Frequent Subgraph Summarization with Error Control

Zheng Liu[1], Ruoming Jin[2], Hong Cheng[1], and Jeffrey Xu Yu[1]

[1] The Chinese University of Hong Kong
[2] Kent State University
{zliu,hcheng,yu}@se.cuhk.edu.hk,
jin@cs.kent.edu

Abstract. Frequent subgraph mining has been an important research problem in the literature. However, the huge number of discovered frequent subgraphs becomes the bottleneck for exploring and understanding the generated patterns. In this paper, we propose to summarize frequent subgraphs with an independence probabilistic model, with the goal to restore the frequent subgraphs and their frequencies accurately from a compact summarization model. To achieve a good summarization quality, our summarization framework allows users to specify an error tolerance σ, and our algorithms will discover k summarization templates in a top-down fashion and keep the frequency restoration error within σ. Experiments on real graph datasets show that our summarization framework can effectively control the frequency restoration error within 10% with a concise summarization model.

Keywords: frequent subgraph, pattern summarization.

1 Introduction

Frequent subgraph mining has been an important research problem in the literature, with many efficient algorithms proposed [10,12,21,24,1,8,17]. Given a collection $\mathcal{D}$ of graphs, frequent subgraph mining is to discover all subgraphs whose frequencies are no less than a user-specified threshold f_{min}. Frequent subgraphs are useful in many applications, for example, as active chemical structures in HIV-screening datasets, spatial motifs in protein structural families, discriminative features in chemical compound classification [4], and indexing features [26] in graph databases to support graph queries.

One major issue in frequent subgraph mining is the difficulty of exploring and analyzing numerous patterns generated due to the exponential number of combinations. Given a graph with n edges, the total number of possible subgraphs could be 2^n. Hundreds of thousands of frequent subgraphs may be generated under a moderate minimum frequency threshold. A partial solution to this problem is mining closed or maximal frequent subgraphs [25,9,19], which generates fewer subgraph patterns. But in many cases the maximal and closed graph patterns can still be quite numerous, so the difficulty of exploring a large number of patterns still exists.

J. Wang et al. (Eds.): WAIM 2013, LNCS 7923, pp. 1–12, 2013.

In the literature, there have been several methods which use probabilistic models to summarize frequent itemsets [23,22,11]. These probabilistic models, as a concise summarization, are effective to restore all the frequent itemsets and their frequencies. In this paper, we also use a probabilistic model to summarize frequent subgraphs by preserving their structure and frequency information as much as possible. Given a set of frequent subgraphs, we partition them into a few subsets. From each subset we choose a representative, which is called a template subgraph. All the frequent subgraphs in a subset are subgraphs of the corresponding template subgraph. The template subgraph is a summarization model which can restore the subgraphs in the subset and their frequencies. This problem is more challenging than itemset summarization, due to two difficulties in subgraph mining: (1) "multiple embeddings", i.e., a subgraph can have multiple embeddings in a summarization model which is a template subgraph; and (2) "topological constraint", i.e., the topological structure specifies the connectivity constraint among nodes and edges, while in itemset summarization, there is no such constraint between individual items. To solve the problem, we make an independence assumption between edges in a frequent subgraph. We take a regression approach to estimate the parameters in the independence probabilistic model by least square estimation. The probabilistic model allows users to restore frequent subgraphs and their frequencies from template subgraphs. To ensure a good summarization quality, we allow users to specify an error tolerance σ and our algorithms take a top-down approach to discover k template subgraphs. Multiple regression models will be built based on the k template subgraphs to control the frequency restoration error within σ. We have evaluated our subgraph summarization approach on real graph datasets. Experimental results show that our method can achieve a concise summarization with high accuracy in terms of subgraph frequency restoration error.

The rest of this paper is organized as follows. We formally define the problem of frequent subgraph summarization in Section 2 and propose the summarization algorithms in Section 3. We report the experimental results in Section 4 and discuss the related work in Section 5. Finally, Section 6 concludes the paper.

