

Graph Mining using Graph Pattern Profiles

Sofus A. Macskassy, Claude C. Nanjo
Fetch Technologies, 841 Apollo Ave, El Segundo, CA 90245
{sofmac,cnanjo}@fetch.com

Submitted to: ICAI 2008 - Knowledge Bases for Learning

Keywords: Graph mining, Data mining, Data Analysis, Network Analysis, Statistics

Abstract

This paper presents our investigation into graph mining methods to help users understand large graphs. Our approach is a two-step process: First calculate subgraph labels and then calculate distribution statistics on these labels. Our approach is flexible in that it can identify a range of patterns from very abstract to very specific (e.g., isomorphisms). The statistics that we calculate can be used to find rare and common patterns, patterns that are (dis)similar to the distribution of induced subgraphs of the same size, patterns that are (dis)similar to each other, as well as variance of graph patterns given a specific set of input node types. We also investigate a method to understand structural characteristics by analyzing clusters that are created by “collapsing” overlapping instances of user-specified patterns. We evaluated our approach on two publicly available networks—the Texas CS web-site from WebKB and the internet movie database.

1. Introduction

Most work in graph mining and pattern finding has focused on finding (nearly) exact matches of graph patterns within one or more graphs, where a pattern is defined as an induced subgraph consisting of (un)labeled nodes and explicit linkages between them [5, 7, 15, 13, 1]. One limitation with these is that they are generally too exact—they do not allow for graph-based similarity scores beyond graph-edit distance. These methods look for explicit graph patterns, such that two connected subgraphs consisting of the same types of nodes but with very different connectivities would be deemed different. A simple example, for illustration purposes, is shown in Figure 1. The four patterns depicted in that figure would be considered as four distinct patterns, where the last pattern (bottom right) is a superset of the three others. Therefore any of the first three patterns would only count as a match on the pattern itself and pattern (d), but not all four. While existing methods can take into account graph-edit distances and be more encompassing, they generally cannot handle the question addressed in this paper: what are the kind of connectivities observed for a given set of nodes (e.g., actor, editor and movie) and how

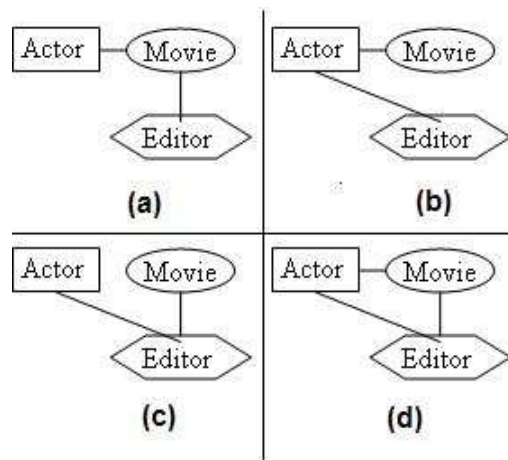


Figure 1. The four possible graph patterns of the “actor movie editor” graph signature that current graph mining techniques would consider being distinct with no reasonable measure of similarities between them.

well do these observed connectivities fit the “expected” connectivity of other sets of nodes or the graph in general. This limitation is due in part to the fact that the predominant way of comparing patterns is by use of graph edit distance, a measure we don’t feel is particularly good in a larger statistical sense. The reason for emphasizing this is that these methods are somewhat brittle and can be sensitive to noise and missing data (which show up as extra or missing edges). Another limitation is that this makes it difficult to find general interactions between types of nodes—for example it may be that all four types of patterns shown in Figure 1 show up regularly in a graph but that this may not be as evident if we need to consider them all as separate types of patterns. We address these two limitations in this paper.

