An efficiently computable subgraph pattern support measure: counting independent observations

Yuyi Wang · Jan Ramon · Thomas Fannes

Received: 18 November 2012 / Accepted: 19 April 2013
© The Author(s) 2013

Abstract Graph support measures are functions measuring how frequently a given subgraph pattern occurs in a given database graph. An important class of support measures relies on overlap graphs. A major advantage of overlap-graph based approaches is that they combine anti-monotonicity with counting the occurrences of a subgraph pattern which are independent according to certain criteria. However, existing overlap-graph based support measures are expensive to compute. In this paper, we propose a new support measure which is based on a new notion of independence. We show that our measure is the solution to a sparse linear program, which can be computed efficiently using interior point methods. We study the anti-monotonicity and other properties of this new measure, and relate it to the statistical power of a sample of embeddings in a network. We show experimentally that, in contrast to earlier overlap-graph based proposals, our support measure makes it feasible to mine subgraph patterns in large networks.

Keywords Graph mining · Frequency counting · Overlap graph · Linear program · Variance on sample estimates

Responsible editor: Tijl De Bie and Peter Flach.

Y. Wang (✉) · J. Ramon · T. Fannes
Department of Computer Science, KU Leuven, 3001 Heverlee, Leuven, Belgium
e-mail: yuyi.wang@cs.kuleuven.be

J. Ramon
e-mail: jan.ramon@cs.kuleuven.be

T. Fannes
e-mail: thomas.fannes@cs.kuleuven.be

1 Introduction

Graph mining is a subfield of structured data mining. An important task is *frequent subgraph pattern mining*, which concerns the problem of finding subgraph patterns that occur frequently in a collection of graphs or in a single large graph. In this paper, we consider the single-graph setting, and we will call the large graph containing all data the *database graph*. Referring to the many applications, such as social networks, the Internet, chemical and biological interaction networks, traffic networks and citation networks, the database graph is also often called the *network*.

In order to precisely define a frequent subgraph pattern mining problem, a *support measure* (also called a *frequency measure*) is needed. In the problem setting where subgraph patterns are mined in a set of transactions [e.g., itemset mining (Agrawal et al. 1993)], a simple support measure is to count the number of transactions in which the subgraph pattern occurs. However, in the context of a single large graph, the issue is less straightforward as several articles have demonstrated (Bringmann and Nijssen 2008; Calders et al. 2011; Fiedler and Borgelt 2007; Vanetik et al. 2002).

An important drawback of just using the number of occurrences of a subgraph pattern (either embeddings or images) as its support is that this support is not *anti-monotonic*, i.e., the support of a subgraph pattern may be larger than the support of one of its subpatterns. The anti-monotonicity of the support measure (or more generally interestingness measure) plays a very important role in the design of a subgraph pattern miner, as it makes it possible to prune the search space (Kuramochi and Karypis 2005). Nevertheless, anti-monotonicity is not sufficient. For example, a support measure that just returns a constant is anti-monotonic, but not informative. From a statistical point of view, the more independent examples are, the more valuable this set of examples. Calders et al. (2011) proposed using a situation in which occurrences of a subgraph pattern occur independently (i.e., they do not overlap according to some notion of overlap) as a reference. In particular, the notion of a normalized graph support measure was defined: a support measure is *normalized* if every subgraph pattern which only has non-overlapping occurrences in a database graph has a support in this database graph that equals the number of occurrences.

An important class of support measures relies on *overlap graphs*. In an overlap graph, the vertices represent occurrences of a given subgraph pattern, and two vertices are adjacent if and only if the corresponding occurrences overlap in the database graph (according to some notion of overlap, such as sharing a vertex or an edge). An overlap graph therefore indicates how often a subgraph pattern occurs in the database graph, and how independent these occurrences are. An overlap-graph based support measure (OGSM) takes an overlap graph of a subgraph pattern in a database graph as its input, and outputs the support of that subgraph pattern in that database graph. Vanetik et al. (2002) proposed the maximum independent set (MIS) measure, i.e. the size of the maximum independent set of the overlap graph. This is intuitively appealing since it measures how often we observe a subgraph pattern occurring independently. Unfortunately, computing the MIS of an overlap graph is NP-hard (Garey and Johnson 1979), even for bounded degree graphs. Moreover, it has been shown that MIS cannot be approximated even within a factor of $n^{1-o(1)}$, where n is the order of the overlap graph, in polynomial time, unless $P = NP$ (Feige et al. 1991). Calders et al. (2011)

proposed the Lovász theta function ϑ (see, e.g., Knuth 1994; Lovász 1979), which is computable in time polynomial in the order of the overlap graph using semidefinite programming (SDP). A straightforward application of a general purpose SDP solver yields a running time of $O(n^{6.5})$ (Klein and Lu 1996). An SDP primal-dual algorithm for approximating ϑ with a multiplicative error of $(1 + \epsilon)$ has been proposed (Chan et al. 2009), which has a running time of $O(\epsilon^{-2}n^5 \log n)$. Iyengar et al. (2011) considered subgradient methods for approximating ϑ , which run in time $O(\epsilon^{-2} \log^3(\epsilon^{-1})n^4 \log n)$ in the worst case. Unfortunately, even these approximation methods are still computationally too expensive for our purposes.

In this paper, we propose a new support measure $\mathbf{s}$ that is based on bounding the value of all occurrences of a subgraph pattern that share a particular part of the database graph, and we show that $\mathbf{s}$ can be computed efficiently using a linear program (LP). The measure $\mathbf{s}$ is not a traditional OGSM, because its output does not merely depend on the overlap graph. We introduce the notion of an *overlap hypergraph*, and represent $\mathbf{s}$ as an overlap-hypergraph based support measure (OHSM). We prove that $\mathbf{s}$ is anti-monotonic and normalized. Furthermore, we show that all normalized anti-monotonic OHSMs are bounded between two extreme support measures. Our empirical analysis shows that this approach yields the first support measure which is both overlap based and computationally feasible. This is appealing from a statistical point of view as we also show that the measure is related to the statistical power of a sample of embeddings of a pattern in a network.

This paper is an extended version of the ECML-PKDD 2012 paper (Wang and Ramon 2012). Next to providing more detail, in this paper we add several new contributions.

- We provide a statistical motivation for our work by proving that the $\mathbf{s}$ support measure of a pattern can be interpreted as a measure of the statistical power of that pattern under certain assumptions.
- We compare to other support measures, such as the *minImage* support measure. We also point out that the Schrijver graph measure can be used as a support.
- We provide a discussion of our implementation and of several optimization strategies. Amongst others, we discuss candidate generation in more detail and explain how we handle patterns with a huge number of embeddings.

The remainder of this paper is structured as follows. In the next section, we briefly review some basic notations from graph theory. Section 3 formalizes support measures, overlap graphs, and overlap hypergraphs. We review several existing overlap-graph based support measures. We also introduce a new overlap-graph based support measure which is polynomial time computable and closer to the MIS support comparing than the Lovász ϑ value. In Sect. 4, we discuss related work. In Sect. 5, we present the main contribution of this paper, the new support measure $\mathbf{s}$ and model it as an LP. We also prove that $\mathbf{s}$ is normalized and anti-monotonic. The properties of normalized anti-monotonic OHSMs, particularly the $\mathbf{s}$ support, are shown in Sect. 6. Section 6 also points out a phase transition phenomenon between frequent and infrequent subgraph patterns, analyzes the statistical power of a subgraph pattern and generalizes the definition of overlap that can be plugged in to the $\mathbf{s}$ support measure. Section 7 gives

possible optimizations to solve the LPs of the **S** support. Section 8 presents experimental results. Section 9 concludes the paper with an overview of our contributions.

2 Preliminaries

We review a number of basic graph theoretical notions used in this paper. For more background in this area, see also (Diestel 2010).

2.1 Graphs

Definition 1 (*Graph*) A graph G is an ordered pair (V, E) , where V is a set of vertices and E is either a set of edges $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$ or a set of arcs $E \subseteq \{(u, v) \mid u, v \in V, u \neq v\}$. In the former (latter) case, we call the graph *undirected* (*directed*).

Definition 2 (*Adjacency and incidence*) Vertices are *adjacent* if there is an edge (arc) between them. For an edge $e = \{u, v\}$ (arc $e = (u, v)$), u and v are *incident* with e .

Definition 3 (*Labeled graph*) A *labeled graph* is a quadruple $G = (V, E, \Sigma, \lambda)$, with (V, E) a graph, Σ a non-empty finite set of labels, and λ a function assigning labels in Σ to the vertices or edges (or arcs), or both.

For simplicity, we use the term ‘labeled graph’ to refer to *vertex-labeled graph* unless stated otherwise. We use the notation $V(G)$, $E(G)$ and λ_G to refer to the set of vertices, the set of edges (or arcs) and the labeling function of a graph G , respectively. We denote the class of all graphs by $\mathcal{G}$.

Definition 4 (*Subgraph*) g is said to be a *subgraph* of G , denoted as $g \subseteq G$, if $V(g) \subseteq V(G)$, $E(g) \subseteq E(G)$ and for all $v \in V(g)$, $\lambda_g(v) = \lambda_G(v)$.

Definition 5 (*Induced subgraph*) For an undirected graph $G = (V, E)$, an *induced subgraph* on vertices set $U \subseteq V$ is denoted as $G[U] = (U, \{e \in E \mid e \cap U = e\})$.

Definition 6 (*Complete graph*) A *complete graph* K_n is a graph on n vertices, and every pair of its vertices is adjacent.

Definition 7 (*Complement of a graph*) The *complement graph* of a graph $G = (V, E)$ is denoted as $\overline{G}$, and $\overline{G} = (V, V \times V - E - \{(v, v) \mid v \in V\})$. For example, the complement graph of the complete graph K_n is n isolated vertices, write $\overline{K_n}$.

Definition 8 (*Independent set*) An *independent set* I of $G \in \mathcal{G}$ is a subset of $V(G)$ such that no pair of distinct vertices of I is adjacent in G .

Definition 9 (*Clique*) A *clique* C of $G \in \mathcal{G}$ is a subset of $V(G)$ such that for all distinct vertices $u, v \in C$, u and v are adjacent in G .

Definition 10 (*Clique partition*) A *clique partition* $\mathcal{Q} = \{q_1, q_2, \dots, q_k\}$ of $G \in \mathcal{G}$ is a partition of $V(G)$ such that every set q_i in $\mathcal{Q}$ is a clique.

