Mining Subjectively Interesting Attributed Subgraphs
Anes Bendimerad, Ahmad Mel, Jefrey Lijffijt, Marc Plantevit, Céline Robardet, Tijl de Bie

To cite this version:
Anes Bendimerad, Ahmad Mel, Jefrey Lijffijt, Marc Plantevit, Céline Robardet, et al.. Mining Subjectively Interesting Attributed Subgraphs. INTERNATIONAL WORKSHOP ON MINING AND LEARNING WITH GRAPHS, held with SIGKDD 2018, Aug 2018, Londres, United Kingdom. hal-02060190

HAL Id: hal-02060190
https://hal.archives-ouvertes.fr/hal-02060190
Submitted on 7 Mar 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Mining Subjectively Interesting Attributed Subgraphs

Work-in-progress paper

Anes Bendimerad∗
Univ Lyon, INSA, CNRS UMR 5205
F-69621 France
aabendim@liris.cnrs.fr

Ahmad Mel∗
IDLab, Ghent University
Ghent, Belgium
ahmad.mel@ugent.be

Marc Plantevit
Univ Lyon, UCBL, CNRS UMR 5205
F-69622 France
marc.plantevit@liris.cnrs.fr

Céline Robardet
Univ Lyon, INSA, CNRS UMR 5205
F-69621 France
celine.robardet@liris.cnrs.fr

Jefrey Lijffijt
IDLab, Ghent University
Ghent, Belgium
jefrey.lijffijt@ugent.be

Ahmad Mel∗
IDLab, Ghent University
Ghent, Belgium
ahmad.mel@ugent.be

Tijl De Bie
IDLab, Ghent University
Ghent, Belgium
tijl.debie@ugent.be

ABSTRACT
Community detection in graphs, data clustering, and local pattern mining are three mature fields of data mining and machine learning. In recent years, attributed subgraph mining is emerging as a new powerful data mining task in the intersection of these areas. Given a graph and a set of attributes for each vertex, attributed subgraph mining aims to find cohesive subgraphs for which (a subset of) the attribute values has exceptional values in some sense. While research on this task can borrow from the three abovementioned fields, the principled integration of graph and attribute data poses two challenges: the definition of a pattern language that is intuitive and lends itself to efficient search strategies, and the formalization of the interestingness of such patterns. We propose an integrated solution to both of these challenges. The proposed pattern language improves upon prior work in being both highly flexible and intuitive. We show how an effective and principled algorithm can enumerate patterns of this language. The proposed approach for quantifying interestingness of patterns of this language is rooted in information theory, and is able to account for prior knowledge on the data. Prior work typically quantifies interestingness based on the cohesion of the subgraph and for the exceptionality of its attributes separately, combining these in a parameterized trade-off. Instead, in our proposal this trade-off is implicitly handled in a principled, parameter-free manner. Extensive empirical results confirm the proposed pattern syntax is intuitive, and the interestingness measure aligns well with actual subjective interestingness.

KEYWORDS
Graphs, Networks, Subjective Interestingness, Subgraph Mining

1 INTRODUCTION
The availability of network data has surged both due to the success of social media and ground-breaking discoveries in experimental sciences. Consequently, graph mining is one of the most studied tasks for the data mining community. The value of graphs stems from the presence of meaningful relationships among the data objects (the vertices). These can be explored by approaches as different as graph embeddings[7]—which map the nodes of a graph into a low dimensional space while preserving the local and global graph structure as well as possible—, community detection[9]—the discovery of groups of vertices that somehow ‘belong together’—, or subgraph mining—the identification of informative subgraphs.

Besides the relational structure, graphs may carry information in the form of attribute-value pairs on vertices and/or edges. Such graphs are called attributed graphs[16, 18, 20]. We focus here on graphs with attribute-value pairs on vertices (vertex-attributed graphs). Mining interesting subgraphs in attributed graphs is challenging, both conceptually and computationally. A prominent problem is defining the interestingness of a subgraph. Desirable properties of subgraph would be that it is cohesive (the attribute values of the vertices are similar) and that the vertices form an easy to describe pattern in the graph (e.g., vertices should be close to each other). A specific form of subgraph mining is to look for exceptional subgraphs, i.e., subgraphs whose attribute values are cohesive within the subgraph but exceptional in the full graph.

Few works in this direction exist. For example Atzmueller et al.[1] study mining communities (densely connected subgraphs) that can also be described well in terms of attribute values, while Bendimerad et al. [2] look for exceptional subgraphs that are connected. Various quality measures are used in the first work and the second relies on Weighted Relative Accuracy. We introduce a new generically applicable interestingness measure for exceptional subgraph patterns based on Information Theory which is more flexible and can incorporate prior knowledge about the graph to steer the scoring of subgraph patterns. Besides, while previous works use certain hard constraints to arrive at subgraphs that are somehow interpretable, we integrate the interpretability into the interestingness measure. Hence, the trade-off between informativeness and interpretability can be made in a principled manner.