2 Problem Statement

A graph G is a triple (V, E, Γ), where V and E are the node set and the edge set, respectively. Γ is a finite set of labels, and each node $v \in V$ or edge $(u, v) \in E$ is mapped to a label in Γ, denoted as $\Gamma(v)$ or $\Gamma(u, v)$. A graph g is a subgraph of another graph G if there exists a subgraph isomorphism from g to G, denoted as $g \subset G$. G is called a supergraph of g.

Definition 1. ***Subgraph Isomorphism.*** *For two graphs g and G, G contains a subgraph that is isomorphic to g, if there is an injective function $h : V_g \rightarrow V_G$, such that $\forall v \in V_g, \Gamma_g(v) = \Gamma_G(h(v))$, and $\forall (u, v) \in E_g, (h(u), h(v)) \in E_G$ and $\Gamma_g(u, v) = \Gamma_G(h(u), h(v))$, where Γ_g and Γ_G are the label sets of g and G, respectively.*

Definition 2. ***Frequent Subgraph.*** *Given a collection $\mathcal{D}$ of graphs, a graph g is frequent if $f(g) \geq f_{min}$, where $f(g)$ is the number of graphs in $\mathcal{D}$ containing g, and f_{min} is a user-specified minimum frequency threshold.*

A frequent subgraph g is a maximal one if and only if there does not exist another frequent subgraph g' and $g \subset g'$. A frequent subgraph g is a closed one if and only if there does not exist another frequent subgraph g', $g \subset g'$ and $f(g') = f(g)$. Anti-monotonicity holds for frequent subgraphs in a graph database, which means the subgraphs of a frequent subgraph are also frequent.

Given a set of frequent subgraphs $\mathcal{F}$, we study how to summarize them into a small number of representatives, called template subgraphs, so that all frequent subgraphs and their frequencies can be restored accurately.

3 Summarization Algorithms

3.1 Subgraph Summarization by Regression

We propose to use a regression approach to frequent subgraph summarization. Based on the subgraph containment relationship, we can build a partial order graph (POG) consisting of all frequent subgraphs. Fig. 1 shows an example. Each node in the POG is a frequent subgraph. Subgraphs in the same level are of the same size, measured by the number of edges. In the POG, two subgraphs are connected by a directed edge from the larger one to the smaller one, if the smaller one is a subgraph of the larger and differs by one edge. Suppose g is a subgraph in the POG, we use connected children to represent its connected neighbors smaller than g, and g is called a connected parent. We use reachable children to represent all the nodes that can be reached by traveling along the edges in the POG, starting from g. A maximal subgraph does not have a connected parent in POG. If there is more than one maximal frequent subgraph, we add a union of all maximal frequent subgraphs as the root of the POG.

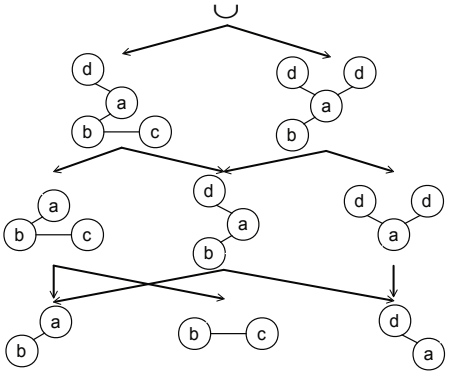


Fig. 1. Partial order graph of frequent subgraphs based on containment relationship

Suppose g and g' are two connected subgraphs in the POG, where g and g' differ by only one edge e, that is, $g \cup \{e\} = g'$. Let $p(g'|g)$ denote the conditional probability that a graph G from $\mathcal{D}$ containing g also contains g'. Since $g \cup \{e\} = g'$, we denote $p(g'|g)$ as $p(e|g)$, the conditional probability that a graph G from a graph database $\mathcal{D}$ containing g also contains edge e. Let $f(g)$ denote the frequency of a frequent subgraph g, then

$$p(g'|g) = p(e|g) = \frac{f(g')}{f(g)}. \tag{1}$$

Given any two frequent subgraphs g_1 and g_l that differ by $l-1$ edges $\{e_1, e_2, \cdots, e_{l-1}\}$, then