This paper focuses on finding interesting *signatures*, a set of nodes of specific node types that are connected in some way or another as depicted in Figure 1. More formally, signatures are connected subgraphs, or clusters, that ignore *how* they are linked except for the fact that all the nodes are connected. The work presented here also considers the general structure of those connections (e.g., is it a clique or a star-graph, or any other of the possible graph-

structures for that size of nodes, regardless of where each node may be placed in the given graph structure.) Our approach is based on generating *pattern profiles* for given signatures and calculating statistics over these pattern profiles. For example, one use of these is to generate a distribution over observed profiles for a given signature. These are compared with the overall distribution of profiles over the whole graph, enabling us to compute general similarities between observed patterns of different groups of nodes. This is similar to recent work on calculating network motifs [11], wherein they compare patterns in an observed network to patterns in a random network. This paper focuses on using *degree profiles* as the pattern profile, but we note that our approach is general and that it can be used to capture graph statistics for a wide range of abstractions, both on a general level as we described above as well as at the explicit level that current graph mining techniques work at. Our approach facilitates computing probabilities on signatures and distributions of observed instances of signatures. For example, for clusters of size n , we can compute the likelihood of seeing a particular signature or specific pattern profile. In addition, being able to calculate empirical distributions of observed patterns enables the use of similarity measures between clusters of nodes—for example, do actors, editors and movies, when connected in closed a subgraph, cluster in different (or similar) patterns than what its expected from a random closed subgraphs of size three. As far as we know, these are new capabilities that are not possible with current graph mining techniques.

The rest of the paper is outlined as follows: We next describe related work, followed by the formal description of our graph mining approach and techniques. We then perform two case studies. The first study is on the Texas Computer Science web-site from the WebKB dataset, where we investigate mining methods for finding common, rare, and distinct/general patterns. The second study is on a subset of the internet movie database (www.imdb.com) data, where we apply our algorithm to first find common patterns and then analyze the clusters formed by instances of these patterns. We conclude with a discussion of the results and limitations of our approach.

2. Related work

The field of graph based data mining (GBDM) has received a lot of attention and publications over the last ten years [7, 9, 10, 14, 15]. GBDM has generally focused on the problem of finding frequent subgraphs—i.e., all subgraphs in the graph who have a minimum amount of support. There is a case to be made that graph-based relational learning (GBRL) (cf. [6]) is generally more useful in that GBRL tries to find useful patterns in some information theoretic sense instead of just mining for frequency. GBRL has seen little attention and only a few approaches have been developed so far, the two most developed ones being Subdue [2] and GBI [16]. For a good overview of GBDM we refer the reader to Washio and Motoda [13]; Holder and Cook [6] have a similarly good review of GBRL.

The core problem of both GBDM and GBRL is the search for common subgraphs and count the number of can-

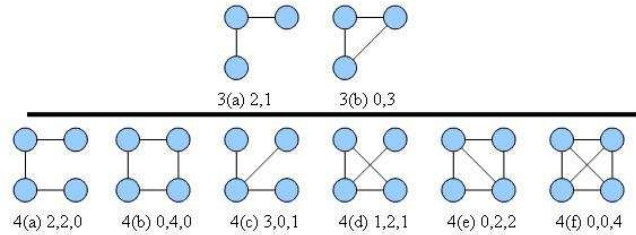


Figure 2. Possible degree profiles of clusters of size 3 and size 4. Clusters of size 3 only have two possible degree profiles; clusters of size 4 have 6 possible degree profiles. Degree profile 4(c) is “3,0,1” because 1 node has a degree of 3 and the other three nodes have a degree of 1.

didate subgraphs that appear in the data. Some problem domains have many graphs, where the task is to count the number of graphs where a specific pattern appears [15]. In other problem domains, such as the one in this paper, there is one large graph and the task is to count how many times a pattern appears in that graph [10].

Similar to GBDM is recent work on Network Motifs [11], where the focus is to mine a single large graph for patterns of size n that are different from what would be expected from a random graph. The approach taken is to generate a random graph that has the same distribution of patterns of size $n - 1$ as the graph to be mined. The two graphs (the source graph and random graph) are then compared to find patterns of size n in the original graph that appear more frequently than in the random graph.