More specifically, we consider the problem to identify informative subgraphs that can be concisely described. The informativeness of a subgraph depends on the number of vertices it covers (more is better) and how surprising the statistics (attribute values) of those vertices are. Surprise is important because showing the user statistics they expect to see does not teach them anything. Vertices that are spread out over the graph, or that share no statistics cannot be summarized well, so the end-user cannot generalize over the structure of the vertices in a pattern and hence the graph structure is effectively transmitted to a user without any compression. Here instead, we look for subgraphs that are both homogeneous and
Vis a set of the centre of that area (which is average). There are ‘surprisingly’ many food establishments, except in nightlife and food venues but limited educational venues. Several types of venues are consistently overrepresented.

Figure 1: Example results on a graph based on Foursquare data covering the presence of various types of venues in London. P₁: around the orange block (west/south of Hyde Park) there are ‘surprisingly’ many food establishments, except in the centre of that area (which is average). P₂: in the City several types of venues are consistently overrepresented. P₃: around Hackney there is a strip of blocks with lots of nightlife and food venues but limited educational venues.

Localized, hence possess shared properties which can be exploited to produce concise descriptions.

Fig. 1 shows example patterns from our method, applied to a setting where we want to explore the district structure of cities. The patterns should be interpreted as follows: certain attributes have surprisingly high/low values (marked +/- respectively) in the given neighbourhoods as compared to a background model. We ensure the regions are localized by forcing them to have a description of the form “all vertices that are within distance d₁ of vertex x AND within distance d₂ of vertex y AND etc.”. E.g., in pattern P₂ of Fig. 1, the covered areas (green blocks) are the intersection of blocks within distance two of either purple block.

Contributions. We present a pattern syntax for cohesive subgraphs with exceptional attributes (Sec. 2). We formalize their subjectivity interestingness in a principled manner using information theory, accounting for both the information they provide, as well as their interpretability (Sec. 3). We study how to mine such subgraphs efficiently (Sec. 4). We provide a thorough empirical study on real data that evaluates (1) the relevance of the subjective interestingness measure compared to state-of-the-art methods, and (2) the efficiency of the algorithms (Sec. 5). We discuss related work in Sec. 6 and the conclusions are presented in Sec. 7.

2 COHESIVE SUBGRAPHS WITH EXCEPTIONAL ATTRIBUTES

Before formally introducing the pattern language we are interested in, let us establish some notation:

Notation An attributed graph is denoted \( G = (V, E, \hat{\mathcal{A}}) \), where \( V \) is a set of \( n \) vertices, \( E \subseteq V \times V \) is a set of \( m \) edges, and \( \hat{\mathcal{A}} \) is a set of \( p \) numerical attributes on vertices (formally, functions mapping a vertex onto an attribute value), with \( \hat{a}(v) \in \text{Dom}_a \) denoting the value of attribute \( \hat{a} \in \hat{\mathcal{A}} \) on \( v \in V \). We use hats in \( \hat{a} \) and \( \hat{\mathcal{A}} \) to signify the empirical values of the attributes, whereas \( a \) and \( \mathcal{A} \) denote (possibly random) variables over the same domains. We also define the function \( N_d(v) \) to denote the neighborhood of range \( d \) of a vertex \( v \), i.e., the set of vertices whose geodesic distance to \( v \) is at most \( d \):

\[
N_d(v) = \{ u \in V \mid \text{dist}(v, u) \leq d \}.
\]

Cohesive Subgraphs with Exceptional Attributes (CSEA) As described in the introduction, we are interested in patterns that inform the user that a given set of attributes has exceptional values throughout a set of vertices in the graph.

Thus, and more formally, a CSEA pattern is defined as a tuple \((U, S)\), where \( U \subseteq V \) is a set of vertices in the graph, and \( S \) is a set of restrictions on the value domains of the attributes of \( A \), or more specifically, \( S = \{ [k_a, \ell_a] \mid a \in A \} \). A pattern \((U, S)\) is said to be contained in \( G \) iff

\[
\forall \{k_a, \ell_a\} \in S \text{ and } \forall u \in U, \ k_a \leq \hat{a}(u) \leq \ell_a.
\]

Informally speaking, a CSEA pattern will be more informative if its description is more concise in some natural easier-to-interpret definition. Thus, along with the pattern language, we must also specify how a pattern from this language will be intuitively described.

To this end, we propose to describe the set of vertices \( U \) as a neighborhood of a specified range from a given specified vertex, or more generally as the intersection of a set of such neighborhoods. For enhanced expressive power, we additionally allow for the description to specify some exceptions on the above: vertices that do fall within this (intersection of) neighborhood(s), but which are to be excluded from \( U \). Exceptions are a detriment to the interestingness of a pattern, but we can discount these naturally.