$$f(g_l) = f(g_1) \times p(e_1, e_2, \cdots, e_{l-1}|g_1). \tag{2}$$

Let g_i denote the graph $g_1 \cup \{e_1, \cdots, e_{i-1}\}$. Following the chain rule of conditional probabilities, we have

$$f(g_l) = f(g_1) \times \prod_{i=1}^{l-1} p(e_i|g_i). \tag{3}$$

To simplify the joint probability estimation, we apply the following independence assumption: whether a frequent subgraph g contains an edge e is independent of the structure of $g \setminus \{e\}$. Without loss of generality, we use $p(e)$ to denote $p(e|*)$, where $*$ denotes an arbitrary subgraph g that $g \cup \{e\}$ is frequent. Under this assumption, we can rewrite Eq. (3) for subgraph g_l and g_1 as follows:

$$f(g_l) = f(g_1) \times \prod_{i=1}^{l-1} p(e_i). \tag{4}$$

Given a frequent subgraph g, let $\mathcal{G} = \{g_1, g_2, \cdots, g_n\}$ be the frequent subgraphs reachable from g in the POG. Suppose we know the frequency $f(g)$ of g, as well as all the probabilities $p(e_j)$ of edges in g, with the independent assumption, we can estimate the frequency of each graph $g_i \in \mathcal{G}$ according to Eq. (4). By applying the logarithmic transformation on both sides of the equation, we have

$$\log f(g) = \log f(g_i) + \sum_{j=1}^{i-1} \log p(e_j). \tag{5}$$

Similar to the regression approach in [11], we can build a regression model $Y = X\beta + E$ for $\mathcal{G}$, where E is the matrix of error terms,

$$Y = \begin{bmatrix} \log f(g) - \log f(g_1) \\ \cdots \\ \log f(g) - \log f(g_n) \end{bmatrix}, X = \begin{bmatrix} \mathbf{1}_{e_1 \in g_1} & \cdots & \mathbf{1}_{e_l \in g_1} \\ \cdots & \cdots & \cdots \\ \mathbf{1}_{e_1 \in g_n} & \cdots & \mathbf{1}_{e_l \in g_n} \end{bmatrix}, \text{ and } \beta = \begin{bmatrix} \log p(e_1) \\ \cdots \\ \log p(e_l) \end{bmatrix}. \tag{6}$$

Here, $\mathbf{1}_{e_i \in g_j}$ is an indicator that edge e_i belongs to graph g_j. $\mathbf{1}_{e_i \in g_j} = 1$ if $e_i \in g_j$, and $\mathbf{1}_{e_i \in g_j} = 0$, otherwise. The least square estimation [18] of the above regression model is to minimize the sum of squares of the errors (residues), which is

$$\delta = \min_{\beta} \Big\{ (Y - X\beta)'(Y - X\beta) \Big\}. \tag{7}$$

Then the solution is

$$\begin{aligned} \hat{\beta} &= \arg\min_{\beta} \Big\{ (Y - X\beta)'(Y - X\beta) \Big\} \\ &= (X'X)^{-1}X'Y. \end{aligned} \tag{8}$$

By applying the above regression approach, we are able to summarize any frequent subgraph g in the POG, together with all its reachable children, as a regression model. We call g a template subgraph. The template subgraphs can restore all frequent subgraphs and their frequencies. We define the relative restoration error as follows.

Definition 3. ***Average Relative Restoration Error.*** *Let $\mathcal{F}$ denote the set of frequent subgraphs. For each subgraph $g \in \mathcal{F}$, $f(g)$ and $r(g)$ are the true frequency and the restored frequency of g, respectively. The relative frequency restoration error of g is $|r(g) - f(g)|/f(g)$, denoted as $\delta(g)$. The average relative restoration error of the frequent subgraph set $\mathcal{F}$ is*

$$\delta_{avg}(\mathcal{F}) = \frac{1}{|\mathcal{F}|} \sum_{g \in \mathcal{F}} \frac{|r(g) - f(g)|}{f(g)}. \tag{9}$$

Subgraph Summarization with Error Tolerance σ. Given a set of frequent subgraphs $\mathcal{F}$, and a maximum error tolerance σ, the problem of frequent subgraph summarization is to partition $\mathcal{F}$ into as few groups as possible, and each group $\mathcal{G}$ satisfies the following: (1) $\mathcal{G}$ can be summarized as a single regression model, and (2) $\delta_{avg}(\mathcal{G}) \leq \sigma$, where δ_{avg} is defined in Definition 3.

