Frequent Generalized Graph Mining via Graph Edit Distances

Richard Palme, Pascal Welke

Abstract

Computing the subgraphs which occur frequently in a database of labeled graphs is a fundamental problem in data mining. Graph data is often equipped with semantic information in form of an ontology, for example when dealing with linked data or knowledge graphs. Previous work suggests to exploit this semantic information in order to compute frequent generalized patterns, i.e. patterns for which the total frequency of all more specific patterns exceeds the frequency threshold. However, the problem of computing the frequency of a generalized pattern has not yet been addressed. In this work, we propose a method for computing the frequency of generalized patterns which is based on the graph edit distance.

1. Introduction

Nowadays, an ever-increasing amount of graph data is collected, often in form of linked data or knowledge graphs. Linked data, and especially knowledge graphs, often come with an ontology, which provides background knowledge about the entities and entity relations that appear in the dataset. Naturally, the question arises if it is possible to exploit the semantic information given by an ontology, in order to improve the performance of data mining methods on graph data.

A common graph data mining task is to generate the set of frequent subgraphs of a graph database. The frequent subgraph mining problem (FSM) has many applications, ranging from database compression [1] to machine learning [2]. In order to improve the results of FSM, the semantic information provided by a label hierarchy or taxonomy can and sometimes must be used as background knowledge.

As an example, suppose the graphs in a database contain vertex labels such as “donkey”, “rabbit”, “carrot” or “cabbage”, and suppose these four vertex labels do not appear frequently in the database. If there is a label hierarchy which tells us that “donkey” and “rabbit” are herbivores, while “carrot” and “cabbage” are vegetables, then we can exploit this semantic information in order to find frequently occuring patterns in the database, such as “herbivore eats vegetable”. These patterns are called generalized patterns. The problem of frequent generalized subgraph mining has a long history [3, 4] and recently gained more traction, again [5, 6, 7].

Definition 1 (Frequent Generalized Subgraph Mining (FGSM))**.** *We say that there is a generalized subgraph isomorphism (GSGI) between two graphs H and G if G contains a subgraph H' (up to isomorphism) s.t. H' can be constructed from H by replacing any label of H by a more specific*

Under review at SeDaMi'22 Workshop at ECML/PKDD
palme@cs.uni-bonn.de (R. Palme); welke@cs.uni-bonn.de (P. Welke)
0000-0002-2123-3781 (P. Welke)

label w.r.t. the label hierarchy. Here, the root in the label hierarchy is the most general label. A graph H is a frequent generalized subgraph w.r.t. a graph database D if there are at least t graphs $G_1, \ldots, G_t$ in D s.t. there is a GSGI between H and G_i for any $i = 1, \ldots t$. The FGSM problem is to compute the set of all frequent generalized subgraphs of D.

In order to determine the frequency of a generalized pattern, we need an algorithm for solving the generalized subgraph isomorphism problem (GSGI). Unfortunately, to our knowledge, no previous work gives an algorithm for the GSGI problem. A naive solution to GSGI solves a subgraph isomorphism (SGI) problem with input H' and G for every specialization H' of H. Since the number of specializations of H is exponential in the size of H, this naive solution is not a feasible method for solving GSGI.

In this work we reduce GSGI to the graph edit distance problem (GED), thereby solving GSGI by a single computation of a specific GED between two graphs. Subsequently, we use a heuristic solver for GED within a frequent subgraph mining framework to enumerate frequent generalized subgraphs of arbitrary labeled graph databases.

2. Reduction of Generalized Subgraph Isomorphism (GSGI) to Graph Edit Distance (GED)

The graph edit distance (GED) is a measure for the dissimilarity between two labeled graphs [8]. Two graphs H and G are interpreted to be dissimilar w.r.t. GED if, for any sequence of edit operations that transforms H into G, the cost incurred by the sequence of edit operations is high. We remark that, like SGI and GSGI, GED is NP-hard. However, there are efficient heuristics to approximate GED [9].