Fig. 1 Homomorphism and isomorphism. A homo-image (but not iso-image) of P is highlighted in D_1 , and an iso-image of P is highlighted in D_2

2.2 Morphisms

The following concepts defined in terms of undirected labeled graphs are also valid for directed and/or unlabeled graphs with slight modifications.

Definition 11 (*Homomorphism*) A homomorphism ψ from $G \in \mathcal{G}$ to $G' \in \mathcal{G}$ is a mapping from $V(G)$ to $V(G')$ such that for all $v \in V(G) : \lambda_G(v) = \lambda_{G'}(\psi(v))$ and for all $\{u, v\} \in E(G) : \{\psi(u), \psi(v)\} \in E(G')$.

We call ψ *vertex-surjective* if $\forall v' \in V(G') : \exists v \in V(G) : \psi(v) = v'$, and call it *edge-surjective* if $\forall \{u', v'\} \in E(G') : \exists \{u, v\} \in E(G) : \psi(u) = u'$ and $\psi(v) = v'$.

Definition 12 (*Surjective homomorphism*) A homomorphism is *surjective* if it is both vertex- and edge-surjective.

Definition 13 (*Bijective mapping*) A mapping ψ is called *bijective* if and only if it is surjective and for all u and v , if $v \neq u$ then $\psi(v) \neq \psi(u)$.

Definition 14 (*Isomorphism*) An *isomorphism* from $G \in \mathcal{G}$ to $G' \in \mathcal{G}$ is a bijective homomorphism ψ from G to G' . In this case, we say that G is *isomorphic* to G' and write $G \cong G'$.

Definition 15 (*Subgraph isomorphism*) There exists a *subgraph isomorphism* from G to G' if $G \cong g$ for some subgraph g of G' . We will use $G \preceq G'$ to denote that G is subgraph isomorphic to G' .

Definition 16 (*Image and embedding*) An *iso-image* (*homo-image*) g of $P \in \mathcal{G}$ in $D \in \mathcal{G}$ is a subgraph $g \subseteq D$ for which there exists an isomorphism (surjective homomorphism) ψ from P to g . The subgraph g is called the *iso-image* (*homo-image*) through ψ . An individual isomorphism (surjective homomorphism) ψ from P to g is called an *iso-embedding* (*homo-embedding*) of P in D .

Figure 1 shows examples of images and embeddings.

In this paper, we only consider iso-images, although the measure $\mathbf{s}$ can be generalized for other matching operators such as homomorphism. We subsequently use the term *image* instead of iso-image, and denote by $\text{Img}(D, P)$ the set of all images of P in D . We denote by $\text{Emb}(D, P)$ the set of all embeddings of P in D . Supposing that $g \in \text{Img}(D, p)$ and $g' \in \text{Img}(D, P)$, if g is a subgraph of g' , then we call g a *subimage* of g' and g' a *superimage* of g .

2.3 Hypergraphs

Definition 17 (*Hypergraph*) A *hypergraph* is an ordered pair (V, E) , where V is a set of vertices and E is a set of *hyperedges* $E \subseteq 2^V$.

We denote by $\mathcal{H}$ the class of all hypergraphs. As in the case of graphs, for $H \in \mathcal{H}$, $V(H)$ denotes the set of vertices and $E(H)$ denotes the set of hyperedges. To every hypergraph H which has n vertices and m hyperedges, we associate an $n \times m$ *incidence matrix* $M_H = (m_{ij})$ where $m_{ij} = 1$ if $v_i \in e_j$ and $m_{ij} = 0$ otherwise.

Definition 18 (*Induced subhypergraph*) For a hypergraph $H = (V, E)$, a *induced subhypergraph* on the vertices set $U \subseteq V$ is denoted as $H[U] = (U, \{e \in E | e \cap U = e\})$.

3 Support measures

In this section, we review the concepts and properties of support measures and overlap graphs, and present a new overlap-graph based support measure which is similar to the Lovász ϑ value but closer to the MIS support measure. As this new measure is mainly of theoretical interest, it is not further explored in this paper. In Sect. 5 we introduce the $\mathbf{S}$ measure and show that it can be computed using algorithms with much lower asymptotic complexity.

3.1 Support

We first review a number of concepts related to the support of a pattern.

Definition 19 (*Support*) A *support measure* is a function $f : \mathcal{G} \times \mathcal{G} \mapsto \mathbb{R}$ that maps pairs (D, P) to a non-negative number $f(D, P)$, where P is called the subgraph pattern, D the database graph and $f(D, P)$ the *support* of P in D .

For efficiency reasons, most graph miners generate subgraph patterns from smaller subgraph patterns to larger ones (Chakrabarti and Faloutsos 2006), exploiting the anti-monotonicity property of their support measure to prune the search space.

Definition 20 (*Anti-monotonicity*) A support measure f is *anti-monotonic* if for all p, P, D in $\mathcal{G} : p \subseteq P \Rightarrow f(D, P) \leq f(D, p)$.

As explained in the introduction, anti-monotonicity is not a sufficient condition for an informative support measure. The support measure should also be able to account for the independence of the occurrences of the subgraph patterns. *Overlap* can be defined in different ways [see Calders et al. (2011) and later in this paper Sect. 6.3]. Popular definitions are, for instance as *vertex-overlap*, i.e., two images g_1 and g_2 overlap if $V(g_1) \cap V(g_2) \neq \emptyset$, and as *edge-overlap*, i.e., two images g_1 and g_2 overlap if $E(g_1) \cap E(g_2) \neq \emptyset$. Edge-overlap implies vertex-overlap. In this paper, we use 'overlap' to mean vertex-overlap, although our results are also applicable in the edge-overlap setting.

While one can argue about the value a support measure should have for patterns with overlapping embeddings, things are clearer when embeddings do not overlap. Hence, we will use non-overlapping cases as reference points.

Definition 21 (*Normalized support*) A support measure f is *normalized* if for all P, D in $\mathcal{G}$: $f(D, P) = |\text{Img}(D, P)|$ when there do not exist two distinct images g_1 and g_2 in $\text{Img}(D, P)$ satisfying $V(g_1) \cap V(g_2) \neq \emptyset$.

3.2 Overlap graphs

The notion of an overlap graph plays an important role in the design and computation of anti-monotonic support measures in networks.

Definition 22 (*Overlap graph*) Given a subgraph pattern P and a database graph D , the *overlap graph* of P in D is an unlabeled undirected graph G_P^D . Every vertex of G_P^D is an image of P in D , that is, $V(G_P^D) = \text{Img}(D, P)$. Two vertices u and v are adjacent in G_P^D if they overlap, i.e., if we use vertex-overlap, $V(u) \cap V(v) \neq \emptyset$.

Vanetik et al. (2006) define the induced support measure $f(D, P) = f'(G_P^D)$ where f' is a measure on overlap graphs. We call the induced support measure an overlap-graph based support measure (OGSM).

3.3 Existing overlap-graph based support measures

The first normalized anti-monotonic OGSM which has been proposed was the size of the MIS of the overlap graph (Vanetik et al. 2002).

Definition 23 (*MIS support*) Given a database graph D and a subgraph pattern P the *MIS support* of P in D is

$$\text{MIS}(G_P^D) = \max\{|I| \mid I \text{ is an independent set of } G_P^D\}.$$

Later, Calders et al. (2011) proposed two normalized anti-monotonic OGSMs, the size of a minimum clique partition (MCP) of the overlap graph and the Lovász theta value (ϑ) of the overlap graph.

Definition 24 (*MCP support*) Given a database graph D and a subgraph pattern P the *MCP support* of P in D is

$$\text{MCP}(G_P^D) = \min\{|Q| \mid Q \text{ is a clique partition of } G_P^D\}.$$

The Lovász ϑ function is a well-known function sandwiched between MIS and MCP which can be computed in polynomial time. The concept of Lovász feasible matrix will be used in the definition of the Lovász ϑ support measure.

Definition 25 (*Lovász feasible matrix*) Given an overlap graph G_P^D , a *Lovász feasible matrix* A for G_P^D is a symmetric positive semidefinite matrix with (i) $A_{u,v} = 0$ for all u and v such that $\{u, v\} \in E(G_P^D)$ and (ii) $\text{Tr}(A) = 1$.

Definition 26 (Lovász ϑ support measure) Given a database graph D and a subgraph pattern P the Lovász ϑ support measure of P in D is

$$\vartheta(G_P^D) = \max_A \left\{ \sum_{i,j} A_{i,j} \mid A \text{ is a Lovász feasible matrix of } G_P^D \right\}.$$

As mentioned in the introduction, these existing OGSMs are very expensive to compute. In particular, MIS and MCP are NP-hard to compute while the Lovász ϑ value is the solution of a SDP which takes $O(|V(G_P^D)|^{6.5})$ to solve for a general graph.

3.4 Operations on overlap graphs

Vanetik et al. (2006) provide a way to prove the anti-monotonicity of OGSMs. The result is based on three operations defined on overlap graphs.

- Vertex addition (VA): for a new vertex v , $VA(G, v) = (V \cup \{v\}, E \cup \{\{v, u\} \mid u \in E\})$.
- Edge removal (ER): for a given edge $e \in E$, $ER(G, e) = (V, E - \{e\})$.
- Clique contraction (CC): for a given clique $K \subseteq V$ and a new vertex k , $CC(G, K, k) = (V - K \cup \{k\}, E - \{e \mid e \cap K \neq \emptyset\} \cup \{\{k, v\} \mid \forall u \in K : \{v, u\} \in E\})$.

Theorem 1 (Vanetik et al. 2006) *An overlap-graph based support is anti-monotonic if and only if the corresponding overlap graph measure does not decrease when we perform VA, ER and CC on overlap graphs.*

The anti-monotonicity of OGSMs mentioned in Sect. 3.3 can be proved by this theorem. We generalize these results to overlap hypergraphs (see Sect. 5).

3.5 The Schrijver graph measure as a support measure

The material in this section is not essential for the sequel of the paper and can be easily skipped by readers eager to get to the main contribution.

The first function which was shown to be a normalized anti-monotonic OGSM computable in polynomial time was the Lovász ϑ value of the overlap graph. In the graph theory literature, many other measures on graphs are studied which could be of interest from the point of view of support measures. Often, the literature also shows relations to the size of the maximum independent set and other important measures. We believe that it may be valuable for the data mining community to further explore this literature. As an example, we point out that the Schrijver graph measure (Schrijver 1979), which is similar to the Lovász ϑ value, can also be interpreted as a normalized anti-monotonic OGSM. The Schrijver graph measure has nearly the same computational complexity as the Lovász ϑ value, and it is closer to the MIS support measure. The latter can be an advantage for certain statistical tasks in which we want to stay as close as possible to an independent set of images.