A premise of this paper is that this way of describing the set \( U \) is intuitive for human analysts, such that the length of the description of a pattern, as discussed in detail in Sec. 3.2, is a good measure of the complexity to assimilate or understand it. Our qualitative experiments in Sec. 5 do indeed confirm this is the case.

3 THE SUBJECTIVE INTERESTINGNESS OF A CSEA PATTERN

The previous sections already hinted at the fact that we will formalize the interestingness of a CSEA pattern \((U, S)\) by trading off its information content with its description length. Here we show how the FORSIED framework for formalizing subjective interestingness of patterns, introduced in [3, 4], can be used for this purpose.

The information content depends on both \( U \) and \( S \). It is larger when more vertices are involved, when the intervals are narrower, and when they are more extreme. We will henceforth denote the information content as \( IC(U, S) \). The description length depends on \( U \) only (as the attribute ranges require a fixed description length), and will be denoted as \( DL(U) \). The subjective interestingness of a
CSEA pattern \((U, S)\) is then expressed as:

\[
SI(U, S) = \frac{\text{IC}(U, S)}{\text{DL}(U)}.
\]

One of the core capabilities of the FORSIEED framework is that it quantifies the information content of a pattern against a prior belief state about the data. It rigorously models the fact that the more plausible the data is (subjectively) according to a user or (objectively) under a specified model, the less information a user receives, and thus the smaller the information content ought to be.

This is achieved by modeling the prior beliefs of the user as the Maximum Entropy (MaxEnt) distribution subject to any stated prior beliefs the user may hold about the data. This distribution is referred to as the background distribution. The information content \(\text{IC}(U, S)\) of a CSEA pattern \((U, S)\) is then formalized as minus the logarithm of the probability that the pattern is present under the background distribution (also called the self-information or surprisal) [8]:

\[
\text{IC}(U, S) = -\log(\Pr(U, S)).
\]

In Sec. 3.1, we first discuss in greater detail which prior beliefs could be appropriate for CSEA patterns, and how to infer the corresponding background distribution. Then, in Sec. 3.2, we discuss in detail how the description length \(\text{DL}(U)\) can be computed.

### 3.1 The information content of a CSEA pattern

#### Positive integers as attributes

For concreteness, let us consider the situation where the attributes are positive integers \((a : V \to \mathbb{N}, \forall a \in A)\), as will be our main focus throughout this paper.\(^1\)

For example, if the vertices are geographical regions (with edges connecting vertices of neighboring regions), then the attributes could be counts of particular types of places in the region (e.g., one attribute could be the number of shops). It is clear that it is less informative to know that an attribute value is large in a large region than it would be in a small region. Similarly, a large value for an attribute that is generally large is less informative than if it were generally small. The above is only true, however, if the user knows (or believes) a priori at least approximately what these averages are for each attribute, and what the ‘size’ of each region is. Such prior beliefs can be formalized as equality constraints on the values of the attributes \(A\) on all vertices, or mathematically:

\[
\sum_{A} \Pr(A) \left( \sum_{v \in V} a(v) \right) = \sum_{v \in V} \hat{a}(v), \quad \forall a \in A,
\]

\[
\sum_{A} \Pr(A) \left( \sum_{v \in V} a(v) \right) = \sum_{v \in V} \hat{a}(v), \quad \forall a \in A.
\]

The MaxEnt background distribution can then be found as the probability distribution maximizing the entropy \(-\sum_{A} \Pr(A) \log \Pr(A)\), subject to these constraints and the normalization \(\sum_{A} \Pr(A) = 1\).

As shown in [4], the optimal solution of this optimization problem is a product of independent Geometric distributions, one for each vertex attribute-value \(a(v)\). Each of these Geometric distributions is of the form \(\Pr(a(v) = z) = \rho_{av}(1-\rho_{av})^{z}, \ z \in \mathbb{N}\), where \(\rho_{av}\) is the success probability and it is given by: \(\rho_{av} = 1 - \exp(X_{av} + \lambda_{av}^{c})\), with \(X_{av}\) and \(\lambda_{av}\) the Lagrange multipliers corresponding to the two constraint types. The optimal values of these multipliers can be found by solving the convex Lagrange dual optimization problem.

Given these Geometric distributions for the attribute values under the background distribution, we can now compute the probability of a pattern \((U, S)\) as follows:

\[
\Pr(U, S) = \prod_{v \in U} \prod_{\{ka, \ell_{ca}\} \in S} \Pr(a(v) \in \{ka, \ell_{ca}\}),
\]

\[
= \prod_{v \in U} \prod_{\{ka, \ell_{ca}\} \in S} \left(1 - p_{av} \right)^{\ell_{ca}} - (1 - p_{av})^{\ell_{ca}+1}.
\]

This can be used directly to compute the information content of a pattern on given data, as the negative log of this probability. However, the pattern syntax is not directly suited to be applied to count data, when different vertices have strongly differing total counts. The reason is that the interval of each attribute is the same across vertices, which is desirable to keep the syntax understandable. Yet, if neighboring regions have very different total counts, it is less likely to find any patterns, and, even if we do, end-users would still need to know the total counts to interpret the patterns properly, as the same interval is not equally surprising for each region.