3. Graph Mining using Signature Profiles

As mentioned above, one limitation of current graph mining techniques is that they focus on very explicit graph structures in such a way that it is difficult to identify and mine for more general types of patterns. Our approach is able to mine for general patterns that match on each of the four patterns in Figure 1, based on the specificity level of the patterns that are being mined for. We here provide the methodology for more general patterns but note that we can easily provide mining capabilities to the same level of specificity as current mining methods, by using a more explicit graph representation of graph patterns. We will highlight below where the level of specificity can be set.

Our approach works in two steps. The first step is to define the subgraph representation to use to calculate our graph statistics, and the second step involves calculating the actual statistics on the graph. We describe each of these below. We then describe the algorithms we use to generate and count subgraphs and finish by briefly describing the tool we used to perform the case studies.

3.1. Graph Signatures and Labels

The focus in this paper is on the identification of general cluster-patterns—clusters of nodes (connected subgraphs) containing a specific set of node-types, which we call a *signature*. An example signature is one actor, one movie, and one editor which would match any of the subgraphs shown in Figure 1 above. The key idea is to count the number

of induced subgraphs of a given signature, where the given types of nodes are connected in some way. Figure 1 shows the four possible ways three specific nodes of the example signature just given could be connected.

The specific approach we take is guided by the observation that a subgraph of a specific size (number of nodes) can take on a fixed number of *configurations* or *patterns*. One naive and lossy way of representing these patterns is using an n -dimensional *degree profile*—calculating the degree of each node and storing, for each possible degree $0 < d < n$, how many nodes had that degree. Figure 2 shows the possible degree profiles of subgraphs of size 3 and size 4. Notice that we remove any node-type information because we are looking for generalities and are less interested in stricter patterns as is usual for standard graph mining.

The reason to focus on degree profiles is twofold:

1. The degree profile is an approximation of what is known as the canonical label of a graph [4, 12, 8, 10]. A canonical label of a graph is a unique label which represents a specific graph pattern such as those shown in Figure 1. The degree profile approximation is much faster to calculate than the canonical label and enables more general graph matching. However, our approach also facilitates deeper statistical analysis using canonical labels as the representation to generate statistics over. This makes our approach very flexible.
2. The degree profile focuses on the general connectivity of a graph without the exact placement of the various nodes. For example, the first three patterns in Figure 1 all have the same degree profile and would be indistinguishable from each other using only the degree profile. An added benefit is that we can calculate similarity metrics in ways not possible before.

Note, that we can represent a given subgraph to an arbitrary level of detail, from the degree profile we use in this paper to the canonical representation that is often used in graph mining. The graph statistics we calculate (described below) are agnostic to the graph representation.

3.2. Statistics on Graph Patterns

We present our statistics in the context of mining for subgraphs of size n .¹ Specifically, we focus on calculating statistics over *signatures* as described in the previous section. For example, consider the case where $n = 4$ and using the degree profile as the labeling representation. There are 6 possible degree profiles for a signature of size 4 as depicted in Figure 2.² Using degree profiles of subgraphs, we compute an empirical distribution over the number of times each degree profile is observed among the instances of a specific signature. For example, a graph of size 4 has 6 possible degree profiles, and the empirical discrete distribution of a signature of size 4 will therefore be represented as a vector of size 6.

¹Our method can be used to analyze subgraphs for any $n > 3$. Each subgraph size will necessarily have its own empirical label distribution.

²The number of possible patterns would obviously increase if we were to use canonical labels, but our statistics would still work.

```

procedure FIND(size,  $G$ )
1: Instances  $\leftarrow \emptyset$ 
2: for all  $v \in \text{Vertices}[G]$  do
3:    $S \leftarrow \text{GENERATE}(v, \text{NIL}, \text{Vertices}[G])$ 
4:   Instances  $\leftarrow \text{Instances} \cup \{S\}$ 
5: end for
6: for index = 1 . . . size do do
7:   NewInstances  $\leftarrow \emptyset$ 
8:   for all  $S \in \text{Instances}$  do
9:     for all  $v \in S.\text{Periphery}$  do
10:       $S' \leftarrow \text{GENERATE}(v, S, \text{Vertices}[G])$ 
11:      NewInstances  $\leftarrow \text{NewInstances} \cup \{S'\}$ 
12:     end for
13:   end for
14:   Instances  $\leftarrow \text{NewInstances}$ 
15: end for
16: return Instances

```

Table 1. Count the instances of all patterns of a given size.