Definition 2 (Graph Edit Distance)**.** *Let H and G be labeled graphs, let Σ be a finite label alphabet, and let ε be a symbol which is not an element of Σ. Denoting $\Sigma \cup \{\varepsilon\}$ by Σ_ε, we call a function*

$$c\colon \Sigma_\varepsilon \times \Sigma_\varepsilon \to [0, \infty)$$

an edit cost function if

$$\forall \alpha \in \Sigma_\varepsilon\colon c(\alpha, \alpha) = 0.$$

An edit cost function assigns an edit cost to each edit operation. Table 1 contains a comprehensive list of all considered edit operations and their associated edit costs. An edit path π between H and G is a finite sequence of edit operations $(o_i)_{i=1}^k$ that transforms H into a graph $\pi(H)$ that is isomorphic to G. The cost incurred by π is defined as

$$c(\pi) \coloneqq \sum_{i=1}^{k} c(o_i),$$

where $c(o_i)$ denotes the edit cost of the edit operation o_i. We denote the set of edit paths between H and G by $\Pi(H, G)$, and define the graph edit distance between H and G as follows:

$$\mathit{GED}(H, G) \coloneqq \min_{\pi \in \Pi(H,G)} c(\pi).$$

Edit operation	Edit cost
Insert an isolated vertex with label $\alpha \in \Sigma$	$c(\varepsilon, \alpha)$
Delete an isolated vertex u	$c(\lambda(u), \varepsilon)$
Substitute the label of a vertex u by $\alpha \in \Sigma$	$c(\lambda(u), \alpha)$
Insert an edge with label $\alpha \in \Sigma$	$c(\varepsilon, \alpha)$
Delete an edge e	$c(\lambda(e), \varepsilon)$
Substitute the label of an edge e by $\alpha \in \Sigma$	$c(\lambda(e), \alpha)$

Table 1
Edit operations and their associated edit costs. Deleting an edge $\{u, v\}$ does not delete u or v, and inserting an edge $\{u, v\}$ is only possible if u and v have been previously inserted or are vertices of H.

The GED can be used to solve the subgraph isomorphism problem (SGI) by imposing the following three constraints on the edit cost function:

$$\begin{aligned} \forall \beta \in \Sigma_\varepsilon \colon c(\varepsilon, \beta) = 0 && \text{(free insertions)} \\ \forall \alpha \in \Sigma \colon c(\alpha, \varepsilon) > 0 && \text{(paid deletions)} \\ \forall \alpha, \beta \in \Sigma \colon c(\alpha, \beta) > 0 \iff \alpha \neq \beta && \text{(paid substitutions)} \end{aligned}$$

We call the graph edit distance between two graphs the subgraph edit distance (SGED), if the edit cost function obeys these three constraints. For any two graphs H and G, we get

$$H \preceq G \iff \mathrm{SGED}(H, G) = 0,$$

where $H \preceq G$ is a shorthand for H being subgraph isomorphic to G. Thus, the SGI problem can be solved using the SGED problem.

Assuming the edit cost function obeys the triangle inequality in addition to the three constraints above, we get

$$H' \preceq H \implies \mathrm{SGED}(H', G) \leq \mathrm{SGED}(H', H) + \mathrm{SGED}(H, G) = \mathrm{SGED}(H, G).$$

In other words, SGED is monotone in its first argument. Many algorithms for frequent subgraph mining rely on the monotonicity of SGI in its first argument, and SGED also being monotone in its first argument ensures that SGED can be used as a drop-in replacement for SGI in many pattern mining algorithms.

To solve the generalized SGI problem, we impose the following four constraints on the edit cost function:

$$\begin{aligned} \forall \beta \in \Sigma_\varepsilon \colon c(\varepsilon, \beta) = 0 && \text{(free insertions)} \\ \forall \alpha \in \Sigma \colon c(\alpha, \varepsilon) > 0 && \text{(paid deletions)} \\ \forall \alpha, \beta \in \Sigma \colon c(\alpha, \beta) > 0 \iff \alpha \neq \beta \text{ and } \alpha \text{ is not more general than } \beta && \text{(paid substitutions)} \\ \forall \alpha, \beta \in \Sigma \colon c(\alpha, \beta) = 0 \iff \alpha = \beta \text{ or } \alpha \text{ is more general than } \beta && \text{(free specializations)} \end{aligned}$$

With these constraints, we get

$$\mathrm{GSGI}(H, G) = \text{true} \iff \mathrm{GED}(H, G) = 0.$$

Thus, the GSGI problem can be solved using the GED problem with an edit cost function that satisfies the four constraints given above. We can then use this solution to the GSGI problem in order to solve the frequent generalized subgraph mining problem (FGSM). Alternatively, we can impose the following four constraints on the edit cost function c:

$$\begin{aligned}
\forall \beta \in \Sigma_\varepsilon \colon c(\varepsilon, \beta) &< \infty \\
\forall \alpha \in \Sigma \colon c(\alpha, \varepsilon) &= \infty \\
\forall \alpha, \beta \in \Sigma \colon c(\alpha, \beta) = \infty &\iff \alpha \neq \beta \text{ and } \alpha \text{ is not more general than } \beta \\
\forall \alpha, \beta \in \Sigma \colon c(\alpha, \beta) < \infty &\iff \alpha = \beta \text{ or } \alpha \text{ is more general than } \beta
\end{aligned}$$

Then we get

$$\mathrm{GSGI}(H, G) = \text{true} \iff \mathrm{GED}(H, G) < \infty.$$

These constraints leave us the freedom to choose insertion costs and specialization costs as we wish. We can use this freedom in order to infuse additional background knowledge into the GED computation.

