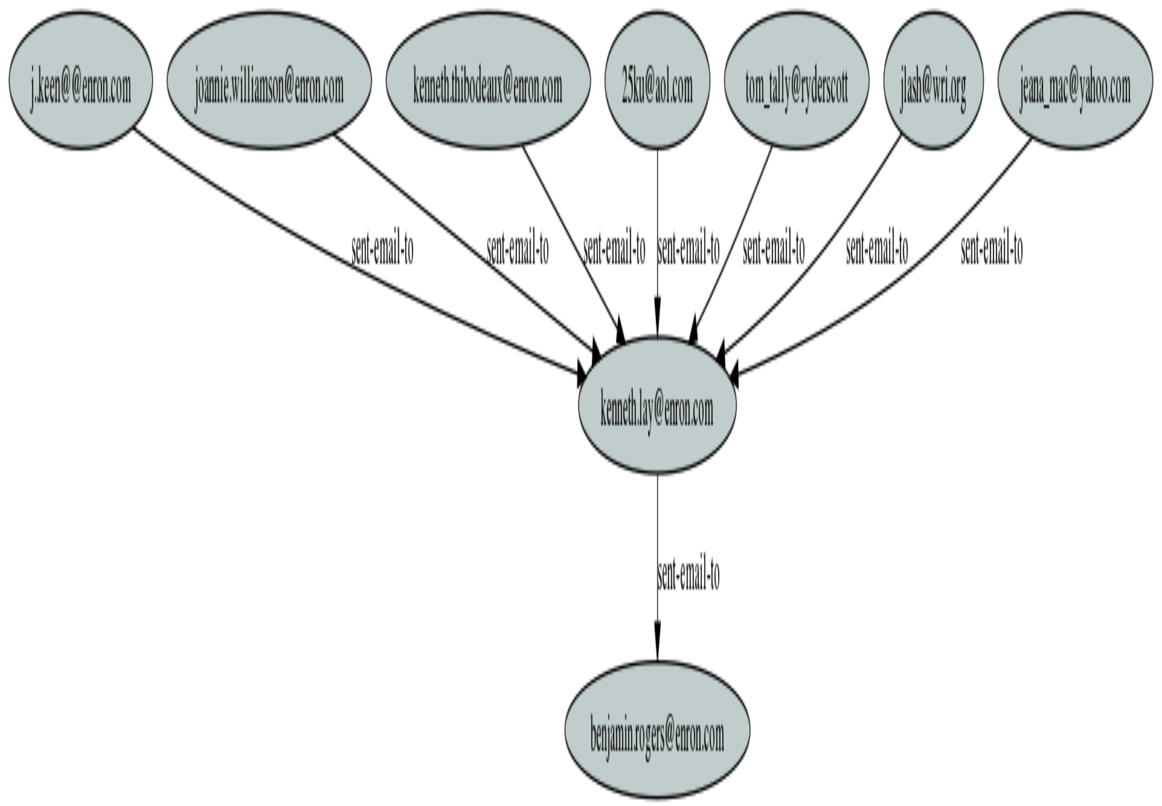

Fig. 36 Alternative graph of information with only employees represented as nodes in the graph.