We compute similarities between distributions by representing the two distributions as vectors in k -dimensional space, where k is the number of possible labels a subgraph of size n can take on ($k = 6$ in the case of using degree profiles on subgraphs of size 4). It is then simple to compute the cosine distance between those two vectors. As noted above, the power of our approach is that this distribution can be calculated for *any* graph-labeling scheme. Although the results presented here are based on using the degree profile mentioned above, we note that using canonical labels will yield a superset of the results returned by current graph mining methods (each unique graph as shown in Figure 1 will be represented). Our method will calculate distributions over all observed canonical labels and do not currently limit their search as current methods do (so as to generate proper distributions and facilitate comparison to the overall graph structure). As such, by changing the labeling scheme we can vary the level of mining capabilities.

3.3. Generating and Counting subgraphs

Tables 1 and 2 show the algorithms we use to find the subgraphs from which we calculate the pattern representations and subsequently the distributions. The primary algorithm, FIND (Table 1), starts by instantiating all singleton nodes as their own subgraphs (lines 2-5). It then iteratively expands the subgraphs one neighbor at a time until we have identified all unique instances of the given input size (lines 6-14). Each subgraph, keeps track of its periphery—the list of neighbors that it can expand into (shown in the GENERATE algorithm in Table 2). We would like to re-emphasize that our method can be used to analyze patterns of size 3 to size n but that we here focus on patterns for a specific n .

One subtle, but very important, point of our Periphery and NewInstance data structures is that they are *sets*. This is important because this means they will only keep one copy of a specific set of nodes or labels (e.g., $\{a, b\}$ which is reached both from growing set from $\{a\}$ as well as from $\{b\}$ will only be counted once).

Another important fact is that when we compute the label for a specific set of nodes, we consider only the induced subgraph (e.g., if pattern 4(e) in Figure 2 is the complete

```

procedure GENERATE( $v, S, \text{Vertices}$ )
1: Periphery  $\leftarrow \emptyset$ 
2: Core  $\leftarrow \emptyset$ 
3: if  $S \neq \text{NIL}$  then
4:   Periphery  $\leftarrow \{ S.\text{Periphery} \}$ 
5:   Core  $\leftarrow \{ S.\text{Core} \}$ 
6: end if
7: Core  $\leftarrow \text{Core} \cup \{v\}$ 
8: if  $v \in \text{Periphery}$  then
9:   Periphery  $\leftarrow \text{Periphery} - \{v\}$ 
10: end if
11: Periphery  $\leftarrow \text{Periphery} \cup \{ \text{NEIGHBORS}\{v\} - \text{Core} \}$ 
12:  $S'.\text{Periphery} \leftarrow \text{Periphery}$ 
13:  $S'.\text{Core} \leftarrow \text{Core}$ 
14: return  $S'$ 

```

Table 2. Expand subgraph S by adding vertex v .

connectivity before 4 nodes, then that is the only degree profile that would be counted. Patterns 4(a), 4(b), and 4(c) would not be considered). This is done because we are more concerned with signatures and how they fully appear in the graph than in explicit sub-patterns.

3.4. GraphExplorer Tool

We are developing a graph mining tool, GraphExplorer, in which we have implemented the algorithms and statistics described above. This tool has helped focus the algorithms both in terms of making them practical as well as making the statistics useful.

The tool is built on top of the Java Universal Network/Graph Framework (JUNG).³ Some of the key functionalities of the platform is to facilitate mining for interesting patterns. In the context of this paper, this involves mining a given graph for subgraphs of a specified size n . We then group and rank these subgraphs using the measures we have described above. These provide multiple “views” of a graph, each one offering some additional insights about interesting interactions between particular (sets of) nodes.

