Knowledge-Based Systems 66 (2014) 166–177

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Mining maximal frequent patterns in a single graph using inexact matching

M. Flores-Garrido *, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad

Instituto Nacional de Astrofísica, Óptica y Electrónica, Tonantzintla, Puebla, Mexico

ARTICLE INFO

Article history:
Received 24 October 2013
Received in revised form 25 April 2014
Accepted 25 April 2014
Available online 5 May 2014

Keywords:
Maximal frequent pattern
Inexact matching
Approximate patterns
Frequent patterns in a single graph
Graph mining

ABSTRACT

Despite being a popular problem during the last years, frequent graph mining has some important research lines still open for further exploration; among them, the possibility of using inexact matching in order to discover patterns otherwise missed, and the aim at reducing the set of mined patterns to facilitate their analysis and use. We present an algorithm that mines maximal patterns in a single graph using inexact matching. By allowing structural differences, in vertices as well as in edges, between a pattern and its occurrences, our algorithm is able to identify different, often larger, patterns than those found by other state-of-the-art algorithms. On the other hand, by focusing on maximal graphs the output set is significantly downsized without losing patterns, as they can be recovered from the maximal ones. Experiments show that the "extra" patterns found by our algorithm can help in supervised tasks like classification, thus suggesting that useful information about the input data can be captured through the maximal patterns mined with inexact matching.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

Graph-based data mining has received extensive attention in the last years, perhaps due to the growing popularity of graph databases; they are being used in areas like chemistry [1], decision making [2], document analysis [3], social networks analysis [4,5], among others.

Frequent graph patterns are subgraphs found in a graph collection, or a single graph, at least as many times as a user specified frequency threshold. These patterns have proven their usefulness in a variety of graph mining tasks and their mining process has become an important problem within the area of data mining.

In the most common scenario, the mining process takes place in a graph collection and aims to find *all* the frequent subgraphs with enough *exact* occurrences to satisfy the frequency threshold. Whether a subgraph is an occurrence of a given pattern, or not, is determined by solving the graph isomorphism problem, thus, occurrences are identical to the pattern they represent. Several algorithms have been developed, which successfully find frequent patterns in this standard scenario, among them, GraphSig [6], Gaston [7], gSpan [8] and gRed [9]. However, as new problems are modeled through graphs, new scenarios emerge in the mining of graph patterns.

The particular problem in which we are interested is to mine maximal frequent patterns in a single graph, using inexact matching. Mining frequent patterns from a single graph has been less explored than the case in which patterns are mined from a collection of graphs. Furthermore, focusing on the maximal patterns and using inexact matching are both characteristics tackling challenges that have recently arisen in graph mining.

First, due to the combinatorial explosion in the number of subgraph patterns, inherent to the problem, the mining result is often a large amount of patterns, making difficult the process of examining and using them. Because of this, in recent years a shift in mining algorithms has become noticeable, changing the focus from finding the whole set of frequent graph patterns, to finding a subset of them that would be easier to analyze due to its smaller size [10].

A common approach to filter out redundant patterns is to find *maximal* frequent graph patterns, i.e., frequent patterns that are not a subgraph of any other frequent pattern. As it has been pointed out [11,12], the set of maximal patterns could be orders of magnitude smaller than the complete set of patterns. Furthermore, non-maximal patterns can be reconstructed from the maximal patterns, thus, although information about the support of non-maximal patterns is not preserved, all the frequent

* Corresponding author. Tel.: +52 2221236872.
E-mail addresses: mflores@inaoep.mx (M. Flores-Garrido), ariel@inaoep.mx (J.A. Carrasco-Ochoa), fmartine@inaoep.mx (J.F. Martínez-Trinidad).

http://dx.doi.org/10.1016/j.knosys.2014.04.040
0950-7051/© 2014 Elsevier B.V. All rights reserved.

subgraphs are summarized in the maximal patterns without information loss. Although some algorithms have been proposed to find representative patterns [13], discriminative patterns [14], top largest patterns [15], among others, maximal patterns represent one of the most general yet effective approaches to reduce the set of frequent patterns.

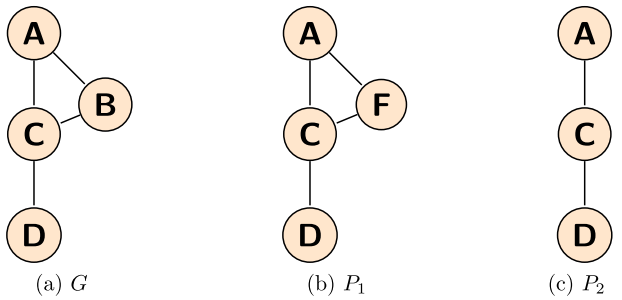
The second research direction that we aim to explore, and that can be spotted in the current state of graph mining, is working towards certain flexibility in the patterns that are being looked for in the data. Allowing inexact matching could be a good choice when working in certain contexts. Jia et al. [16] have posed the case in which it is required to mine useful patterns from a noisy graph database; in this case, it is important to allow differences in the labels of vertices or edges that have certain probability of being mislabeled. Similarly, Chen et al. [17] consider a protein database where frequent *approximate* patterns are "biologically interesting". Unlike these algorithms, we have interest in the use of inexact matching allowing structural differences in vertices, in such a way that two graphs with a different number of vertices could be considered a match. An example is provided in Fig. 1.