#### \(p\)-values as attributes

To address this problem, we propose to search for the patterns not on the counts themselves, but rather on their significance (i.e., \(p\)-value or tail probability), computed with the background distribution as null hypothesis in a one-sided test. More specifically, we define the quantities \(\hat{c}_{a}(v)\) as

\[
\hat{c}_{a}(v) = \Pr(a(v) \leq \hat{a}(v)),
\]

and use this instead of the original attributes \(a(v)\). This transformation of \(\hat{a}(v)\) to \(\hat{c}_{a}(v)\) can be regarded as a principled normalization of the attribute values to make them comparable across vertices.

To compute the IC of a pattern with the transformed attributes \(\hat{c}_{a}\), we must be able to evaluate the probability that \(ca(v)\) falls within a specified interval \([\ell_{ca}, \ell_{ca}]\) under the background distribution for \(a(v)\). This is given by:

\[
\Pr(ca(v) \in [\ell_{ca}, \ell_{ca}]) = \Pr \left( (1 - p_{av})^{\ell_{ca}} - (1 - p_{av})^{\ell_{ca}+1} \right),
\]

\[
= \Pr \left( a(v) \leq \frac{\log(\ell_{ca})}{\log(1 - p_{av})} \right) \land a(v) \geq \frac{\log(\ell_{ca})}{\log(1 - p_{av})},
\]

\[
= \left(1 - p_{av}\right)^{\log(\ell_{ca})} - (1 - p_{av})^{\log(\ell_{ca})+1},
\]

\[
= \ell_{ca} - (1 - p_{av})^{\ell_{ca}},
\]

\[
= k_{ca} + p_{av}^{\ell_{ca}}.
\]

Thus, the IC of a pattern on the transformed attributes \(\hat{c}\) can be calculated as:

\[
\text{IC}(U, S) = -\log(\Pr(U, S)),
\]

\[
= - \sum_{[\ell_{ca}, \ell_{ca}] \in S} \sum_{v \in U} \log(\ell_{ca} + k_{ca} + p_{av}k_{ca}). \tag{2}
\]

In this paper, we focus on intervals \([k_{ca}, \ell_{ca}]\) where either \(k_{ca} = 0\) (the minimal value) or \(\ell_{ca} = 1\) (the maximal value). Such intervals state that the values of an attribute are all significantly large\(^2\) or significantly small respectively, for all vertices in \(U\). We argue such

\(^1\)The presented results can be extended relatively straightforwardly for other cases.

\(^2\)Note empty regions have tail probabilities \(\hat{e}_{a}(v) = 1\) for any attribute and thus fall within any upper interval, but also IC = 0 for any attribute of that region as both \(k_{ca} = 1\) and \(\rho_{av} = 1\).
An overview of the method is given in Algorithm 1 and explained further below.

**Algorithm 1: SIAS-Miner-Enum**

**Input:** $G = (V, E, \hat{A})$, $D$

**Output:** $Result$, the set of CSEA patterns $(U, S)$ that are closed simultaneously wrt. $U$, $S$, and the neighbourhood description. Second, it ranks patterns according to their SI values.

1. // transformation to entity-relation model
2. $D \leftarrow $ transformToER$(G, D)$
3. // enumeration of the patterns
4. $Result \leftarrow P-N-RMiner(D)$
5. // computation of the minimum description for each found pattern
6. $\text{minDesc} \leftarrow \{\}$
7. for $(U, S), N(U) \in Result$ do
8. \hspace{0.5cm} $\text{bestDesc} \leftarrow \emptyset$
9. \hspace{1cm} DL-Optimise($U, 0, N(U), \text{bestDesc}$)
10. \hspace{0.5cm} $\text{minDesc}[[U,S]] \leftarrow \text{bestDesc}$

4.1 Pattern enumeration

In the first step, we enumerate candidate tuples $(U, S)$ where the vertices $U$ can be concisely described as an intersection of neighbourhoods $X \subseteq N(U)$. Disregarding the description for a moment, since the pattern syntax is chosen such that each interval $[a_i, b_i] \in S$ should cover every vertex $u \in U$ (Eq. 1), the IC of a tuple $(U, S)$ increases monotonically by adding vertices to $U$ and intervals to $S$. Hence, to decrease the number of candidate patterns and increase computational efficiency we focus on closed patterns, i.e., tuples $(U, S)$ where no vertex can be added without enlarging intervals and where no interval can be reduced without omitting vertices.