The primary goal of this tool has been to provide users with a rapid way to find “seed” patterns which can then be used to focus in on interesting parts of the graph or get better insight into the overall graph structure and interactions between nodes.

The case studies below were done using GraphExplorer.

4. Case Study I - WebKB

Our first case study is in the domain of web-sites. Specifically, we focus on a web-site from computer science, where we analyze how different types of pages (e.g., student and project pages) connect together. We analyze various signatures and use our distributions to ask questions such as what are signatures that are deviant.

4.1. Data

We here use a data set from the WebKB Project [3].⁴ It consists of web pages from four computer science departments, with each page manually labeled into 7 categories: course, department, faculty, project, staff, student or other.

³<http://jung.sourceforge.net>

⁴We use the WebKB-ILP-98 data.

Category	Number of web-pages
course	54
faculty	50
project	29
staff	6
student	163
Total	302

Table 3. Distribution of labels for the Texas graph.

faculty	project	project	student	9.68%
course	faculty	project	student	7.36%
course	faculty	faculty	student	7.36%
faculty	project	student	student	6.96%
faculty	project	staff	staff	0.02%
course	project	staff	student	0.02%
course	faculty	staff	student	0.02%
course	course	course	course	0.02%

Table 4. Ranking signatures by frequency.

This study uses the Texas dataset. We excluded from consideration all nodes and edges involving department nodes. The resulting graph consisted of 302 nodes and 529 edges. There were 5 labels: project, course, student, faculty, staff nodes. Table 3 shows the distribution of labels.

4.2. Experimental Methodology

The Texas WebKB graph is a directional graph. For the purpose of this experiment the directionality was ignored, resulting in an undirected graph.

For our case study we focus on signatures of size 4. 4306 subgraphs and 45 distinct signatures were identified. These served as the starting point of our analysis. The set of all subgraphs was partitioned into 45 clusters of subgraphs each containing elements bearing the same signature.

4.3. Results

The goal of this experiment was to find “interesting” signatures. A signature is deemed to be of interest if it is either very common or quite rare in the graph, if it resembles numerous other signatures based on some similarity measure (cosine in this paper) or diverges greatly from them, and whether it’s instance subgraphs are highly homogeneous or highly heterogeneous. We will examine each of these in detail below.

4.3.1 Common and Rare Signatures

We first ranked signatures by their relative frequency in the overall set of subgraphs of size 4. Table 4 shows the most frequent and rare subgraphs. The ranking reveals which association of node types is most common and least common in a graph. However, it does not reveal how these nodes are associated to one another and therefore provides an important albeit incomplete view of a graph. For example, the signature “faculty project project student” is the most common pattern among these subgraphs, yet we do not know from this ranking whether the association between the nodes is a star pattern with the faculty at the center or whether the nodes are associated in a linear fashion.

course	course	course	student
faculty	faculty	faculty	project
student	student	student	student
course	course	course	faculty
course	course	faculty	faculty
course	faculty	faculty	faculty
faculty	faculty	faculty	student

Table 5. Homogeneous signatures; all these signatures were only found as star-patterns (4(c) in Figure 2).

Interestingly, but not surprisingly, most of the rare signatures were ones containing one or more staff. This is obvious in retrospect because there were so few staff nodes in the graph. This is something that should be taken into account in the future.

4.3.2 Deviant Signatures

We identify deviant signatures by how different their distributions are to the overall distribution of all n -subgraphs. Ideally we would like to be using a statistical test such as Chi-Squared to test the likelihood that the distributions come from the same underlying distribution. It was invariably the case that the more observations we had, the smaller this likelihood was although the distributions were qualitatively very similar. More importantly, the ordering of deviants did not produce good results. We therefore use the cosine similarity measure, which resulted in a better (subjective) ordering.