The Schrijver graph measure is defined on a Lovász feasible matrix with nonnegative elements.

Definition 27 (*Schrijver feasible matrix*) Given an overlap graph G_P^D , a *Schrijver feasible matrix* A for G_P^D is a Lovász feasible matrix which is nonnegative, i.e. for all i and j , $A_{i,j} \geq 0$.

Definition 28 (*Schrijver graph support measure*) Given a database graph D and a subgraph pattern P the *Schrijver graph measure (SGM)* support of P in D is

$$SGM(G_P^D) = \max_A \left\{ \sum_{i,j} A_{i,j} \mid A \text{ is any Schrijver feasible matrix of } G_P^D \right\}.$$

From the definition, we can easily see that for any overlap graph G_P^D , it holds that the $SGM(G_P^D) \leq \vartheta(G_P^D)$ since any Schrijver feasible matrix is a Lovász feasible matrix. In addition, the Schrijver graph measure can also be modeled as an SDP which has similar variables and constraints as the Lovász ϑ value, so it is almost equally expensive to compute the Schrijver graph measure of an overlap graph as to compute its Lovász ϑ value.

Now, we prove the following result using Theorem 1:

Theorem 2 *The SGM support measure is a normalized anti-monotonic OGSM.*

Proof First, we prove SGM is normalized. This is equivalent to proving that $SGM(\overline{K}_n) = n$. As we know, $SGM(\overline{K}_n) \leq \vartheta(\overline{K}_n) = n$. The matrix A with every element $1/n$ is a Schrijver feasible matrix of $\overline{K}_n$. Then, $SGM(\overline{K}_n) = n$.

Next, we prove that SGM is anti-monotonic using the conditions Theorem 1.

Let $G' = VA(G, v)$, and A be the optimal Schrijver feasible matrix of G . The matrix $A' = \begin{bmatrix} A & 0 \\ 0 & 0 \end{bmatrix}$ is a Schrijver feasible matrix of G' . Then, $SGM(G') \geq \sum_{i,j} A'_{i,j} = \sum_{i,j} A_{i,j} = SGM(G)$.

Let $G' = ER(G, e)$, and A be the optimal Schrijver feasible matrix of G . The matrix $A' = A$ is a Schrijver feasible matrix of G' . Then, $SGM(G') \geq \sum_{i,j} A'_{i,j} = \sum_{i,j} A_{i,j} = SGM(G)$.

Let $G' = CC(G, K, k)$, and A be the optimal Schrijver feasible matrix of G . Without loss of generality, we assume that the vertices in K are the last vertices (i.e. they correspond to the bottom rows and rightmost columns). Suppose $|V(G)| = n$, for any real vector $x = [x_1, \dots, x_n]$, $x^T Ax \geq 0$ because A is positive semidefinite. Let A' be a symmetric matrix of size $(n - |K| + 1) \times (n - |K| + 1)$ that $A'_{i,j} = A_{i,j}$ for every $1 \leq i, j \leq n - |K|$, $A'_{n-|K|+1,j} = \sum_{n-|K|+1 \leq i \leq n} A_{i,j}$ for every $1 \leq j \leq n - |K|$, $A'_{i,n-|K|+1} = \sum_{n-|K|+1 \leq j \leq n} A_{i,j}$ for every $1 \leq i \leq n - |K|$, and $A'_{n-|K|+1,n-|K|+1} = \sum_{n-|K|+1 \leq i \leq n} A_{i,i}$. For every real vector $y = [y_1, \dots, y_{n-|K|+1}]$, there is a vector x , $x_i = y_i$ for $1 \leq i \leq n - |K|$ and $x_i = y_{n-|K|+1}$ for $n-|K| \leq i \leq n$, such that $y^T A' y = x^T Ax \geq 0$. Therefore, A' is a Schrijver feasible matrix of G' and $SGM(G') \geq \sum_{i,j} A'_{i,j} = \sum_{i,j} A_{i,j} = SGM(G)$. $\square$

Fig. 2 Database graph D contains two independent images of the subgraph pattern P . However, $\minImage(D, P) = 4$ (and we can make this value arbitrarily large by adding more vertices with label b (resp. a) and link them to the *top-left* vertex with label a (resp. *bottom-right* vertex with label b). As a consequence, if we remove just a single vertex (the *top-left* or *bottom-right* one) the support of the pattern in the network can suddenly drop to one

The bounding theorem in [Calders et al. \(2011\)](#) tells us that given an overlap graph G_P^D ,

$$MIS(G_P^D) \leq SGM(G_P^D) \leq \vartheta(G_P^D) \leq MCP(G_P^D).$$

4 Related work

Besides the existing overlap-graph based support measures described in the previous section, several previous studies have explored the support of a subgraph pattern in a database graph. In this section, we review a widely used support measure called the min-image based measure as well as another definition of overlap, which ignores those overlaps that do not harm the anti-monotonicity of the MIS support. The definitions given in this section are not needed later in this paper.

4.1 Min-image based support

In [Bringmann and Nijssen \(2008\)](#), the authors proposed an anti-monotonic support measure named *min-image based support*.

$$\minImage(D, P) = \min_{v \in V(P)} |\{\psi_i(v) \mid \psi_i \text{ is a subgraph isomorphism from } P \text{ to } D\}| \quad (1)$$

This support counts the minimum number of images of subgraph pattern vertices. Although its anti-monotonicity of this support is obvious, and it can be computed very efficiently, it has several drawbacks.

First, from a statistical point of view, *minImage* overestimates the evidence. In particular, as Fig. 2 shows, a vertex can be counted arbitrarily many times.

Fig. 3 A database graph D has two connected components D_1 and D_2 . $\minImage(D_1, P) = \minImage(D_2, P) = 1$, but $\minImage(D, P) = 3 > \minImage(D_1, P) + \minImage(D_2, P)$. The min-image based support does not explain this

Second, $\minImage$ is not additive. Given a subgraph pattern P , if a database graph D has n ($n \geq 2$) connected components, i.e., $D = \bigcup_{1 \leq i \leq n} D_i$, then $\minImage(D, P) \geq \sum_{1 \leq i \leq n} \minImage(D_i, P)$. For many realistic database graphs strict inequality holds. In this case, it is unclear how much a connected component contributes to the whole support. Figure 3 shows an example.

4.2 Harmful overlap

Fiedler and Borgelt (2007) observed that when using edge-overlap some of the overlaps can be disregarded without harming the anti-monotonicity of the MIS support measure. The authors introduce the notion of *harmful overlap support* which relies on the nonexistence of equivalent ancestor embeddings.

Definition 29 (*Harmful overlap*) Given a database graph D and a subgraph pattern P , there is harmful overlap between two embeddings ψ_1 and ψ_2 of P in D if $\exists v \in V(P) : \psi_1(v), \psi_2(v) \in V(g_1) \cap V(g_2)$, where g_1 (resp. g_2) is the image of P in D through ψ_1 (resp. ψ_2).

The notion of harmful overlap makes sense, especially in situations where objects (vertices) in the pattern play different roles. Hence if one vertex participates in two embeddings in a different role, this is not considered as a dependency between the two embeddings.

5 The $\mathbf{s}$ support measure

In this section we introduce our new support measure $\mathbf{s}$. We start by introducing overlap hypergraphs, which are overlap graphs carrying additional information on the cause of the overlap.

5.1 Overlap hypergraphs

For simplicity of explanation, we will only consider vertex-overlap. As we are using vertex-overlap, each vertex v in a database graph D determines a clique in the overlap graph G_P^D in which P is a subgraph pattern. That is, suppose v is a vertex in D , then

Fig. 4 Overlap graph and overlap hypergraph. Given a subgraph pattern P , a database graph D , the overlap graph G_P^D and the overlap hypergraph H_P^D are shown on the right. In the overlap hypergraph, the (dominating) hyperedges are determined by the highlighted vertices in the database graph, and a dominated hyperedge is given in a dashed ellipse

$\text{Img}_v(D, P) = \{g \in \text{Img}(D, P) \mid v \in V(g)\}$ induces a clique in G_P^D since the images overlap at the vertex v .

Definition 30 (*Overlap hypergraph*) The *overlap hypergraph* of P in D , denoted by H_P^D is a hypergraph whose vertices are the images $\text{Img}(D, P)$, and for each vertex $v \in V(D)$ there is a hyperedge $e_v \in E(H_P^D)$ such that $e_v = \{g \in V(H_P^D) \mid v \in V(g)\}$. The hyperedges represent cliques in G_P^D .

In an overlap hypergraph H_P^D , we say that a hyperedge e is *dominated* by another hyperedge e' if $e \subset e'$, and a hyperedge e is *dominating* if it is not dominated by any other hyperedge. For any D and P , we define the reduced overlap hypergraph $\tilde{H}_P^D$ to be the hypergraph for which $V(\tilde{H}_P^D) = V(H_P^D)$ and $E(\tilde{H}_P^D)$ is the set of all dominating hyperedges of H_P^D . In the sequel we only refer to $\tilde{H}_P^D$. We will abuse terminology and simply call $\tilde{H}_P^D$ the overlap hypergraph. See Fig. 4 for an example.

We henceforth refer to the overlap hypergraph measures, which we denote by $f'(\tilde{H}_P^D)$, instead of referring to the induced support measure $f(D, P)$. Such induced support measures are called overlap-OHSM. We call OHSMs and OGSMs overlap based support measures.

5.2 Origin and definition

Given an overlap hypergraph $\tilde{H}_P^D$, we can derive the corresponding overlap graph G_P^D by replacing every hyperedge with a clique. Therefore, we can rephrase the definition of the MIS measure using overlap hypergraphs. Suppose D is a database graph and P is a subgraph pattern:

$$\text{MIS}(D, P) = \text{MIS}(\tilde{H}_P^D) = \max |\{I \subseteq V(\tilde{H}_P^D) \mid \forall e \in E(\tilde{H}_P^D) : |e \cap I| \leq 1\}| \quad (2)$$

The MIS measure requires that a vertex of an overlap (hyper)graph is either in the independent set I or not. Our new measure $\mathbf{s}$ is a relaxation of the MIS support measure by allowing for counting vertices of an overlap hypergraph only partially.