However, in Section 2 we additionally argued that a pattern $(U, S)$ where the vertices $u \in U$ are unrelated will be difficult to understand and remember, which we expressed in the DL. Since computing $DL(U)$ is NP-Complete, it is not clear that enumeration of closed patterns wrt. the SI can be done. Nonetheless, enumeration of all closed $(U, S)$ appears wasteful because most sets $U$ will have a high DL. What appears feasible is to restrict enumeration of sets $U$ that are exactly intersections of neighbourhoods, i.e., a description without any exceptions.

While closed sets $(U, S)$ may be most efficiently enumerated by an itemset mining algorithm, if we want $(U, S)$ to additionally be closed with respect to intersections of neighbourhoods this yields a relational schema with two relations: (1) vertices are connected to all intervals that cover their attribute values, and (2) vertices are connected to every neighborhood that they are contained in. A tool that would indeed enumerate precisely the required closed patterns and no other is $RMiner$.

In general, there are multiple ways of describing a given set of nodes $U$, by using a various combinations of neighborhoods. The best one is thus the one that minimizes $f$. This finally leads us to the definition of the description length of a pattern as:

$$DL(U) = \min_{X \subseteq N(U)} f(X, U).$$

4 AN ENUMERATION APPROACH TO MINING INTERESTING CSEA PATTERNS

SIAS-Miner-Enum mines interesting patterns using an enumerate-and-rank approach. First, it enumerates all CSEA patterns $(U, S)$ that are closed simultaneously wrt. $U$, $S$, and the neighbourhood description. Second, it ranks patterns according to their SI values.

Notice that such a description does not necessarily minimize the description length of a vertex set $U$.\footnote{Notice that such a description does not necessarily minimize the description length of a vertex set $U$.}
right for an example. In the relational context, a pattern is called a
maximal CCS (MCCS) if no entity can be added, i.e., patterns we
referred to as closed in the discussion above. It is worth to notice
that an MCCS provides, in addition to a tuple \((U, S)\), the set \(N(U)\)
of neighborhoods that contain \(U\), from which DL\((U)\) is computed.

Notice P-N-RMiner also ranks patterns based on interestingness
under a known-degree background model, but that is not useful in
our setting as the IC and the DL are very different here. Hence,
we only use it to enumerate all candidates. The computational
complexity is clearly exponential as the number of outputs may be
exponential in the size of the input, plus in this setting no fixpoint-
enumeration-based algorithm may have polynomial-time delay [15].
Scalability experiments are presented in Sec. 5.

4.2 Computing DL\((U)\)

The calculation of DL\((U)\) is NP-Complete and equivalent to Set
Cover: it consists in finding the optimal cover of the set \(U\) based on
unions of complements \(N_i(v)\) and exceptions \(e\) such that \(x \in U\).
Nevertheless, we propose a branch-and-bound approach that takes
benefit from several optimisation techniques.

In order to find the optimal description of a pattern \((U, S)\), we
explore the search space \(2^{N(U)}\) with a branch-and-bound approach
described in Algorithm 2. Let \(X\) and Cand be subsets of \(N(U)\)
that are respectively the current enumerated description and the
potential candidates that can be used to describe \(U\). Initially, DL-
Optimise is called with \(X = \emptyset\) and \(\text{Cand} = N(U)\). In each call, a
neighborhood \(e \in \text{Cand}\) is chosen and used to recursively explore
two branches: one made of the descriptions that contain \(e\) (by adding \(e\) to \(X\)), and the other one made of descriptions that do not
contain \(e\) (by removing \(e\) from \(\text{Cand}\)). Several pruning techniques
are used in order to reduce the search space and are detailed below.

**Function LB (line 1)** lower bounds the lengths of the descriptions
that can be generated in the subsequent recursive calls of DL-Optimise. If LB is higher or equal than the length of the current
best description of \(U\) \(f(\text{bestDesc}, U)\), there is no need to carry on
the exploration of the search subspace as no further description
can improve \(f\). The principle of LB is to evaluate the maximum
reduction in exceptions that can be obtained when description \(X\)
is extended with neighborhoods of \(Y\):

\[
\text{gain}_Y(X, U) = |\text{exc}(X, U)| - |\text{exc}(X \cup Y, U)|, \text{ with } Y \subseteq \text{Cand}.
\]

**Algorithm 2: DL-Optimise\((U, X, \text{Cand}, \text{bestDesc})\)**

**Input:** \(U\) the set of vertices to describe, \(X\) the current
enumerated description, \(\text{Cand}\) the set of candidates,
\(\text{bestDesc}\) the current best description found.