In this study, the distribution for a given signature was compared to the overall distribution for all instances of subgraph of size 4. After ranking the signatures, we found the most deviant signatures to be the most homogeneous ones (i.e. specific signatures that always had the same single degree profile). In fact, we found that all of the clusters which had only one type of degree profile all turned out to be star patterns. Table 5 shows these signatures. Except for one signature (“course course faculty faculty”), all signatures had 3 of one type of node and the last node being different. A random sample of these verified that it was always the odd node that was in the middle of this star pattern.

4.3.3 Distinct vs. General Signatures

In order to evaluate the homogeneity of n -graph instances within a single signature we analyzed the degree profile distributions for each signature using the following approach. We first ordered the degree profile distribution histogram in such a way that the bin containing the highest number of instances appeared first, the second most numerous bin second, etc. We then proceed through the bins in order until over 90% of all instances for the cluster were accounted for. The number of bins processed and the percent of instances encountered formed the ranking criteria.

The intent of this measure is to identify signatures whose instances tend toward one or few degree profiles as opposed to signatures that represent a wide range of possible degree profiles. Signatures whose instances tend to be more homogeneous in their degree profile distribution might reveal an inherent structural relationship between their nodes. A snippet listing the most distinct and most general signatures is

```

1: PUSH( $\mathcal{P}, m$ ), where  $m$  is a movie picked at random.
2:  $\mathcal{D} \leftarrow \emptyset$ 
3: while  $|\mathcal{D}| < 1000$  do
4:    $m \leftarrow \text{POP}(\mathcal{P})$ .
5:    $\mathcal{D} \leftarrow \mathcal{D} \cup \{m\}$ 
6:   for all node types,  $t$  do
7:      $i \leftarrow$  random number between one and five.
8:      $S_t \leftarrow i$  neighbors of  $m$  of node type  $t$  picked at random.
9:      $\mathcal{D} \leftarrow \mathcal{D} \cup \{S_t\}$ 
10:    for all  $n \in S_t$  do
11:       $i \leftarrow$  random number between one and five.
12:       $M_n \leftarrow i$  neighbors of  $n$  (which by design are movies), picked at random.
13:      PUSH( $\mathcal{P}, m'$ ) for all  $m' \in M_n$ .
14:    end for
15:  end for
16: end while
17: return  $\mathcal{D}$ 

```

Table 7. Process for extracting a smaller data set from the IMDb database.

displayed in Table 6. Bin (c) is the star pattern, and as mentioned previously, we see that the star-shaped pattern occurs exclusively for several of the patterns of size 4. In fact, the star pattern is the most common pattern that appears as the single all-encompassing pattern. Though the central node (the node of degree 3—all other nodes have degree 1 in the specific subgraph) may vary within the instances of such clusters, such homogeneity is most likely due to some well-established relationship between these types of nodes. For example, the “course course course faculty” signature is arranged in a star pattern.

Also, as expected, for a relatively sparse graph, highly connected profiles such as profiles 4(e) and 4(f) in Figure 2 are not very common. Fully connected cliques (profile f) are rare (4 in this set) and reveal more unusual interactions between nodes (for example, “faculty project project student”). In other cases, these interactions may seem less unusual such as the clique “faculty project project project”. Here one might surmise that a faculty member is involved in three projects that cross-reference one another.

5. Case Study II - IMDb

Our second case study is in the domain of movies. The purpose of this study is to look for interesting signatures that can be “collapsed” into interesting clusters that reveal interesting structural information about the graph.

5.1. Data

We focused in this study on data acquired from the internet movie database (www.imdb.com). The way this data is represented, everything is centered around movies—a movie connects to any non-movie node (e.g., actor, production company, director, etc.) and vice versa. We sampled 1009 nodes from the full IMDb database using the process shown in Table 7.