Let $\tilde{H}_P^D$ be an overlap hypergraph. We start by assigning to each vertex v of $\tilde{H}_P^D$ a variable x_v . We then consider vectors $x \in \mathbb{R}^{V(\tilde{H}_P^D)}$ of variables where for every $v \in V(\tilde{H}_P^D)$, x_v denotes the variable (component of x) corresponding to v . x is *feasible* if and only if it satisfies

$$\begin{aligned} \text{(i)} \quad & \forall v \in V(\tilde{H}_P^D) : 0 \leq x_v \\ \text{(ii)} \quad & \forall e \in E(\tilde{H}_P^D) : \sum_{v \in e} x_v \leq 1. \end{aligned} \quad (3)$$

We denote the feasible region (the set of all feasible $x \in \mathbb{R}^{V(\tilde{H}_P^D)}$) by $\mathfrak{R}(\tilde{H}_P^D)$, which is a convex polytope.

Definition 31 (*s support measure*) The measure $\mathbf{s}$ is defined by

$$\mathbf{s}(\tilde{H}_P^D) = \max_{x \in \mathfrak{R}(\tilde{H}_P^D)} \sum_{v \in V(\tilde{H}_P^D)} x_v \quad (4)$$

Clearly, $\mathbf{s}$ is the solution to a LP.

We will call an element $x \in \mathfrak{R}(\tilde{H}_P^D)$ which makes $\sum_{v \in V(\tilde{H}_P^D)} x_v$ maximal a *solution* to the LP of $\mathbf{s}$.

There are very effective methods for solving LPs, including the simplex method which is efficient in practice although its complexity is exponential, and the more recent interior-point methods (Boyd and Vandenberghe 2004). The interior-point method solves an LP in $O(n^2m)$ time, where n (here $\min\{|V(\tilde{H}_P^D)|, |E(\tilde{H}_P^D)|\}$) is the number of variables, and m (here $|V(\tilde{H}_P^D)| + |E(\tilde{H}_P^D)|$) is the number of constraints. Usually, subgraph patterns are not large, so the LPs for computing $\mathbf{s}$ are sparse. Almost all LP solvers perform significantly better for sparse LPs.

5.3 An intuitive interpretation

So far, the $\mathbf{s}$ support has been explained as a relaxation of the MIS support. In more detail, the basic ideas underlying the $\mathbf{s}$ measure are the following:

1. x_v is the contribution of the image v , and $x_v \geq 0$ for all v By finding an additional image (corresponding to a ‘Vertex Addition’ in the overlap (hyper)graph), it should positively contribute to the total support.
2. The $\mathbf{s}$ support measure is the sum of contributions of all images This idea addresses the question of how to use the contributions to define the total support. We can simply sum the contributions of all images, which is intuitive and is one of the main factors in the nice mathematical properties of $\mathbf{s}$ including its additivity (the $\mathbf{s}$ of a graph is the sum of the $\mathbf{s}$ of its connected components).
3. Every image should contribute as much as possible (but not more than 1) This principle implies that $\mathbf{s}$ is obtained through a process of maximization. In fact, in this maximization process we identify a set of images of the pattern which is most interesting/informative.

4. All images which share a common vertex cannot contribute more than 1 in total, i.e., the sum of contributions of images in a hyperedge ≤ 1 If only the first three principles applied and if no additional constraints were imposed on the contributions of the individual images, this would yield a trivial support which just counts the number of images. As pointed out earlier, this support is not anti-monotonic because overlapping images are counted independently. When discussing the *minImage* support measure, we also pointed out that, ideally, any vertex in a database graph cannot contribute more than 1 to the support measure. This is equivalent to saying that, all images sharing a common vertex cannot jointly contribute more than 1 in total.

The reasons listed above provide a motivation for using the **S** support measure and its underlying principles, and allow us to derive the LP for **S**.

In Sect. 6.2 we provide further statistical reasons, showing in particular that the **S** measure corresponds to the statistical power of the embeddings of the pattern under certain assumptions. The **S** measure therefore has several uses. It is statistically meaningful in its own right, and as a relaxation of the maximal independent set, it can give a quick approximation of the MIS value. Still, if overlap needs to be avoid at all cost [such as in Dreweke et al. (2007)], the maximum independent set size will need to be computed or approximated from below if that is an acceptable alternative.

5.4 Conditions for anti-monotonicity

Vanetik et al. (2006) described necessary and sufficient conditions for anti-monotonicity of OGSMs on labeled graph using edge-overlap. In Calders et al. (2011), this result was generalized to any OGSM on labeled or unlabeled, directed or undirected graphs using edge overlap or vertex overlap and isomorphism, homomorphism or homeomorphism. Our conditions for anti-monotonicity are similar but based on overlap hypergraphs. In particular, we show that an OHSM is anti-monotonic if and only if it is non-decreasing under three operations on the overlap hypergraph.

We begin by defining these three operations on any overlap hypergraph, which we will then use in our conditions for anti-monotonicity. These operations are different from those used in Calders et al. (2011), Vanetik et al. (2006), but play a similar role. As mentioned in these earlier papers, the motivation for the operations defined below is that it is often easier to show that an OHSM satisfies the conditions of the theorem (being non-decreasing under the three operations), than to directly demonstrate the anti-monotonicity of a measure.

Definition 32 (*Hypergraph operators*) For $H \in \mathcal{H}$, we define:

- **VA**: A new vertex v is added to every existing hyperedge: $VA(H, v) = (V(H) \cup \{v\}, \{e \cup \{v\} \mid e \in E(H)\})$.
- **Subset Contraction**: Let $K \subseteq V(H)$ be a set of vertices of the hypergraph such that $\exists e \in E(H) : K \subseteq e$. Then, the subset contraction operation contracts K into a single vertex k , which remains in only those hyperedges that are supersets of K . Formally, $SC(H, K, k) = (V(H) - K \cup \{k\}, E_1 \cup E_2)$ where $E_1 = \{e - K \cup \{k\} \mid e \in E(H) \text{ and } K \subseteq e\}$ and $E_2 = \{e - K \mid e \in E(H) \text{ and } K \not\subseteq e\}$.

- Hyperedge Split: This operation splits a size k hyperedge into k hyperedges of size $(k - 1)$ each: $HS(H, e) = (V(H), E(H) - \{e\} \cup \{e - \{v\} \mid v \in e\})$, where $e \in E(H)$.

For example, suppose H_0 is a hypergraph, $V(H_0) = \{v_1, v_2, v_3, v_4\}$, and $E(H_0)$ contains two hyperedges $\{v_1, v_2, v_3\}$ and $\{v_1, v_4\}$. Let $H_1 = VA(H_0, v_5)$, then $V(H_1) = \{v_1, v_2, v_3, v_4, v_5\}$ and $E(H_1)$ contains hyperedges $\{v_1, v_2, v_3, v_5\}$ and $\{v_1, v_4, v_5\}$. Let $H_2 = SC(H_1, \{v_1, v_3\}, v_6)$, then $V(H_2) = \{v_2, v_4, v_5, v_6\}$ and $E(H_2)$ contains hyperedges $\{v_2, v_5, v_6\}$ and $\{v_4, v_5\}$. Let $H_3 = HS(H_2, \{v_2, v_5, v_6\})$, then $V(H_3) = V(H_2)$ and $E(H_3)$ contains four hyperedges $\{v_2, v_5\}$, $\{v_2, v_6\}$, $\{v_5, v_6\}$ and $\{v_4, v_5\}$.

5.4.1 Sufficient condition

We present a sufficient condition for support measure anti-monotonicity in terms of the three operations on the overlap hypergraph that we have defined.

Theorem 3 *Let $f' : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ be a support measure, and $f : \mathcal{H} \rightarrow \mathbb{R}$ with $f'(D, P) = f(\tilde{H}_P^D)$ be the induced OHSM. If f is non-decreasing under VA, SC and HS, then f' is an anti-monotonic support measure.*

Proof Suppose D is a database graph, and p and P are two subgraph patterns such that p is a subgraph of P . We prove that $\tilde{H}_p^D$ can be obtained from $\tilde{H}_P^D$ by applying only the operations VA, SC and HS. It follows then that $f'(D, P) = f(\tilde{H}_P^D) \leq f(\tilde{H}_p^D) = f'(D, p)$ for any D, P and p , proving the theorem.

Let $<$ be an arbitrary order defined on $V(\tilde{H}_P^D)$. For $v \in V(\tilde{H}_P^D)$, we define the set $\Pi_v = \{u \in V(\tilde{H}_P^D) \mid v \leq u \text{ and } \forall w < v : w \not\leq u\}$. Remember that the vertices of $\tilde{H}_p^D$ are images of p and hence $v \leq u$ refers to a subgraph isomorphism relationship between v and u .

The Π_v are pairwise disjoint and $\cup_{v \in V(\tilde{H}_P^D)} \Pi_v = V(\tilde{H}_P^D)$. We point out that there may exist vertices v for which $\Pi_v = \emptyset$. We divide $V(\tilde{H}_P^D)$ into two sets $V_0 = \{v \mid \Pi_v = \emptyset\}$ and $V_1 = \{v \mid \Pi_v \neq \emptyset\}$.

Let H be a hypergraph initially equal to $\tilde{H}_P^D$. We will perform operations VA, SC and HS on H , until it is finally equal to $\tilde{H}_p^D$.

First, H is modified by a sequence of VA operations. For each $v \in V_0$, we do $H := VA(H, v)$. Now, $\forall e \in E : V_0 \subseteq e$.

Then, for each $v \in V_1$, we perform $H := SC(H, \Pi_v, v)$. The operations are valid because for $v \in V_1$ each vertex $u \in \Pi_v$ stands for a superimage of the same v , i.e., $v \leq u$ and hence $\exists e \in E(H) : \Pi_v \subseteq e$. It is easy to verify that now $V(\tilde{H}_P^D) = V(H)$ holds.

Consider a hyperedge $e'_x \in E(\tilde{H}_p^D)$ which is determined by $x \in V(D)$, i.e., $e'_x = \{v \in V(\tilde{H}_p^D) \mid x \in V(v)\}$. We know that $e'_x \cap V_0$ is a subset of any $e \in E(H)$. $E(\tilde{H}_P^D)$ has a dominating hyperedge e''_x determined by x , i.e., $e''_x = \{v \in V(\tilde{H}_P^D) \mid x \in V(v)\}$ (or has another hyperedge e''_y which is a superset of the dominated hyperedge e''_x). We have $e'_x \subseteq e''_x$ (or e''_y). Thus, $\forall v \in e'_x \cap V_1 : \Pi_v \subseteq e''_x$ (or e''_y). Therefore,

there must be a hyperedge $e \in E(H)$ such that $e'_x \subseteq e$. This property shows that every hyperedge in $E(\tilde{H}_p^D)$ either exists in $E(H)$ or can be obtained later by performing a sequence of HS on H . $\square$

Theorem 4 $\mathbf{s}(D, P) = \mathbf{s}(\tilde{H}_p^D)$ is a normalized anti-monotonic support measure.