3.2 Summarization Framework

Our summarization framework is presented in Algorithm 1, which is in a top-down fashion. The algorithm starts from a single template subgraph, the root of the POG, which is a union template subgraph for all maximal frequent subgraphs. Let g be a template subgraph, we use $\delta_{avg}(g)$ to denote the average restoration error of the regression model built on g with its reachable children. In each repeated loop from line 3 to line 8, the algorithm first divides the template subgraph with the maximum average restoration error into two template subgraphs, if the error is larger than the threshold σ, and then tries to merge the newly generated template subgraphs with other template subgraphs, until all the template subgraphs have an average restoration error $\leq \sigma$.

We use a binary tree T to maintain the generated template subgraphs. At line 1 in Algorithm 1, T is the root of POG $\mathcal{G}$. At line 6, we use the symbol of set

ALGORITHM 1: Summarization Framework

Input: POG $\mathcal{G}$, Error tolerance σ
Output: Template Subgraphs with Regression Models

$T \leftarrow$ the root of POG $\mathcal{G}$;
while *true* **do**
  $g' \leftarrow \arg\max_g \{\delta_{avg}(g) | g \in T\}$;
  if $\delta_{avg}(g') > \sigma$ **then**
    $\{g_1, g_2\} =$ **divide**(g');
    $T \leftarrow T \cup \{g_1, g_2\}$;
    merge(T, g', g_1, g_2);
  end
  else break;
end
return *Template Subgraphs with Regression Models in* T.

ALGORITHM 2: **divide**(Template g)

C $\leftarrow$ the directed children of g in POG $\mathcal{G}$;
Cand $\leftarrow \emptyset$;
foreach $c_i \in$ **C do**
  let G_i be the POG subgraph of g rooted at c_i;
  let G_g be $g \setminus G_i$ /*g corresponds to G_g*/;
  Build regression models R_i and R_g for G_i and G_g (or equivalently c_i and g);
  $\epsilon_i \leftarrow$ total residue of R_i;
  $\epsilon_g \leftarrow$ total residue of R_g;
  Cand $\leftarrow$ **Cand** $\cup \{(c_i, g, \epsilon_i + \epsilon_g)\}$;
end
$(c_{min}, g_{min}) \leftarrow \arg\min_{c_i} \{\epsilon \mid (c_i, g, \epsilon) \in$ **Cand**$\}$;
Update the regression models for c_i and g in $\mathcal{G}$ based on c_{min} and g_{min};
return $\{c_{min}, g_{min}\}$

union to denote the update of binary tree T. g_1 is a child of g' in POG, and g_2 is g' with a different regression model. The update is done by adding the new template subgraph g_1 to T as a child of g' and update the regression model of g' by the one of g_2.

3.3 Template Subgraph Division

Algorithm 2 presents the procedure to divide a template subgraph. Consider a template graph g to be divided, the potential new template subgraphs are the connected children of g. Take Fig. 2 as an example. c_1, c_2 and c_3 are g's connected children in the POG. Suppose c_i is selected ($1 \leq i \leq 3$). Then we have two template subgraphs: c_i and g. There exist frequent subgraphs that are the descendants of both c_i and g. We restrict them to belong to only one

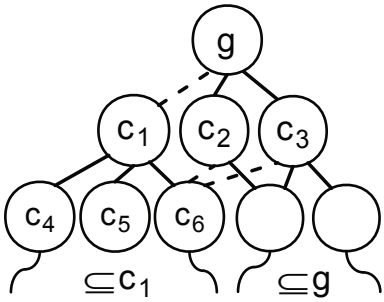


Fig. 2. Template Subgraph Division

template subgraph, either c_i or g, in order to obtain better regression models. The rule is to let the sharing part belong to the smaller template subgraph c_i.