Table 8 shows the details of the resulting data set, including the specifics for edges. Remember, each edge will by design be between a movie and a non-movie.

pattern				degree profile						profiles processed	proportion of instances processed
				(a)	(b)	(c)	(d)	(e)	(f)		
faculty	faculty	faculty	project	0	0	56	0	0	0	1	1
student	student	student	student	0	0	35	0	0	0	1	1
course	course	course	faculty	0	0	25	0	0	0	1	1
course	course	faculty	faculty	198	0	0	0	0	0	1	1
course	faculty	faculty	faculty	0	0	220	0	0	0	1	1
faculty	faculty	faculty	student	0	0	20	0	0	0	1	1
course	course	course	student	0	0	16	0	0	0	1	1
course	course	faculty	project	0	0	46	1	0	0	1	0.978
.											
project	project	student	student	112	0	76	48	9	0	3	0.963
faculty	project	project	student	161	0	142	96	17	1	3	0.956
faculty	project	project	project	26	0	47	31	4	2	3	0.945
project	project	project	project	4	0	5	4	1	0	3	0.928
project	project	project	student	48	2	26	34	9	1	3	0.9

Table 6. Signatures ranked by homogeneity. The ‘profiles processed’ column shows how many profiles needed to be included in order to include more than 90% of observed instances. The last column shows the ratio of instances that were included in those profiles. The fewer bins, the more homogeneous the signature is (it is likely to only appear in the context of one degree profile).

Category	Number of	
	instances	edges
movie	116	N/A
actor	252	299
director	111	151
editor	86	112
producer	155	210
writer	145	185
country	14	122
production company	98	143
FX company	32	49
Total	1009	1271

Table 8. Distribution of node types and edges for the IMDb data set. Every non-movie node is linked to a movie and vice versa.

5.2. Experimental Methodology

The purpose of this case study is to get a better understanding of clusters that come out of ‘‘collapsing’’ the graph around a signature. For example, consider the IMDb data set described above.

Because of the regularity of the graph (non-movie nodes only connect to movie-nodes and vice versa), looking at degree profiles of size 3 is not going to be very informative as a given signature will only have one degree profile. Instead, we consider how instances of signatures cluster.

If we identify all the instances of the ‘‘actor movie writer’’ signature, then it is clear that many of these instances will overlap (share nodes). For example, consider a movie (in this data set) with two actors and one writer. We would identify that as two instances of the signature (one per actor). In fact, we can now consider the clusters that fall out of these overlapping instances. By clustering all overlapping instances, we analyze how these clusters differ, thereby gaining insight into underlying structure of the graph.

5.3. Analyzing Clusters

The first signature we analyze is the ‘‘actor movie writer’’ signature, meaning that there is an observed subgraph of size 3 which contains an actor node, a movie node, and a writer node. This signature occurred in 5.5% of all size 3 subgraphs.

As mentioned above, rather than treating each signature instance separately, this collapsed view results in a greater reduction of overall nodes and at the same time visually shows how much instances overlap. Further, merging instances in this manner produces a ‘‘community’’ view, where these nodes represent a community of entities that relate to each other in one way or another based on the signature being analyzed. The collapsed graph contains 45 clusters (collapsed nodes) and 529 original nodes that did not belong to the selected pattern, thereby reducing the graph by 43%.

Analyzing how these clusters differ or are similar reveals some interesting facts in this domain. The clusters all contained movies from only one country (there are 14 countries represented in this graph). This is not surprising as actors from one country generally do not appear in movies from other countries and this view highlights that behavior. Most of the smaller clusters (containing less than 10 nodes) were centered on one or two movies which were generally either from before 1970 or foreign. This finding again is not surprising given that IMDb obviously has better coverage of newer movies from the U.S. than any other time or country. Therefore, older movies or foreign movies have sparser data and hence smaller clusters.

We also investigated other signature clusters and found other noteworthy facts that are very informative about this domain. For example, the instances of the ‘‘actor country movie’’ signature clustered across countries and we ended up with two large clusters, one consisting of U.S. and Europe and another with China and Hong Kong (Singapore and Australia were represented as two small individual clus-

ters). The “actor movie producer” signature generated surprisingly many clusters (35), most of them being relatively small (< 10), suggesting that producers generally keep to themselves or only co-produce with a limited set of other producers. This behavior interestingly also appears to be the case with production companies (analyzing the “actor movie production-company” signature).