**Output:** \(\text{bestDesc}\) the best description found while
exploring the current search sub-space.

1. if \(LB(X, U, \text{Cand}) < f(\text{bestDesc}, U)\) then
2. elseif \(\text{Cand} = \emptyset\) then
3. \(\text{pruneUseless}(U, X, \text{Cand})\)
4. \(\text{pruneLowerBounded}(U, X, \text{Cand})\)
5. \(e \leftarrow \arg\min_{e \in \text{Cand}} f(X \cup \{e\}, U)\)
6. DL-Optimise\((U, X \cup \{e\}, \text{Cand} \setminus \{e\}, \text{bestDesc})\)
7. DL-Optimise\((U, X, \text{Cand} \setminus \{e\}, \text{bestDesc})\)
8. else if \(f(X, U) < f(\text{bestDesc}, U)\) then
9. \(\text{bestDesc} \leftarrow X\)

This function can be rewritten using neighbourhood complements as
\(\text{gain}_e(X, U) = |\bigcup_{Y \subseteq \text{Cand}} (\bigcap N_i(Y) \setminus \text{exc}(X, U))|\) \(\bigcup \{g_i\}_{i \in \text{Cand}}\). We can obtain an upper
bound of the gain function using the ordered set \{\(g_i\), \ldots, \(g_{|\text{Cand}|}\)\}
of \(\text{gain}(X, U, e)\) \(1 \leq e \leq \text{Cand}\) such that \(g_i \geq g_j\) if \(i < j\):

**Property 1.** \(\text{gain}_Y(X, U) \leq \sum_{i=1}^{|Y|} g_i\), for \(Y \subseteq \text{Cand}\).

**Proof.** Since the size of the union of sets is lower than the sum of
the set sizes, we have \(\text{gain}_Y(X, U) \leq \sum_{g \in \text{Cand}} (\bigcap N_i(g) \setminus \text{exc}(X, U)) \leq \sum_{i=1}^{|Y|} g_i\).

This is the foundation of the function LB defined as

\[
LB(X, U, \text{Cand}) = \min_{e \in \{0, |\text{Cand}|\}} \left(|X| + i + 1\right) \times \log(|N|)
\]

\[
+ \left(1 + \max \left(0, |\text{exc}(X, U)| - \sum_{i=1}^{|Y|} g_i\right) \times \log(|V|)\right) \quad (4)
\]

**Property 2.** \(f(X \cup Y, U) \geq LB(X, U, \text{Cand})\), for all \(Y \subseteq \text{Cand}\).

**Proof.** Based on Property 1, we have \(|\text{exc}(X \cup Y, U)| \geq \max(0, \text{exc}(X, U) - \sum_{i=1}^{|Y|} g_i)\). This means that \(f(X \cup Y, U) \geq (|X| + |Y| + 1) \times \log(|N|) + \left(1 + \max(0, |\text{exc}(X, U)| - \sum_{i=1}^{|Y|} g_i)\right) \times \log(|V|)\) and thus.

\(LB(X, U, \text{Cand}) \leq (|X| + |Y| + 1) \times \log(|N|) + \left(1 + \max(0, |\text{exc}(X, U)| - \sum_{i=1}^{|Y|} g_i)\right) \times \log(|V|)\) and it concludes the proof.

\(4 = |\text{exc}(X, U)| - |\text{exc}(X \cup U)| = |\bigcap_{i \in \text{Cand}} N_i(X) \setminus \bigcup_{i \in \text{Cand}} N_i(Y)\) \(\setminus \bigcup_{i \in \text{Cand}} N_i(Y)\)\)

\(= |\bigcup_{i \in \text{Cand}} N_i(X) \setminus \bigcup_{i \in \text{Cand}} N_i(Y)\) \(\setminus \bigcup_{i \in \text{Cand}} N_i(Y)\)\)

\(= |\text{exc}(X, U) \cap \bigcap_{i \in \text{Cand}} N_i(Y)\) \(\setminus \bigcup_{i \in \text{Cand}} N_i(Y)\)

\(= |\text{exc}(X, U) \cap \bigcap_{i \in \text{Cand}} N_i(Y)\) \(\setminus \bigcup_{i \in \text{Cand}} N_i(Y)\)\)

\(= |\text{exc}(X, U) \cap \bigcup_{i \in \text{Cand}} N_i(Y)\) \(\setminus \bigcup_{i \in \text{Cand}} N_i(Y)\)\)
Algorithm 3: pruneUseless(U,X,Cand)
1. Cand ← {e ∈ Cand | gain(e), X, U} > 0

Algorithm 4: pruneLowerBounded(U,X,Cand)
1. Cand ← {e1 ∈ Cand | ∀e2 ∈ Cand \ {e1} : (exc(X ∪ {e2}, U) ∉ exc(X ∪ {e1}, U)) ∨ (exc(X ∪ {e1}, U) = exc(X ∪ {e2}, U) ∧ i < j)}

In other terms, in the recursive calls, a description length will never be lower than LB(X, U, Cand).