We discuss how to build regression models for this case. Let e be the edge that appears in g but does not appear in c_i. All the descendants of c_i in the POG do not contain e. In other words, the descendants of g are divided into two parts: frequent subgraphs containing e and frequent subgraphs not containing e. By selecting c_i, the POG rooted at g is divided into two subgraphs. One POG subgraph G_i is rooted at c_i and contains all descendants of c_i. The other POG subgraph G_g contains g and all its descendants excluding those in G_i. As indicated in Fig. 2, the dotted lines indicate some descendant of G_g may be a supergraph of some graph in G_i because of the existence of the edge e, and must be deleted. We build regression models for G_i and G_g, respectively. Among all the possible children of g, we select the child node c_i of g in the POG which results in the minimum sum of residues of the regression models for both G_i and G_g. As the division continues, the residues and the average relative restoration errors will decrease. Eventually, the algorithm will stop when the average restoration error $\leq \sigma$.

3.4 Template Subgraph Merging

After the update of the binary tree for template subgraphs, a merging step is conducted at line 7 in Algorithm 1. The merging step serves as a refinement step of the binary tree with the hope to reduce the total number of template subgraphs by merging the updated template subgraphs with its parent subgraph or sibling subgraphs. Algorithm 3 presents the procedures of the merging step.

For a divided template subgraph g, g_1 is a child of g in the POG. Due to the division, all subgraphs reachable from g are divided into two sets, where g_1 and g_2 are the template subgraphs, respectively. Due to the change of the regression models, it is possible that g_2 could be merged with its parent g_p in the binary tree T and the average restoration error is below σ. Apparently, either the average restoration error of g_2 or that of g_p should be smaller than σ. For example, in Fig. 2, suppose c_1 is the divided template subgraph and c_4 is the child with the

ALGORITHM 3: merge(BinaryTree T, Template g, g_1, g_2)

```
if T contains only two template subgraphs then
    return;
end
Let g_p be the parent of template subgraph g in T;
if lowerbound(g_p, g_2) ≤ σ then
    δ ← average restoration error of g_p after merging g_2;
    if δ ≤ σ then
        Update g_p by merging g_p and g_2;
        Remove g_2 from T;
    end
end
```

minimum total residue. Then it is possible that merging c_1 and g will generate a regression model whose average restoration error is below σ.

3.5 Queriable Summarization

Given our frequent subgraph summarization with restoration error control, we can provide an answer when a user wants to know the frequency of a frequent subgraph. Since every frequent subgraph in the POG belongs to one and only one template subgraph, we only need to know which template subgraph it belongs to. Based on Eqs. (6) and (8), we can estimate the restored frequency $r(g)$ of a frequent subgraph g according to the following equation,