6. Discussion and Limitations

In this paper we introduced a new and novel framework for graph mining in the context of mining for interesting graph signatures—types of nodes that are connected in some way in an induced subgraph. We do this in two steps—first representing subgraphs in a lossy fashion using degree profiles, and then calculating distribution statistics on these degree profiles. We noted that the second step is agnostic of the graph representation and that using the more common canonical form would work just as well and would provide more detailed mining capabilities.

We evaluated four mining capabilities on a computer science department web-site. The mining techniques were based on analyzing distributions of graph pattern for specific signatures, where we used a degree profile representation for an observed instance of a graph signature. We showed how our statistics could be used to find rare and common signatures, signatures who were (dis)similar to the distribution of induced subgraphs of the same size as the signature, as well as signatures that always appear in certain in a certain graph pattern (degree profile in this study).

We also evaluated a different type of mining capability: clustering a graph based on a given signature. In this case, we looked for structural patterns that come to light when analyzing clusters that come out of “collapsing” a graph around overlapping instances of the graph signature. We showed how this capability unveiled some interesting and useful patterns in the movie domains such as how movies cluster around countries and how producers (and production companies) tend to form small cliques.

Although our new approach is powerful and facilitates new graph mining capabilities, its main drawback is that it is computationally expensive. Specifically, counting all instances of subgraphs of a given size is very expensive, especially for graphs with high-degree nodes. This is a known problem in graph mining and our work was not focused on addressing this issue. However, our approach does provide one potential way of making this tractable. Because we count distribution statistics over observed instance graph patterns, we believe that using standard statistical sampling methods can be used to calculate approximate statistics. Sampling in graphs is an open area, but is a direction we plan to pursue in future work.

Acknowledgments

This work was sponsored in part by the Office of Naval Research under award number N00014-06-M0109. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Office of Naval Research or the U.S. Government.

References

- [1] J. Coble, D. J. Cook, R. Rathi, and L. B. Holder. Iterative structure discovery in graph-based data. *International Journal of Artificial Intelligence Techniques*, 14(1–2):101–124, 2005.
- [2] D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.
- [3] M. Craven, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and C. Y. Quek. Learning to Extract Symbolic Knowledge from the World Wide Web. In *15th Conference of the American Association for Artificial Intelligence*, 1998.
- [4] S. Fortin. The graph isomorphism problem. Technical Report TR96-20, Department of Computing Science, University of Alberta, 1996.
- [5] L. Holder, D. Cook, and S. Djoko. Substructure discovery in the subdue system. In *Proceedings of the Workshop on Knowledge Discovery in Databases*, pages 169–180, 1994.
- [6] L. B. Holder and D. J. Cook. Graph-based relational learning: Current and future directions. *SIGKDD Explorations special issue on Multirelational Data Mining*, 2003.
- [7] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Principles of Data Mining and Knowledge Discovery*, pages 13–23, 2000.
- [8] A. Inokuchi, T. Washio, and H. Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, 50:321–354, 2003.
- [9] M. Kuramochi and G. Karypis. Frequent subgraph discovery. In *International Conference on Data Mining (ICDM)*, pages 313–320, 2001.
- [10] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. In *SIAM International Conference on Data Mining (SDM)*, 2004.
- [11] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovski, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298, October 2002.
- [12] L. D. Raedt and S. Kramer. The levelwise version space algorithm and its application to molecular fragment finding. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 853–859, 2001.
- [13] T. Washio and H. Motoda. State of the art of graph-based data mining. *SIGKDD Explorations Newsletter*, 5(1):59–68, 2003.
- [14] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Proceedings of the International Conference on Data Mining (ICDM)*, 2002.
- [15] X. Yan and J. Han. Closegraph: Mining closed frequent graph patterns. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2003.
- [16] K. Yoshida, H. Motoda, and N. Indurkha. Graph-based induction as a unified learning framework. *Journal of Applied Intelligence*, 4:297–328, 1994.