Proof First, we prove that $\mathbf{s}$ is normalized. If the subgraph pattern P only has non-overlapping images in the database graph D , every hyperedge in $E(\tilde{H}_p^D)$ contains only one vertex, then setting $x_v = 1$ for every $v \in V(\tilde{H}_p^D)$ is a feasible assignment and is clearly maximal. That is, $\mathbf{s}$ equals the number of non-overlapping images. Therefore, $\mathbf{s}$ is normalized.

Then, we prove $\mathbf{s}$ is anti-monotonic using Theorem 3. Suppose H is an overlap hypergraph and x^* is a solution to the LP of $\mathbf{s}(H)$. Let H_1 be the overlap hypergraph $VA(H, v)$, and let $x_u = x_u^*$ for all vertices $u \neq v$ and $x_v = 0$. x is a feasible solution for the LP of $\mathbf{s}(H_1)$, so $\mathbf{s}(H_1) \geq \sum_v x_v = \mathbf{s}(H)$. Let H_2 be the overlap hypergraph $SC(H, K, k)$, and let $x_u = x_u^*$ for all vertices $u \neq k$ and $x_k = \sum_{v \in K} x_v^*$. x is a feasible solution for the LP of $\mathbf{s}(H_2)$, so $\mathbf{s}(H_2) \geq \sum_v x_v = \mathbf{s}(H)$. Let H_3 be the overlap hypergraph $HS(H, e)$. x^* is also a feasible for the LP of $\mathbf{s}(H_3)$, so $\mathbf{s}(H_3) \geq \mathbf{s}(H)$. $\square$

5.4.2 Necessary condition

We show that the condition for anti-monotonicity mentioned above is not only a sufficient but also a necessary condition.

Theorem 5 Let $f' : \mathcal{G} \times \mathcal{G} \rightarrow \mathbb{R}$ be a support measure, and $f : \mathcal{H} \rightarrow \mathbb{R}$ with $f'(D, P) = f(\tilde{H}_p^D)$ be the induced OHSM. If f' is anti-monotonic, then f is non-decreasing under VA, SC and HS.

Proof Let H_p be any hypergraph and H_p a hypergraph obtained by performing VA, SC or HS on H_p . We show that there exists a database graph D and subgraph patterns P and p such that $\tilde{H}_p^D = H_p$ and $\tilde{H}_p^D = H_p$ (it is not harmful to assume that H_p and H_p only contain dominating hyperedges). It follows then that $f(H_p) = f'(D, P) \leq f'(D, p) = f(H_p)$, which proves the theorem. For convenience, we only show the theorem for undirected labeled graphs, but the proof can be generalized.

In Fig. 5, we give two subgraph patterns P and p ($p \leq P$), and list 4 different possible types of overlap. The numbers of vertices with label a (called a -vertex) and b (called b -vertex) in P and p are not fixed, but the pattern P always has 2 vertices labeled c (called c -vertex) while the pattern p always has only one c -vertex. We construct the database graph D by combining multiple copies of the patterns P and p , overlapping in different ways. We name the different types of overlap O_1 , O_2 , O_3 and O_4 . We say two or more copies of the pattern P have overlap type O_1 if they share a common b -vertex. Two or more copies of P and a copy of pattern p have overlap type O_2 if they share a b -vertex. Two or more copies of the pattern P have overlap type O_3 if they share a subgraph isomorphic to p (i.e. the bottom part of P in the figure). Two or more copies of the pattern P have overlap type O_4 if they share an a -vertex. As in our construction of the database graph D we do not use other types of overlap (e.g. no two copies of P share two c -vertices), every of the possibly overlapping copies of

which satisfies $e \cap K = \emptyset$, we let the P -images corresponding to vertices participating in e share a b -vertex (i.e. their overlap type is O_1). These O_1 overlaps imply that e appears in $E(\tilde{H}_p^D)$ when $e \cap K = \emptyset$. For every hyperedge $e \in E(\tilde{H}_p^D)$ which satisfies $K \subseteq e$, we let the P -images corresponding to vertices participating in $e \setminus K$ and the p -image which corresponds to k share a b -vertex (i.e. their overlap type is O_2). These O_2 overlaps imply that $e \setminus K \cup \{k\}$ appears in $E(\tilde{H}_p^D)$ when $K \subseteq e$. This construction satisfies $\tilde{H}_p^D = SC(\tilde{H}_p^D, K, k)$.

Finally, we consider the case $\tilde{H}_p^D = HS(\tilde{H}_p^D, e)$. We let the pattern P have $|E(\tilde{H}_p^D)|$ b -vertices, and one a -vertex. We let the pattern p also have $|E(\tilde{H}_p^D)|$ b -vertices. The database graph D consists $|V(\tilde{H}_p^D)|$ P -images, each corresponding to a vertex of $\tilde{H}_p^D$, with the overlaps among images determined by $\tilde{H}_p^D$ as follows. We let the P -images corresponding to vertices participating in e share their a -vertex (i.e. their overlap type is O_4), and for every $v \in e$ we let the P -images corresponding to vertices participating in $e \setminus \{v\}$ share a b -vertex (i.e. their overlap type is O_1). These overlaps make sure that e appears in $E(\tilde{H}_p^D)$ and, for all $v \in e$, $e \setminus \{v\}$ appears in $E(\tilde{H}_p^D)$. For every other hyperedge e' in $E(\tilde{H}_p^D)$, we let the P -images corresponding to vertices participating in e' share the b -vertex which determines e' (i.e. their overlap type is O_1). These O_1 overlaps imply that e' also appears in $E(\tilde{H}_p^D)$. This construction satisfies $\tilde{H}_p^D = HS(\tilde{H}_p^D, e)$. $\square$

5.5 Bounding theorem

In [Calders et al. \(2011\)](#), the authors showed that all normalized anti-monotonic OGSMs are bounded (between the MIS size and the MCP size). Similarly, we prove that all normalized anti-monotonic OHSMs are also bounded. We first introduce another OHSM, MSC, the size of a minimum set cover of overlap hypergraphs:

$$MSC(D, P) = MSC(\tilde{H}_p^D) = \min \{ |S \subseteq E(\tilde{H}_p^D) \mid \bigcup_{e \in S} e = V(\tilde{H}_p^D) | \} \quad (5)$$

It is not difficult to verify that MSC is normalized and anti-monotonic. Computing MSC is an NP-hard problem. The maximum independent set size (Eq. (2)) and minimum vertex cover (Eq. 5) are the minimally and maximally possible normalized anti-monotonic OHSMs.

Theorem 6 *Given a database graph D , and a subgraph pattern P , it holds that $MIS(D, P) \leq f(D, P) \leq MSC(D, P)$ for every normalized anti-monotonic OHSM $f(D, P) = f'(\tilde{H}_p^D)$.*

Proof We use Theorem 5 to show the minimality of MIS and the maximality of MCP.

On the one hand, let $I = \{v_1, v_2, \dots, v_k\}$ be a maximum independent set of $\tilde{H}_p^D$. Starting from the hypergraph $H_I = (\{v_1, v_2, \dots, v_k\}, \{\{v_1\}, \{v_2\}, \dots, \{v_k\}\})$, we can get $\tilde{H}_p^D$ by adding vertices $V(H) - I$ using VA first and then splitting hyperedges by a sequence of HS. Since f is normalized, $f'(H_I) = k$. Because f is anti-monotonic

and therefore f' cannot decrease after each step, $f(D, P)$ is larger than or equal to $k = MIS(D, P)$.

On the other hand, let $\{e_1, e_2, \dots, e_k\}$ be a minimum set cover for $\tilde{H}_P^D$ and let $H_{sc} = SC(\dots SC(SC(H, e_1, v_{e_1}), e_2, v_{e_2}) \dots, e_k, v_{e_k})$. H_{sc} only has the hyperedges with exactly one vertex in each of them. Because f is anti-monotonic, f' does not decrease under SC and thus $f(D, P) \leq f'(SC(H, e_1)) \leq \dots \leq f'(H_{sc}) = MSC(D, P)$. $\square$

5.6 Relaxation of the OGSM MIS

One may ask whether the $\mathbf{S}$ support can be defined by relaxing the OGSM MIS instead of the OHSM MIS. In other words, is the concept of overlap hypergraphs really necessary?

Our answer is that the concept of overlap hypergraph is needed for the definition of the $\mathbf{S}$ support measure because it carries additional information on the overlap graph. In particular the hyperedges show which overlaps have a common cause. If we did not have this information, we would not be able to reconstruct it. For instance, if we see a triangle in an overlap graph, we do not know whether this triangle originates from one vertex shared by the three images or from three vertices, each shared by two of the images. This additional information is needed for the definition of $\mathbf{S}$, and for its mathematical properties.

6 Properties of the $\mathbf{S}$ support measure and discussion

In this section, we will discuss several aspects of our approach, pointing out interesting properties and possible future extensions.

6.1 Phase transition from frequent to infrequent

Large real-world networks are known to satisfy properties similar to random graphs. A well-known result is that properties which can be expressed in first order logic are satisfied either by almost all graphs or by almost none [0-1 law, see [Fagin \(1976\)](#)]. For random graphs, it is either very easy to embed a given subgraph pattern P in the network, or very difficult (see also our experiments below). This leads to another 0-1 property: the frequency of many subgraph patterns is either very low or very high (for our $\mathbf{S}$ measure, nearly equal to the network size). Consider for instance a social network and the subgraph pattern “ X is a friend of Y and Y is a friend of Z ”. Since most people have at least two friends, such a subgraph pattern will match about everywhere. This holds more generally for many tree and path subgraph patterns. In fact, most such subgraph patterns are overly general and not very interesting.