$$r(g) = x\hat{\beta}, \tag{10}$$

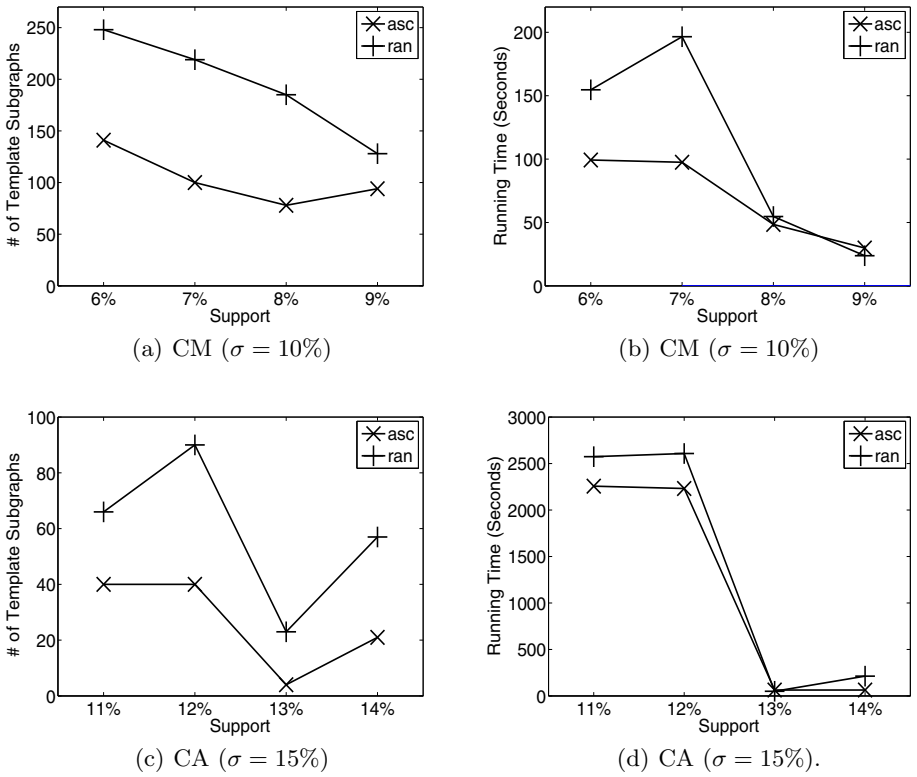
where $x = [\mathbf{1}_{e_1 \in g} \cdots \mathbf{1}_{e_l \in g}]$. Recall that $\mathbf{1}_{e_i \in g}$ is an indicator that edge e_i in the template graph belongs to graph g. $\mathbf{1}_{e_i \in g} = 1$ if $e_i \in g$, and $\mathbf{1}_{e_i \in g} = 0$ if $e_i \notin g$. In our summarization framework, there is a partial order among edges to avoid the problems caused by multiple embeddings in a template graph. Upon all the template subgraphs obtained, we can identify which template subgraph a query graph g belongs to by utilizing the global edge ID's.

4 Experimental Results

In this section, we evaluate the performance of our proposed summarization method. All the experiments were run on a server with 4 CPU and 24GB memory running GNU/Linux.

We use the AIDS antiviral screen compound dataset[1]. There are totally 43850 chemical compounds, which are classified into three categories: Confirmed Active (CA); Confirmed Moderately active (CM); and Confirmed Inactive (CI). As chemical compounds belonging to CI are not useful for the drug discovery, we

[1] `http://dtp.nci.nih.gov/docs/aids/aids_data.html`



(a) CM ($\sigma = 10\%$) (b) CM ($\sigma = 10\%$)

(c) CA ($\sigma = 15\%$) (d) CA ($\sigma = 15\%$).

Fig. 3. Summarization Results on CM and CA

use datasets CA and CM in our experiments. CA contains 463 compounds and CM contains 1093 compounds. We list the number of frequent subgraphs mined from CA and CM under different minimum support in Table 1.

Experiment Setting: We implemented two variants of our summarization framework in Algorithm 1, namely, *asc* and *ran*. As discussed, in a union template subgraph, we could arrange the order of maximal frequent subgraphs in different ways to avoid the issues caused by multiple embeddings and common reachable children. *ran* denotes that we arrange the order randomly in division, while *asc* denotes that maximal frequent subgraphs are always sorted in the

Table 1. Number of Frequent Subgraphs

	CM				**CA**			
Minimum support	6%	7%	8%	9%	11%	12%	13%	14%
# of frequent subgraphs	5997	4265	3415	2627	15231	14318	8094	7612

ascending order of their sizes, measured by number of edges, during the union template creation.

Fig. 3 reports the summarization results on datasets CM and CA. Fig. 3(a) and 3(b) report the quality and running time on CM under different values of minimum support when the restoration error tolerance is 10%. Generally, a smaller value of support means a large number of frequent subgraphs. Under the same error tolerance, *asc* can generate a more compact summarization, i.e., a smaller number of template subgraphs, than *ran*. It is also more efficient than *ran*. Similar performance can be observed in Fig. 3(c) and 3(d) on CA. *acs* always generates fewer template subgraphs than *ran*. Compared with the number of frequent subgraphs in CM and CA shown in Table 1, we can find that the number of generated template subgraphs is up to several hundred times smaller than that of the frequent subgraphs in both datasets under a smaller error tolerance.