As an example, suppose the label hierarchy has been computed by applying hierarchical clustering to the set of labels. Then each leaf node in the cluster hierarchy corresponds to a label, and each non-leaf node corresponds to a generalized label which does not appear in the database. Since the cluster hierarchy is a dendrogram, for any generalized label α, we know the distance $d(\alpha, \beta)$ between α and any label β which is more specific than α.

We can infuse these distances into the GED computation as follows: We set the cost $c(\alpha, \beta)$ of substituting a generalized label α by a more specific label β to $d(\alpha, \beta)$, while the remaining edit costs are chosen s.t. the four constraints above are satisfied. Using these edit costs, the collection of graph edit distances between a generalized pattern H and all graphs G in the database yields an interestingness measure for H. Large values for $\mathrm{GED}(H, G)$ indicate that H is a rather specific pattern, while smaller values indicate that H is a rather general pattern. Since generalized patterns are arguably interesting if they are both frequent and specific, infusing label distances into the GED computation yields a method for ranking the frequent generalized subgraphs. We note that the mere *frequency* of generalized patterns can not be used for ranking them, since maximally general patterns have the highest frequency while not being interesting.

In most cases, label hierarchies do not specify label distances. However, for any generalized label α, we can always set the distance $d(\alpha, \beta)$ between α and any more specific label β to the length of the unique path between α and β in the hierarchy tree.

We implemented our method for solving FGSM by making use of the C++ library GEDLIB [10] to compute graph edit distances. We tested our method on the MUTAG [11] and PTC-MR datasets [12], which contain graphs representations of chemical compounds, and which are provided in a uniform file format in [13]. To compute a label hierarchy on chemical elements, we make use of the inter-cluster distance between clusters of chemical elements given in [14]. This inter-cluster distance has the property that many of the resulting clusters correspond to common groups of elements. Figure 1 shows a small part of the dendrogram which is the result of hierarchical clustering when using this distance.

Our experiments confirm that FGSM can uncover frequent patterns that are not found by frequent subgraph mining. An example of our findings is illustrated in Figure 2.

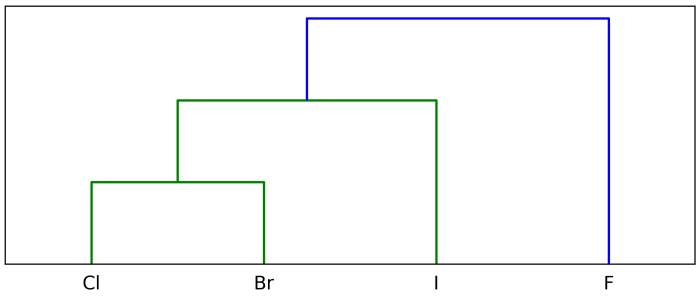

Figure 1: A section of the dendrogram created by clustering chemical elements. The chosen section of the dendrogram shows a cluster which only contains halogens.

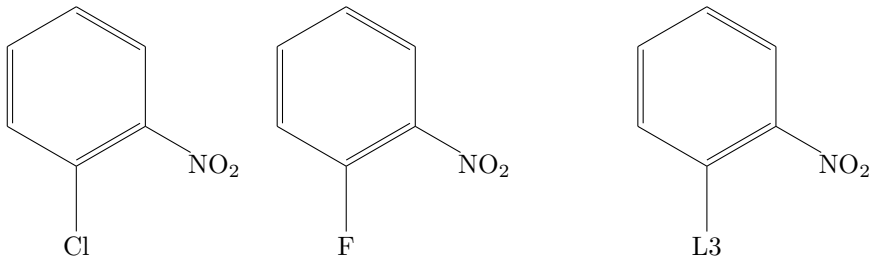

Figure 2: The two molecules on the left appear as subgraphs in the MUTAG database, while the graph on the right is a generalized pattern. Both molecules are infrequent for a relative frequency threshold of 5%, while the generalized pattern is frequent w.r.t. the same frequency threshold.