Thus, we have interest in reducing the amount of mined patterns and, paradoxically, we also want to generate additional patterns that do not match exactly their occurrences, but could be important in certain contexts by providing useful information that would be missed if exact matching were used.

In this work, we propose the algorithm MaxAFG (*Maximal Approximate Frequent Graphs*), which finds maximal frequent patterns in a single graph using inexact matching. We measure the similarity between graphs using a function based on the well known edit distance [18] and we follow a strategy that allows identifying maximal patterns with structural differences, in vertices and edges, respect to their occurrences. Although there are algorithms for finding frequent graphs that allow label substitutions and structural differences in edges [16,17], MaxAFG also allows structural differences in vertices.

Thus, the core of our contribution resides in providing an algorithm to find maximal patterns in a single graph in such a way that structural differences between patterns and their occurrences are allowed. In this way, we are able to find patterns missed by other algorithms reported in the literature, while, by finding only maximal patterns, we reduce the size of the output set and, somehow, counteract the increment in the amount of mined patterns brought by allowing inexact matching.

In the following section, we briefly describe the related work. Then, in Section 3, we present our proposed algorithm MaxAFG, giving details about the search strategy and the similarity function we use in it. In Section 4 we show experiments and results that support the usefulness of our algorithm. Finally, in Section 5, we point out some conclusions and future research lines.



(a) G (b) P_1 (c) P_2

Fig. 1. P_1 is an occurrence of G if label differences are admitted (label F is replacing label B); this is the type of inexact matching used by the algorithm APGM [16]. P_2 is an occurrence of G if structural differences are admitted (the vertice labeled as B is missed in P_2); this is the type of inexact matching that we consider in this work (besides label differences).

2. Related work

In this section we first describe algorithms that mine maximal frequent patterns in a collection of graphs; then we mention works that use inexact matching to mine frequent patterns; finally, we conclude the section with related works that mine frequent patterns in a single graph.

2.1. Maximal frequent patterns

In 2004 Huan et al. [11] brought attention to the problem of mining maximal subgraphs as a way of making more efficient the overall data mining process, decreasing the amount of storage needed and the number of mined patterns. Their algorithm, SPIN, finds all the frequent trees in a collection of graphs, then expands the trees into frequent cyclic graphs, and, finally, constructs the set of maximal frequent subgraphs, using some pruning techniques to make the maximal subgraph mining more efficient.

Later, in 2006, Thomas et al. [19] proposed the algorithm Margin to find maximal patterns in a collection of graphs. For each graph in the input set, they use a graph lattice to represent the search space and identify frequent maximal graphs candidates, which are frequent subgraphs with a non-frequent child. To this end, the authors first find a frequent connected subgraph and expand it until it is maximal; from that graph, represented (like every subgraph) by a point in the lattice, they traverse the lattice to identify other maximal candidates. Finally, in a post-processing step, the authors merge the candidates and select the maximal frequent patterns.

In 2012, Chen et al. [20] proposed a method to find maximal patterns in a collection of graphs by using a top-down mining strategy. First, the authors relabel the graphs in the collection, so that symmetric edges are identified by their label. Next, they construct a tree structure for the larger graph in the collection; each level in the tree consists of subgraphs obtained by removing infrequent edges from each graph in the previous level. Relying on the antimonotonicity property and using the symmetry information contained in edge labels, the algorithm removes edges until reaching a frequent graph, which is assumed to be maximal, since its parents are infrequent. Then, the algorithm continues by adding to the tree the remaining graphs in the collection, sorted in descending order respect to their size, finding for each graph its corresponding level in the tree and using isomorphism to compare it to subgraphs in the same level.

Other algorithms that allow finding maximal patterns, even if it is not their main focus, are FP-GraphMiner [21] and wgMiner [22]. FP-GraphMiner analyzes a collection of graphs by finding the frequent edges in all the graphs and creating a special undirected graph, the FP-Graph, that contains the frequency and structural information of the vertices and edges; from the FP-Graph, frequent graphs can be determined, including maximal ones. The algorithm wgMiner is designed to find closed and maximal patterns in a graph collection where each graph has related importance weights both internally, in its edges, and externally, with respect to the whole collection; the algorithm maximizes the importance or weight sum of the mined patterns, and performs a depth first search with traditional pruning in order to get maximal patterns.

2.2. Graph mining in a single graph

One of the first algorithms allowing pattern mining in a single graph is SUBDUE [23], although it does not find frequent patterns but uses the *Minimum Description Length* principle, a greedy algorithm and beam search, to mine the patterns that better compress the input graph.

Download English Version:

https://daneshyari.com/en/article/402338

Download Persian Version:

https://daneshyari.com/article/402338

Daneshyari.com