5 Related Works

Frequent Subgraph Summarization: One potential issue in frequent subgraph mining is the huge number of the discovered frequent subgraphs. Closed frequent subgraph [25] and maximal frequent subgraph [9,19] can partially solve this issue, but in many cases the number of closed or maximal subgraphs is still very huge. Recently researchers [3,15,28,6] have focused on selecting a small number of representative graph patterns to represent many similar subgraphs. The similarity could be maximum common subgraph [6], graph edit distance [3,28], or Jaccard distance [15]. Sampling is another approach to solve the overloading issue. Hasan et al. [7] proposed a sampling strategy based random walks on the frequent subgraph lattice.

Frequent Subgraph Mining: Many algorithms have been proposed for finding frequent subgraphs in graph databases, where the frequency of a subgraph is the total number of graphs containing the subgraph in the database. Similar to the Apriori-based approaches in frequent itemset mining, Apriori-based algorithms for frequent subgraph mining are proposed in [10,12,21], where the search strategy follows a breadth-first manner. Subgraphs of small sizes are searched first. Once identified, new candidate subgraphs are generated by joining two highly overlapping frequent subgraphs. In each iteration, the size of these candidate subgraphs is increased by one. Other algorithms [24,1,8,17] employ a pattern-growth style. New candidate subgraphs are generated by adding a new edge to the current. It is possible that a candidate subgraph can be extended from multiple frequent subgraphs. In gSpan [24], each candidate subgraph is associated with a depth-first code for duplicate identification. There are also research efforts on finding frequent subgraphs in a single large graph [2,14,5,13], where an important problem is how to calculate the frequency. One solution [13] is to consider the maximum number of non-overlapping embeddings as the frequency.

Other Large Graph Summarization: There are also works for large graph summarization. Navlakha et al. [16] proposed an approach to generate a compact

graph representation which can be restored to the original graph with bounded error. They used a single super-node to represent a clique or near-clique, with an additional table recording the missing edges. The quality of the summary is measured by the minimum description length. Tian et al. [20] proposed two aggregating algorithms for graph summarization. The top-down approach first groups together all the nodes with the same category attributes. Then the groups of nodes are repeatedly split until there are k groups. The bottom-up approach first groups together all the nodes with both the same node attributes and the same edge attributes. Then small groups of nodes are merged into larger ones till there are k groups left. Zhang et al. [27] extended Tian's approach to deal with numerical attributes by automatically categorizing numerical attribute values. They also proposed an interestingness measure to identify the most interesting resolutions.

6 Conclusions

In this paper, we have proposed a frequent subgraph summarization framework with an independence probabilistic model. A regression approach is applied to estimate the parameters in the summarization model. Our summarization framework takes a top-down approach to recursively partition a summarization graph template into two, until the user-specified error tolerance is satisfied. Experimental results on real datasets show that our summarization model can effectively control the frequency restoration error within 10% with a concise representation.

In the future, we plan to study how to integrate our summarization framework into the pattern mining process to save the computation cost of finding all frequent subgraphs. We will also study the frequent subgraph summarization problem in a single large graph. Depending on the definition of frequency, the anti-monotonicity property does not always hold, which will introduce new challenges in the summarization framework to avoid false-positive subgraphs.

Acknowledgments. This work is supported by the Hong Kong Research Grants Council (RGC) General Research Fund (GRF) Project No. CUHK 418512 and 411211.

References

1. Borgelt, C., Berthold, M.R.: Mining molecular fragments: Finding relevant substructures of molecules. In: ICDM, p. 51 (2002)
2. Bringmann, B., Nijssen, S.: What is frequent in a single graph? In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds.) PAKDD 2008. LNCS (LNAI), vol. 5012, pp. 858–863. Springer, Heidelberg (2008)
3. Chen, C., Lin, C.X., Yan, X., Han, J.: On effective presentation of graph patterns: a structural representative approach. In: CIKM, pp. 299–308 (2008)
4. Deshpande, M., Kuramochi, M., Wale, N., Karypis, G.: Frequent substructure-based approaches for classifying chemical compounds. IEEE Trans. on Knowl. and Data Eng. 17(8), 1036–1050 (2005)