6.2 Bounding the variance of sample estimates using the $\mathbf{S}$ measure

An important motivation for investigating support measures is the need to perform statistical analysis on datasets. Statistical theory often assumes that data points are

drawn independently. In networked data, however, where vertices are connected with edges, this is not the case anymore. In this section, we relate the statistical power of a set of observations to its s -measure. In particular, if a pattern has a number of overlapping embeddings in a database graph and every embedding has some properties, one can estimate the distribution of these properties (or its mean, variance, moment, ...) from a sample. We are interested in bounding the variance of such estimates.

When performing statistics on a particular type of observations, we first have to define the properties that the observations of interest will need to satisfy, thus creating a subgraph pattern. For instance, suppose we want to analyze the satisfaction of clients with their first lawsuit where they are assisted by a pro-deo lawyer, i.e. a lawyer paid by the government or by an association to offer legal aid services to those who cannot afford a lawyer. Then, the subgraph pattern representing the observation type of interest would consist of a client node, a lawyer node, a judge node and a lawsuit node to which the former three are connected.

Next, let us assume that the occurrence of these observations occur independently from the properties which are relevant for our statistical analysis. In our example, in order to ensure impartiality, the court randomly assigns judges to cases and the lawyer association randomly assigns pro-deo lawyers to cases. Hence, in order to explore the relationships between the properties of the case and its outcome, we do not need to take into account the dependency between occurrences of the subgraph pattern and its properties.

This simplifying assumption does not imply, however, that we can treat the properties of the nodes of the embeddings of the pattern as independent, since embeddings may share nodes. In our example, the same parties, lawyers or judges may participate in different lawsuits.

Consider the simple task of estimating the expected value of a function over the properties of nodes participating in a random embedding. In particular, let $f(\cdot)$ be a function on embeddings and let μ be its expected value and σ its standard deviation. Consider also a sample, i.e. a set of possibly overlapping embeddings, and the problem of estimating μ as accurately as possible. In our example, f could be the measurement of client satisfaction with the outcome of the lawsuit depending on properties of the client, the lawsuit, the lawyer or the judge.

We will now present two approaches for deriving a relation between sample size and the variance on the estimate obtained.

In a first approach, we take a maximal independent set S_{MIS} of vertices of the overlap hypergraph H_p^D . As we assumed that the embeddings are independent from the properties of the nodes they connect, all elements in S_{MIS} are independent and the values $f(v)$ of the observations v in S_{MIS} are distributed independently with $\mathbb{E}[(f(v) - \mu)^2] = \sigma^2$. Consider now the estimator

$$\hat{\mu}_{\text{MIS}} = \frac{\sum_{u \in S_{\text{MIS}}} f(u)}{|S_{\text{MIS}}|}.$$

As the terms in the sum are independent random variables,

$$\mathbb{E}[(\hat{\mu}_{\text{MIS}} - \mu)^2] = \frac{\sigma^2}{|S_{\text{MIS}}|}. \quad (6)$$

We will now present a second approach based on our $\mathbf{s}$ measure. Suppose that we have a set $V(H_p^D)$ of observations (embeddings of the pattern p in the database graph D), whose overlaps are given by the overlap hypergraph H_p^D , and a vector x of weights x_v for the $v \in V(H_p^D)$, which is a feasible solution to the $\mathbf{s}$ measure related LP (3). We define the estimator:

$$\hat{\mu}_{\mathbf{s}}(f, V(H_p^D), x) = \frac{\sum_{v \in V(H_p^D)} x_v f(v)}{\sum_{v \in V(H_p^D)} x_v} \quad (7)$$

We will now prove the following:

Theorem 7 *Let p be a pattern graph with $V(p) = \{i\}_{i=1}^k$ and $D = \cup_{i=1}^k D_i$ be a database with $k = |V(p)|$ domains. Let the set of embeddings of p in D be k -partite, i.e. $\text{Emb}(D, p) \subseteq D_1 \times \dots \times D_k$. Let the overlap hypergraph H_p^D represent this set of embeddings and their overlaps, i.e. two vertices $u, v \in V(H_p^D)$ overlap if and only if $u(i) = v(i)$ for some $i \in \{1 \dots k\}$. Assume the nodes in D have properties which are independent of these embeddings. Let x be a vector of weights for the embeddings satisfying (3). Let f be a function on the properties of the nodes participating in an embedding. Assume that for a randomly chosen embedding u , $\mathbb{E}[(f(u) - \mu)^2] = \sigma^2$. Then,*

$$\mathbb{E} \left[\left(\hat{\mu}_{\mathbf{s}}(f, V(H_p^D), x) - \mu \right)^2 \right] \leq \frac{\sigma^2}{\sum_{v \in V(H_p^D)} x_v}$$

Proof Without loss of generality, we assume $\mu = 0$ in this proof, then $x_u f(u) = x_u (f(u) - \mu)$. We first focus on the square of the sum of $x_u f(u)$.

$$\mathbb{E} \left[\left(\sum_{u \in V(H_p^D)} x_u f(u) \right)^2 \right] = \sum_{u, v \in V(H_p^D)} x_u x_v \mathbb{E}[f(u) f(v)] \quad (8)$$

Consider now a pair of embeddings u and v . For a (possibly empty) subset $R(u, v) \subseteq V(p)$ of vertices of the pattern p , u and v overlap, i.e. $R(u, v) = \{i \in V(p) \mid u(i) = v(i)\}$. On $V(p) \setminus R(u, v)$, u and v are different. As participants in embeddings are chosen randomly, $u(V(p) \setminus R(u, v))$ and $v(V(p) \setminus R(u, v))$ are independent. We now define

$$\sigma_S^2 = \mathbb{E}_{u, v: R(u, v) = S} [f(u(S), u(V(p) \setminus S)) f(v(S), v(V(p) \setminus S))], \quad (9)$$

i.e. σ_S^2 is the expected value of the product of the values of f of two embeddings which overlap in exactly S . Due to the independence of $u(V(p) \setminus S)$ and $v(V(p) \setminus S)$, we can write

$$\begin{aligned} \sigma_S^2 &= \mathbb{E}_{u(S)} [\mathbb{E}_{u(V(p) \setminus S)} [f(u(S), u(V(p) \setminus S))] \mathbb{E}_{v(V(p) \setminus S)} [f(v(S), v(V(p) \setminus S))]] \\ &= \mathbb{E}_{u(S)} [\mathbb{E}_{u(V(p) \setminus S)} [f(u(S), u(V(p) \setminus S))]^2] \end{aligned} \quad (10)$$

From this point of view, we can rewrite (8) as

$$\mathbb{E} \left[\left(\sum_{u \in V(H_p^D)} x_u f(u) \right)^2 \right] = \sum_{u, v \in V(H_p^D)} x_u x_v \sigma_{R(u, v)}^2 \quad (11)$$

We will now analyze the terms in the sum on the right hand side of this equation in greater detail. First, because of the properties of the LP defining $\mathbf{s}$, we know that for any i and $n \in D_i$,

$$\sum \{x_v \mid v \in V(H_p^D) \wedge v(i) = n\} \leq 1 \quad (12)$$

Second, if $\{Q_1, \dots, Q_m\}$ is a partition of $V(p)$, then we know from the properties of variance that

$$\sum_{i=1}^m \sigma_{Q_i}^2 \leq \sigma^2 \quad (13)$$

(in fact, σ^2 is the sum of the $\sigma_{Q_i}^2$ plus ‘interaction’ terms). Together (12) and (13), imply that for any u ,

$$\sum_{v \in V(H_p^D)} x_v \sigma_{R(u, v)}^2 \leq \sigma^2. \quad (14)$$

Substituting (14) into (11) gives

$$\mathbb{E} \left[\left(\sum_{u \in V(H_p^D)} x_u f(u) \right)^2 \right] \leq \sum_{u \in V(H_p^D)} x_u \sigma^2 \quad (15)$$

from which the theorem follows. $\square$

In conclusion, if we choose x such that $\sum_v x_v = \mathbf{s}(H_p^D)$, we get

$$\mathbb{E} \left[\left(\hat{\mu}_{\mathbf{s}}(f, V(H_p^D), x) - \mu \right)^2 \right] \leq \frac{\sigma^2}{\mathbf{s}(H_p^D)}.$$

Because $\mathbf{s} \geq |MIS|$, the second approach yields a better estimate of μ . Even though we had to make a number of assumptions, this first result linking $\mathbf{s}$ and the statistical power of a sample suggests that closer analysis of its properties may be a valuable direction for further research. Note that the assumptions on which the first method (using a maximum independent set) relies are not necessarily much weaker than those made for the method using $\mathbf{s}$.

6.3 The notion of overlap

Overlap can be defined in several ways, for instance as vertex-overlap or as edge-overlap. In Vanetik et al. (2006) and Calders et al. (2011), the authors showed that overlap-based support measures are anti-monotonic for the vertex-overlap setting. Furthermore, they claimed that the results are also valid for the edge-overlap setting, even though this should be treated more carefully.

Consider, for instance, an unlabeled database graph D which satisfies $|E(D)| > |V(D)|$, a subgraph pattern P_1 which consists of a single vertex and another subgraph pattern P_2 which consists of a single edge. If we use edge-overlap, the support of P_1 in D ($|V(D)|$) is smaller than that of P_2 in D ($|E(D)|$). Still, this violation of the anti-monotonicity property only applies to single-vertex patterns and does not affect larger patterns. As such, it does not significantly inhibit the use of support measures to prune the pattern mining search.

Definitions of overlap can be generalized. For example, the following notions of overlap could also be considered:

- *two-vertex (edge) overlap*: two images overlap if and only if they share two or more common vertices (edges);
- *label-specific overlap*: two images overlap if and only if they share a common vertex (edge) which has a label in a certain set;
- *distance-based overlap*: Two images u and v overlap if u has a vertex x and v has a vertex y such that the distance between x and y is smaller than a specified constant min_dist .

In each of these cases, small patterns need to be treated with caution, but an anti-monotonic support measure is obtained for patterns of minimal size.

The choice of overlap notion may be inspired by several factors, one of the main ones being the statistical assumptions made and the task to be performed. For instance, in Sect. 6.2 we presented a derivation for the statistical power of a sample assuming that the property of interest only depends on the properties of the nodes participating in the embedding. Suppose now that this assumption does not hold. For instance, in our lawsuit example, clients belonging to the same family might share common properties or be influenced by each other, and hence might not be independent. We could then add family relations to the graph and say that two embeddings ($client1, lawyer1, judge1, case1$) and ($client2, lawyer2, judge2, case2$) overlap if $client1$ and $client2$ are members of the same family (i.e. have a distance of at most 1 in the family relationship graph). In this way we can relax our assumptions by strengthening our notion of overlap.