3. Conclusion

Frequent generalized subgraph mining is a variant of graph mining which exploits the semantic information provided by a label hierarchy. In this work, we propose a method for solving FGSM by using graph edit distance computations. Our method imposes constraints on the edit cost function in order to encode the background knowledge given by the label hierarchy. Since these constraints do not fully determine the edit cost function, we are free to choose the values for a subset of the edit costs. We have seen that this freedom of choice can be exploited to achieve additional goals. For example, we were able to assign an interestingness measure to each frequent generalized subgraph by choosing the values for selected substitution costs.

As an outlook, we note that edit cost functions are not restricted to model label hierarchies, and thus graph edit distances are a powerful tool for including domain knowledge beyond label hierarchies into graph mining procedures. While we don't include extensive experiments in this short article, we will make the source code of our mining algorithm publicly available.

References

[1] L. B. Holder, D. J. Cook, S. Djoko, Substucture discovery in the SUBDUE system, in: AAAI Workshop on Knowledge Discovery in Databases, AAAI Press, 1994, pp. 169–180.

[2] M. Deshpande, M. Kuramochi, N. Wale, G. Karypis, Frequent substructure-based approaches for classifying chemical compounds, IEEE Trans. Knowl. Data Eng. 17 (2005) 1036–1050. doi:10.1109/TKDE.2005.127.

[3] A. Inokuchi, Mining generalized substructures from a set of labeled graphs, in: IEEE International Conference on Data Mining, IEEE Computer Society, 2004, pp. 415–418. doi:10.1109/ICDM.2004.10041.

[4] A. Cakmak, G. Özsoyoglu, Taxonomy-superimposed graph mining, in: International Conference on Extending Database Technology, volume 261 of *ACM International Conference Proceeding Series*, ACM, 2008, pp. 217–228. doi:10.1145/1353343.1353372.

[5] A. Faci, M. Lesot, C. Laudy, cgspan: Pattern mining in conceptual graphs, in: International Conference on Artificial Intelligence and Soft Computing, volume 12855 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 149–158. doi:10.1007/978-3-030-87897-9_14.

[6] A. Petermann, G. Micale, G. Bergami, A. Pulvirenti, E. Rahm, Mining and ranking of generalized multi-dimensional frequent subgraphs, in: International Conference on Digital Information Management, IEEE, 2017, pp. 236–245. doi:10.1109/ICDIM.2017.8244685.

[7] T. Martin, V. Fuentes, P. Valtchev, A. B. Diallo, R. Lacroix, Generalized graph pattern discovery in linked data with data properties and a domain ontology, in: Symposium on Applied Computing, ACM, 2022, pp. 1890–1899. doi:10.1145/3477314.3507301.

[8] A. Sanfeliu, K. Fu, A distance measure between attributed relational graphs for pattern recognition, IEEE Trans. Syst. Man Cybern. 13 (1983) 353–362. doi:10.1109/TSMC.1983.6313167.

[9] D. B. Blumenthal, N. Boria, J. Gamper, S. Bougleux, L. Brun, Comparing heuristics for graph edit distance computation, VLDB J. 29 (2020) 419–458. doi:10.1007/s00778-019-00544-1.

[10] D. B. Blumenthal, S. Bougleux, J. Gamper, L. Brun, GEDLIB: A C++ library for graph edit distance computation, in: International Workshop on Graph-Based Representations in Pattern Recognition, volume 11510 of *Lecture Notes in Computer Science*, Springer, 2019, pp. 14–24. doi:10.1007/978-3-030-20081-7_2.

[11] A. K. Debnath, R. L. L. de Compadre, G. Debnath, A. J. Shusterman, C. Hansch, Structure-activity relationship of mutagenic aromatic and heteroaromatic nitro compounds. correlation with molecular orbital energies and hydrophobicity, J Med Chem 34 (1991) 786–797. doi:10.1021/jm00106a046.

[12] C. Helma, R. D. King, S. Kramer, A. Srinivasan, The predictive toxicology challenge 2000-2001, Bioinform. 17 (2001) 107–108. doi:10.1093/bioinformatics/17.1.107.

[13] C. Morris, N. M. Kriege, F. Bause, K. Kersting, P. Mutzel, M. Neumann, Tudataset: A collection of benchmark datasets for learning with graphs, in: ICML Workshop on Graph Representation Learning and Beyond, 2020. arXiv:2007.08663.

[14] W. Leal, G. Restrepo, A. Bernal, A network study of chemical elements: From binary compounds to chemical trends, MATCH Comm Math Comp Chem 68 (2012) 417–442.