5. Fiedler, M., Borgelt, C.: Subgraph support in a single large graph. In: ICDM Workshops, pp. 399–404 (2007)
6. Hasan, M.A., Chaoji, V., Salem, S., Besson, J., Zaki, M.J.: Origami: Mining representative orthogonal graph patterns. In: ICDM, pp. 153–162 (2007)
7. Hasan, M.A., Zaki, M.J.: Output space sampling for graph patterns. Proc. VLDB Endow. 2(1), 730–741 (2009)
8. Huan, J., Wang, W., Prins, J.: Efficient mining of frequent subgraphs in the presence of isomorphism. In: ICDM, p. 549 (2003)
9. Huan, J., Wang, W., Prins, J., Yang, J.: Spin: mining maximal frequent subgraphs from graph databases. In: KDD, pp. 581–586 (2004)
10. Inokuchi, A., Washio, T., Motoda, H.: An apriori-based algorithm for mining frequent substructures from graph data. In: Zighed, D.A., Komorowski, J., Żytkow, J.M. (eds.) PKDD 2000. LNCS (LNAI), vol. 1910, pp. 13–23. Springer, Heidelberg (2000)
11. Jin, R., Abu-Ata, M., Xiang, Y., Ruan, N.: Effective and efficient itemset pattern summarization: Regression-based approaches. In: KDD, pp. 399–407 (2008)
12. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: ICDM, pp. 313–320 (2001)
13. Kuramochi, M., Karypis, G.: Finding frequent patterns in a large sparse graph. Data Min. Knowl. Discov. 11(3), 243–271 (2005)
14. Li, S., Zhang, S., Yang, J.: DESSIN: Mining dense subgraph patterns in a single graph. In: Gertz, M., Ludäscher, B. (eds.) SSDBM 2010. LNCS, vol. 6187, pp. 178–195. Springer, Heidelberg (2010)
15. Liu, Y., Li, J., Gao, H.: Summarizing graph patterns. In: ICDE, pp. 903–912 (2008)
16. Navlakha, S., Rastogi, R., Shrivastava, N.: Graph summarization with bounded error. In: SIGMOD, pp. 419–432 (2008)
17. Nijssen, S., Kok, J.N.: A quickstart in frequent structure mining can make a difference. In: KDD, pp. 647–652 (2004)
18. Freund, R.J., Wilson, W.J., Sa, P.: Regression Analysis: Statistical Modeling of a Response Variable, 2nd edn. Academic Press (2006)
19. Thomas, L.T., Valluri, S.R., Karlapalem, K.: Margin: Maximal frequent subgraph mining. In: ICDM, pp. 1097–1101 (2006)
20. Tian, Y., Hankins, R.A., Patel, J.M.: Efficient aggregation for graph summarization. In: SIGMOD, pp. 567–580 (2008)
21. Vanetik, N., Gudes, E., Shimony, S.E.: Computing frequent graph patterns from semistructured data. In: ICDM, p. 458 (2002)
22. Wang, C., Parthasarathy, S.: Summarizing itemset patterns using probabilistic models. In: KDD, pp. 730–735 (2006)
23. Yan, X., Cheng, H., Han, J., Xin, D.: Summarizing itemset patterns: a profile-based approach. In: KDD, pp. 314–323 (2005)
24. Yan, X., Han, J.: gspan: Graph-based substructure pattern mining. In: ICDM, p. 721 (2002)
25. Yan, X., Han, J.: Closegraph: mining closed frequent graph patterns. In: KDD, pp. 286–295 (2003)
26. Yan, X., Yu, P.S., Han, J.: Graph indexing based on discriminative frequent structure analysis. ACM Trans. Database Syst. 30(4), 960–993 (2005)
27. Zhang, N., Tian, Y., Patel, J.: Discovery-driven graph summarization. In: ICDE, pp. 880–891 (2010)
28. Zhang, S., Yang, J., Li, S.: Ring: An integrated method for frequent representative subgraph mining. In: ICDM, pp. 1082–1087 (2009)