Function pruneUseless line 3 removes candidate elements that can not improve the description length, that is candidates e ∈ Cand for which gain((e), X, U) = 0. Such element does not have the ability to reduce the number of exceptions in X. This also implies that e will not reduce the number of exceptions for descriptions X ∪ Y, with Y ⊆ Cand. Thus, such elements will not decrease the description length of X ∪ Y.

Function pruneLowerBounded line 4 removes a candidate e ∈ Cand if there is a candidate e′ ∈ Cand that is always better than e for all descriptions produced in subsequent recursive calls.

Property 3. Let e, e′ ∈ Cand such that exc(X ∪ {e}, U) ⊆ exc(X ∪ {e′}, U). Then, for all Y ⊆ Cand \ {e, e′}, we have f(X ∪ Y ∪ {e}, U) ≤ f(X ∪ Y ∪ {e′}, U)

Proof. The set of exceptions in X ∪ Y ∪ {e} is equal to exc(X ∪ Y ∪ {e}, U) = exc(X ∪ {e}, U) ∧ exc(Y, U). Since exc(X ∪ {e}, U) ⊆ exc(X ∪ {e′}, U), then exc(X ∪ Y ∪ {e}, U) ⊆ exc(X ∪ Y ∪ {e′}, U). As [X ∪ Y ∪ {e}] = [X ∪ Y ∪ {e′}], we can conclude that f(X ∪ Y ∪ {e}, U) ≤ f(X ∪ Y ∪ {e′}, U). □

Based on Property 3, pruneLowerBounded removes elements e′ ∈ Cand such that exc(X ∪ {e}, U) ⊆ exc(X ∪ {e′}, U). Notice that even if an element e′′ has been removed due to the lower bound of e′, the procedure is still correct since e′′ is lower bound by e by the transitivity of inclusion.

The last optimisation consists in choosing e ∈ Cand that minimises f(X ∪ {e}, U) (line 5 of Algorithm 2). This makes it possible to quickly reach descriptions with low DL, and subsequently provide effective pruning when used in combination with LB.

5 EXPERIMENTS

In this section, we report our experimental results. We start by describing the real-world dataset we used, as well as the questions we aim to answer. Then, we provide a thorough comparison with the state-of-the-art algorithm Cenergetics [2]. Eventually, we provide a qualitative analysis that demonstrates the ability of our approach to achieve the desired goal. For reproducibility purposes, the source code and the data are made available here.5

Experimental setting. Experiments are performed on the real-world dataset of the London graph. The London graph (|V| = 289, |E| = 544, |A| = 10) is based on the social network Foursquare. Each vertex represents a district in London, and edges link adjacent districts. Each attribute stands for the number of places of a given type (e.g. outdoors, colleges, residences, restaurants, etc.) in each district. Considering all the numerical values of attributes is computationally expensive and would lead to redundant results, we pre-process the graph so that for each attribute, the values are binned into five quantiles.

Aims. As stated in Section 6, there is no approach that supports the discovery of subjectively interesting attributed subgraphs in the literature. The closest method to SIAS-Miner-Enum is Cenergetics [2] that aims at discovering closed exceptional attributed subgraphs involving overrepresented and/or underrepresented attributes, and which mined the London graph used here in the experiments (and on similar graphs of other cities). It assesses exceptionality with the weighted relative accuracy (WRAcc) measure that accounts for margins but cannot account for other prior knowledge. The computational problem we tackle is more complex than Cenergetics, but how much is this overhead? Is it worth it in terms of pattern quality? This empirical study aims to answer to these questions.

Quantitative experiments. Fig. 3 reports the execution time per pattern, the number of discovered patterns, and the average quality of the top 500 patterns (i.e., DL and SI) of SIAS-Miner-Enum and Cenergetics according to the number of vertices, the number of attributes and the minimum number of vertices of searched patterns. We post-processed the results of Cenergetics in order to obtain similar redescriptions of the vertices as in SIAS-Miner-Enum, but we do not consider this post-processing step in the reported execution times. These tests reveal that the computational overhead of SIAS-Miner-Enum is important: the discovery of a pattern by SIAS-Miner-Enum is generally one to two orders of magnitude more costly than Cenergetics. However, SIAS-Miner-Enum provides patterns of better quality. Indeed, the average description length of the top 500 patterns discovered by SIAS-Miner-Enum is smaller than those of Cenergetics and the SI of CSEA patterns is greater than the one of the patterns extracted by Cenergetics.