7 Implementation

To conduct the experiments in Sect. 8, we implemented the $\mathbf{s}$ support measure in a pattern mining system.¹ In this section, we discuss a number of implementation-related issues.

¹ This system is part of the MIPS project, available from <http://people.cs.kuleuven.be/~jan.ramon/MiGraNT/MIPS/>

7.1 Pattern matching

As this paper focuses on the computation of a support measure, efficient pattern matching is not our main concern. We implemented a simple pattern matcher with a few optimizations. In particular, patterns are searched by backtracking. Images for the various vertices of a pattern are selected on the basis of a heuristic. Before processing a pattern P of k vertices, we computed the embeddings of all subpatterns of $k - 1$ vertices. We stored the cost of the pattern matching for all these parents and chose the 'best' parent p , i.e. the subgraph of size $k - 1$ with the smallest number of embeddings and the least time spent on pattern matching. Then, for the pattern matching of P , we order the vertices as they were ordered for p , and the vertex of P which is not in p is put at the end of the sequence. Thus, we expect to be able to prune the search space at an early stage.

7.2 Handling patterns with many embeddings

An important problem in constructing a graph pattern miner for a single network is that some patterns have an extremely large number of embeddings. As explained in Sect. 6.1, we believe that such patterns are not very interesting, and we do not calculate their exact frequency. In particular, we impose a threshold τ on the number of embeddings to be found and stop searching when the threshold is reached.

To avoid that all embeddings we found share the same first vertex, we designed our pattern matcher to perform a cyclical breadth-first search: whenever we find an embedding of a pattern, we try another image for the first vertex of the pattern next. When we have tried to find an embedding mapping the first pattern vertex on every vertex of the network, we start over again. Thus, we ensure that embeddings are distributed evenly over the network.

This approach yields three different types of subgraph patterns.

- Interesting subgraph patterns, i.e. subgraph patterns whose support is not smaller than the predefined minimum support threshold, $\{P | s(D, P) \geq \sigma \text{ and } |\text{Emb}(D, P)| \leq \tau\}$.
- Infrequent subgraph patterns, i.e. subgraph patterns whose support is smaller than the predefined minimum support threshold, $\{P | s(D, P) < \sigma \text{ and } |\text{Emb}(D, P)| \leq \tau\}$.
- Overly frequent subgraph patterns, i.e. subgraph patterns of which the number of images is larger than the predefined maximum #embedding threshold $\{P | |\text{Emb}(D, P)| > \tau\}$. Still, in practice these patterns are usually frequent, and the lower bound is close to the real support.

We find that, in practice, a number of embeddings equal to at most ten times the number of vertices in the graph is sufficient to discover all frequent patterns.

7.3 The pattern miner

We follow the classical level-wise pattern mining approach, with some modifications for patterns with an extremely large number of embeddings (see Sect. 7.2).

Our miner contains a candidate generator, a pattern matching algorithm (as discussed above) and a module for support measure computation (the main topic of this paper).

The algorithm maintains a list of candidate patterns and a list of solutions (frequent patterns). It starts with an empty pattern in the candidate list. It processes the list of candidate patterns one by one. First, it calculates the frequency of the pattern. If the pattern is frequent, it is added to the list of solutions, and all its extensions (see the candidate generation explanation) are added to the end of the candidate list. As an exception, the empty candidate has a frequency equal to the number of vertices in the database and is not considered a solution.

Candidate generation is performed by adding a single vertex to a frequent graph. This choice is preferred above the more common choice of building patterns edge by edge. Indeed, sparser graphs have a larger number of embeddings and there is a higher risk of reaching the upper bound on the number of embeddings to compute. Hopping from dense pattern to dense guarantees accurate results for the dense (and hence more interesting) patterns.

Even though we use a canonical form, we do not use canonical refinement. We generate a candidate if there is at least one parent which is frequent in the sense that we have an exact value of or lower bound on its support, higher than the minimum support threshold.

7.4 Image elimination

Although the LP can be solved very efficiently, the number of images may grow exponentially with the size of subgraph patterns. Therefore, effective optimizations can help us compute the support. In this section, we demonstrate that some images can be shown to contribute 0 to the support without being included in the LP. The following theorem holds for every OHSM.

Theorem 8 *Suppose H is an overlap hypergraph and there exist two vertices v and w in $V(H)$ for every edge $e \in E(H)$ such that $v \in e$ implies $w \in e$. Let $H' = H[V(H) - \{w\}]$, then $f(H) = f(H')$ if f is any anti-monotonic OHSM.*

Proof We use the fact that the anti-monotonic OHSM cannot decrease under VA, HS and SC. Let H be an overlap hypergraph and v and w as defined above.

On the one hand, let $H'' = VA(H', w)$, then $f(H'') \geq f(H')$ because f cannot decrease under VA. Because H can be obtained by a series of HS from H'' , we have that $f(H) \geq f(H'')$. According to the two inequalities, $f(H) \geq f(H')$.

On the other hand, because $H' = SC(G, \{v, w\}, v)$ and f cannot decrease under SC, we have that $f(H') \geq f(H)$. Thus, $f(H) = f(H')$. $\square$

This theorem tells us that if an image v is contained in more hyperedges than another image w , then we can eliminate the image v without changing any anti-monotonic OHSM. We call these images *unnecessary images*.

In our experiments, we found that many images are only contained in a single dominating hyperedge. For these images, we can use the following corollary.

Corollary 1 Suppose $H = (V, E)$ is an overlap hypergraph, let $v \in V(H)$ be an image which is only contained in a single hyperedge and $N(v)$ be the set of vertices which is adjacent to v . If f is an anti-monotonic OHSM, then $f(H) = f(H[V - \{v\} - N(v)]) + 1$.

We can show similar results for any OGSM.

Theorem 9 Suppose G is an overlap graph and there exist two adjacent vertices v and w in $V(G)$ such that $N(w) - \{v\} \subseteq N(v) - \{w\}$, let $G' = G[V(G) - \{w\}]$ then $f(G) = f(G')$ if f is any anti-monotonic OGSM.

Proof We use the fact that the anti-monotonic OGSM cannot decrease under VA, ER and CC. Let G be an overlap graph and v and w as defined above.

On the one hand, let $G'' = VA(G', w)$, then $f(G'') \geq f(G')$ because f cannot decrease under VA. We can consider G is obtained by a series of ER from G'' , we have that $f(G) \geq f(G'')$. According to the two inequalities, $f(G) \geq f(G')$.

On the other hand, because $G' = CC(G, \{v, w\}, v)$ and f cannot decrease under CC, we have that $f(H') \geq f(H)$. Thus, $f(G) = f(G')$. $\square$

If an overlap graph has m edges and a maximal degree d it only takes $O(md)$ time to eliminate all the unnecessary images from the OGSM. Hence, eliminating these unnecessary images first can speed up the computation of support measures, especially for the more expensive ones such as the Lovász ϑ and the SGM support measures.

In our experiments, we found that almost half the images in every overlap graph are unnecessary. This may inspire us to find more efficient ways to reduce the size of overlap graphs, enabling us to use the Lovász ϑ support measure and the SGM support measure in practice.

8 Experiments

This section provides experimental results, illustrating the practical potential of our new measure $\mathbf{S}$.

8.1 Experimental setup

Our experiments address the following experimental questions:

- Q1 How does the computational cost of the $\mathbf{S}$ measure compare to other existing overlap based support measures, for instance, the Lovász ϑ value?
- Q2 How high is the cost of computing the $\mathbf{S}$ measure?
- Q3 Is it feasible to mine all $\mathbf{S}$ -frequent subgraph patterns in moderately sized networks?
- Q4 What can we learn about the phase transition between frequent and infrequent patterns and the randomness of the real-world dataset?

All experiments were run on an Intel Core i7-2600 CPU (3.4Gz) with 8Gb RAM.

Table 1 Overview of real-world datasets

Dataset	Version	# Vertices	# Edges	Labels
FACEBOOK	UNIFORM	984,830	371,021	4 vertex labels
FACEBOOK	MHRW	957,359	3584,376	4 vertex labels
DBLP	0305	109,944	228,461	3 edge labels
DBLP	0507	135,516	290,363	3 edge labels

8.2 Data

In our experiments, we consider both synthetic and real-world data. We first give an overview of the datasets used.

8.2.1 Synthetic datasets

We constructed random hypergraphs RNDHYP- V - E ; where V is the number of vertices and E is the number of hyperedges. In these random hypergraphs, for every vertex v and hyperedge e with probability 0.05, v is an element of e .

We also generated scale-free networks of different sizes, using the linear preferential attachment processes described in [Barabási and Albert \(1999\)](#). We will denote such networks with 10^N vertices BARABASI- N . The network begins with an initial network of $m_0 \geq 2$ vertices (here, $m_0 = 5$) which are fully connected. New vertices are added to the network one at a time until the size of the network reaches 10^N . Each new vertex is connected to m existing vertices (in this case, $m = 5$) with a probability proportional to the number of edges that these existing nodes already have. Formally, the probability p_i that the new vertex is connected to vertex i is

$$p_i = k_i / \sum_j k_j$$

where k_j is the degree of the vertex j . Vertices are randomly labeled by 4 different labels (the labels do not affect the network generating procedure).

8.2.2 Real-world datasets

We use two labeled real-world datasets: Facebook and DBLP. Table 1 shows all real-world datasets with elementary statistics.

In the Facebook network, vertices represent users while edges represent friendship relations. The label of a vertex represents the privacy setting of the corresponding user. We consider two Facebook samples²: a uniform sample (FACEBOOK- UNIFORM) and a Metropolis-Hastings random walk sample (FACEBOOK- MHRW) ([Gjoka et al. 2010](#)).

We use two DBLP co-authorship networks (DBLP- 0305 showing co-authorships from 2003 to 2005 and DBLP- 0507 showing co-authorships from 2005 to 2007)

² http://odysseas.calit2.uci.edu/doku.php/public:online_social_networks

Fig. 6 Time required for computing the Lovász ϑ and $\mathbf{s}$

(Berlingerio et al. 2009). If an author i co-authored a paper with author j , the networks contain an undirected edge $\{i, j\}$. The vertices are unlabeled, whereas the edges are labeled with an integer indicating the year in which the edge first appeared.