Qualitative experiments. Finally, we show some examples of patterns (with at least 5 vertices) discovered by both SIAS-Miner-Enum and Cenergetics on London graph. The top 4 CSEA patterns discovered by SIAS-Miner-Enum are given in Tab. 1 and displayed in Fig. 4. Green cells represent vertices covered by a CSEA pattern while blue cells are the centers, purple cells are the centers that do not belong to the pattern, orange cells are centers that are also exception (i.e., behave differently from the pattern but covered by the description) and the red cells are normal exceptions. We also report the top 4 patterns discovered by Cenergetics in Fig. 5. Interestingly, CSEA patterns are more cohesive than Cenergetics ones, described by at most two Neighborhoods, which eases the assimilation by an analyst. Unexpectedly, the second best patterns of SIAS-Miner-Enum and Cenergetics are somewhat similar: the CSEA pattern covers two additional vertices and the two patterns are in agreement on some overrepresented types of venues (e.g., nightlife, shops). Surprisingly, outdoor venues are consistently overrepresented in the CSEA pattern while such venues are considered as limited by Cenergetics. This inconsistency may be due to some extreme values in some regions that impact the mean and then the value of the WRAcc measure by Cenergetics.

Summary. Even if our approach has an obvious computational overhead compared to Cenergetics (the problem tackled is more complex), these experiments show the ability of SIAS-Miner-Enum to discover CSEA patterns that are more intuitive and informative.
Figure 3: SIAS-Miner-Enum vs Cenergetics: runtime per pattern (first column), #patterns (second column), average description length (third column) and subjective interestingness (fourth column) of the top 500 patterns for varying $|V|$ (1st row), $|A|$ (2nd row) and a threshold on the minimum number of vertices in searched patterns (3rd row) for London graph ($D = 3$).

Figure 4: Top 4 patterns discovered in London graph by SIAS-Miner-Enum ($\minVertices = 5, D = 3$). Details are provided in Tab. 1

<table>
<thead>
<tr>
<th>Pattern ID</th>
<th>Characteristics: $S = {s_i, [l_i, k_i]}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_1$</td>
<td>food: $[0, 0.47]$, college: $[1, 1]$, event: $[1, 1]$, art: $[0.57, 1]$</td>
</tr>
<tr>
<td>$P_2$</td>
<td>shop: $[0, 0.43]$, nightlife: $[0, 0.44]$, travel: $[0, 0.44]$, college: $[0, 0.47]$, outdoors: $[0, 0.47]$</td>
</tr>
<tr>
<td>$P_3$</td>
<td>food: $[0.31]$</td>
</tr>
<tr>
<td>$P_4$</td>
<td>food: $[0.47]$</td>
</tr>
</tbody>
</table>

Table 1: Detailed characteristics of the top 6 patterns discovered in London dataset by SIAS-Miner-Enum (see Fig. 4).

6 RELATED WORK

Several approaches have been designed to discover new insights in vertex attributed graphs. The pioneering work of Moser et al. [16] presents a method to mine dense homogeneous subgraphs, i.e., subgraphs whose vertices share a large set of attributes. Similarly, Günnemann et al. [10] introduce a method based on subspace clustering and dense subgraph mining to extract non redundant subgraphs that are homogeneous with respect to the vertex attributes. Silva et al. [20] extract pairs made of a dense subgraph and a Boolean attribute set such that the Boolean attributes are strongly associated with the dense subgraphs. In [18], the authors propose to mine the graph topology of a large attributed graph by finding regularities among numerical vertex descriptors. The main objective of all these approaches is to find regularities instead of peculiarities within a large graph, whereas Exceptional Subgraph Mining mines subgraphs with distinguishing characteristics.

Interestingly, a recent work [1] proposes to mine descriptions of communities from vertex attributes, with a Subgroup Discovery approach. In this supervised setting, each community is treated as a target that can be assessed by well-established measures, like WRAcc. In [12], the authors aim at discovering contextualized subgraphs that are exceptional with respect to a model of the data. Restrictions on the attributes, that are associated to edges, are used to generate subgraphs. Such patterns are of interest if they pass a statistical test and have high value on an adapted WRAcc.
We have introduced a new pattern language in attributed graphs. A so-called CSEA pattern provides to the user a set of attributes and vertex attributes as an intersection of neighborhoods within a chosen distance of selected vertices. The distance and vertices making up the description of the subgraph. We have shown how an effective and principled algorithm can enumerate patterns of this language. Extensive empirical results on two real-world datasets confirm that CSEA patterns are intuitive, and the interestingness measure aligns well with actual subjective interestingness. This paper opens up several avenues for further research such as the development of speed-ups of SIAS-Miner-Enum and how to incorporate non-ordinal attribute types in the pattern syntax and interestingness measure.

Acknowledgements. This work was supported by the ERC under the EU’s Seventh Framework Programme (FP7/2007-2013) / ERC Grant Agreement no. 615517, FWO (project no. G091017N, G0F9816N), the EU’s Horizon 2020 research and innovation programme and the FWO under the Marie Sklodowska-Curie Grant Agreement no. 665501 and Group Image Mining (GIM) which joins researchers of THALES Group and LIRIS Lab. This work is also partially supported by the CNRS project APRC Conf Pap.

REFERENCES