8.3 Results

8.3.1 The Lovász ϑ support measure versus the $\mathbf{s}$ support measure

We compared the time needed to compute the Lovász ϑ function with the time needed to compute the $\mathbf{s}$ measure on random hypergraphs. In particular, we consider random hypergraphs RNDHYP- V - E with $V \in \{20, 40, 60, \dots, 200\}$ and $E \in \{20, 40, 60, 80, 100\}$. We evaluate $\mathbf{s}$ directly on the hypergraphs, while in order to evaluate the Lovász ϑ function we first convert the hypergraphs to graphs by replacing every hyperedge with a clique.

Figure 6 shows the time cost of computing the $\mathbf{s}$ measure and the Lovász ϑ measure for these graphs. The symbols $\theta_{\mathbf{m}}$ and $s_{\mathbf{m}}$ mean that there are m hyperedges in that hypergraph.

8.3.2 Mining patterns in real-world data

Table 2 gives the results of our algorithm for mining frequent subgraph patterns in the Facebook networks with a minimal frequency threshold of 0.2 %. Table 3 provides the same information for DBLP mined with a frequency threshold of 1 %.

As described in Sect. 7 for patterns with a huge amount of embeddings we only collect a limited number of them in our mining algorithm. Here, we put the threshold at 100,000 embeddings. Only few patterns have more than that number of embeddings, e.g. in FACEBOOK- MHRW we discovered 6 of them (all are path patterns). All patterns

Table 2 Frequent subgraph pattern mining in FACEBOOK- UNIFORM network (a) and FACEBOOK- MHRW network (b) with minimal support 0.2 %

Lev	Cand	Freq	T_{map}	T_s
(a) FACEBOOK- UNIFORM network				
1	3	3	0.01	0.00
2	2	2	0.03	1.88
3	7	6	0.26	1.96
4	41	10	0.26	0.48
5	67	11	0.15	0.40
6	72	16	0.36	0.99
7	168	26	0.31	1.63
8	337	36	0.39	2.32
9	515	43	1.63	4.23
10	847	64	0.25	5.60
11	1, 268	85	0.47	11.12
12	1, 849	25	0.83	15.12
(b) FACEBOOK- MHRW network				
1	2	2	0.01	0.00
2	2	2	0.57	1.72
3	7	5	0.79	12.43
4	40	27	0.75	10.87
5	220	101	1.19	8.17
6	480	193	1.71	4.46
7	126	80	0.19	0.62
8	104	66	4.58	0.91
9	93	30	22.73	1.20
10	29	8	12.85	0.89
11	18	0	50.97	0.01

with more than 100,000 embeddings were found to be frequent (thanks to our breadth-first way of searching for embeddings), so we didn't miss any frequent pattern.

8.3.3 Mining patterns in synthetic data

In the BARABASI- N datasets, all tree patterns are very frequent. As this is in line with the predictions of random graph theory, we only report statistics for the non-tree subgraph patterns. We set the frequency threshold to 0.1 %. In these experiments, we use the VF2 algorithm to do pattern matching (Luigi et al. 2004).

Tables 4, 5, and 6 give the results of these experiments, in which BARABASI- N (with $N \in \{2, 3, 4, 5, 6\}$) are mined for frequent non-tree subgraph patterns up to level 6 (except for the network which has 10^6 vertices). Given that we only report non-tree patterns, no patterns smaller than 4 are reported.

8.3.4 Details on some individual patterns

In this section we study a number of simple individual patterns in more detail. Triangles occur frequently in many social networks. In the patterns, labels on edges refer to the

Table 3 Frequent subgraph pattern mining in DBLP- 0305 network (a) and DBLP- 0507 network (b) with minimal support 1 %

Lev	Cand	Freq	T_{map}	T_s
(a) DBLP- 0305 network				
1	1	1	0.00	0.00
2	3	3	0.05	0.91
3	24	21	0.09	2.21
4	559	167	0.24	20.59
5	10, 746	496	0.08	0.16
6	28, 737	1, 694	1.73	0.36
(b) DBLP- 0507 network				
1	1	1	0.00	0.00
2	3	3	0.14	2.16
3	24	17	0.17	2.56
4	204	93	0.17	13.8
5	689	420	0.21	0.44

Table 4 Frequent non-tree subgraph pattern mining in the BARABASI- 2 network (a) and BARABASI- 3 network (b) with minimal support 0.1 %

Lev	Cand	Freq	T_{map}	T_s
BARABASI- 2				
4	20	16	0.014	0.383
5	191	182	0.015	0.388
6	2, 083	2, 033	0.018	0.394
BARABASI- 3				
4	20	20	0.02	0.38
5	215	215	0.03	0.43
6	2, 430	2, 422	0.13	0.48

Table 5 Frequent non-tree subgraph pattern mining in the BARABASI- 4 network (a) and BARABASI- 5 network (b) with minimal support 0.1 %

Lev	Cand	Freq	T_{map}	T_s
BARABASI- 4				
4	20	20	0.09	0.39
5	215	215	0.28	0.41
6	2, 430	2, 349	3.14	1.88
BARABASI- 5				
4	20	5	2.91	0.30
5	99	9	12.44	0.67
6	758	648	354.19	24.41

years the authors started to collaborate. We use a 3-tuple to denote the three labels of a triangle pattern, e.g. (2003, 2004, 2005) is the pattern of three authors a , b and c where a and b started collaboration in 2003, b and c started collaboration in 2004

Table 6 Frequent non-tree subgraph pattern mining in the BARABASI- 6 network with minimal support 0.1 %

BARABASI- 6				
Level	Candidates	Frequent	T_{map}	T_s
4	20	0	216.20	0.43
5	55	12	1565.13	0.99
6	–	–	–	–

Table 7 Triangle patterns and their s support values and #Image in DBLP- 0305 network

Pattern	s	#Image	#Image/ s
(2003, 2003, 2003)	7613.8	135,098	17.744
(2004, 2004, 2004)	8299.3	127,368	15.374
(2005, 2005, 2005)	10969.0	226,015	20.605
(2003, 2004, 2004)	1312.0	17,218	13.123
(2003, 2005, 2005)	1361.5	12,321	9.0496
(2004, 2005, 2005)	1555.5	16,688	10.728
(2003, 2003, 2004)	145.0	365	2.5172
(2003, 2003, 2005)	131.0	369	2.8168
(2004, 2004, 2005)	142.0	343	2.4155
(2003, 2004, 2005)	270.5	849	3.1386

and a and c started collaboration in 2005. Several triangle patterns, their number of images and their s support measures are given in Table 7.

In Table 7, we can distinguish three groups of patterns. The first group contains the patterns where the three pairs of authors started their collaboration in the same year. The second group contains the patterns where one pair started their collaboration earlier than the other two pairs, who started their collaboration in the same year. The third group contains the patterns where one pair started their collaboration later than the other two pairs, who started their collaboration in the same year. The fourth group only contains one pattern where every pair of authors started their collaboration in a different year.

From these results, we can see that the values (s support measure, number of images and their ratio) within a group are similar, while the values from different groups differ significantly.

Suppose now we randomly select in a triple of unseen authors (a, b, c) where a and b started collaboration in 2003 and b and c started collaboration in 2004. We would like to estimate the probability that a and c started a collaboration in 2004. The path of length 2 with the edges labeled 2003 and 2004 has 169,677 images (hence 169,677 embeddings) and an s value of 6467.1. In $2 \times 17218 = 34436$ of these triples, a and c started a collaboration in 2004, so for a randomly selected triple with (a, b) collaborating in 2003 and (b, c) collaborating in 2004, the probability that a and c start collaborating in 2004 is $34,436/169,677 = 20.3\%$.

The question now arises: is this estimation of 20.3 % an accurate estimation? Can we estimate the standard deviation for it? This paper, and in particular Sect. 6.2 attempts to make a first step towards answering this question.

If all 169,677 triples would be independent, the classical way to estimate the standard deviation of a binomial distribution could be used, i.e. we would calculate $\sqrt{p(1-p)/N}$ with $p = 0.203$ and $N = 169677$. We would obtain a standard deviation of 0.10 %.

However, many triples overlap. There are even more triples than there are vertices in our dataset. As a corollary of the result in Sect. 6.2, we could infer that if the triples would be distributed randomly over the graph, the collaborations not depending on the author properties, then the standard deviation could be bounded by $\sqrt{p(1-p)/s} = 0.50$ %, a significantly higher number.

Still, it is plausible that collaborating authors have some affinity in interest. Hence, it is not realistic to assume that authors collaborate randomly, and probably the standard deviation on our estimate is still significantly higher than 0.50 %. This can be seen from the high clustering coefficient (0.22442, the number of triangles over the total number of two-paths) of the network. Dealing with this more complex case where less strong assumptions hold, and more generally the analysis of prediction in networks is an interesting direction for future research.

8.4 Discussion

Based on the results presented above, we can answer the experimental questions as follows:

- Q1 Table 6 shows that, for all randomly generated (hyper)graphs, s can be computed in a very short period of time (< 0.01 s), while the time required to compute ϑ increases rapidly when the number of vertices increases. Clearly, for larger (hyper)graphs on which the s measure can be computed efficiently, it is extremely difficult to compute the ϑ value in a reasonable time by solving the corresponding SDP using existing methods. Therefore, s outperforms ϑ value in terms of efficiency.
- Q2 The cost of computing s mainly depends on the number of embeddings. This cost is low for the smallest patterns (having only a small number of embeddings with little overlap), and also, in several cases, for the largest patterns, probably because they are less frequent on average than the smaller ones.
- Q3 The s measure, allows us to mine frequent subgraph patterns of moderate size in a reasonable amount of time. The main bottleneck for DBLP is the explosive growth of the set of frequent patterns. In practical applications where it is more obvious which types of patterns are of interest, this problem could probably be overcome by selecting a smaller pattern language. For Facebook, only one label is really important, and the number of patterns increases much more slowly. In contrast to earlier approaches using the MIS or ϑ measures, where the computation of the support measure was the bottleneck, here both pattern matching and support measure computation contribute a significant fraction to the runtime. As we mainly selected the linear programming library (GLPK) based on its ease of interfacing with the other C code rather than on efficiency, we expect that the time required for support measure computation can be further reduced.

- Q4 For the synthetic data, we found that cyclic (non-tree) subgraph patterns were fairly infrequent, as we needed a frequency threshold of 0.1 % to mine them. This suggests that whereas in standard random graph models nodes choose their neighbors randomly, in real-world data the connections between candidate neighbors have an important influence.

9 Conclusions

In this paper, we have studied the problem of measuring how frequently a given subgraph pattern occurs in a given database graph. We have proposed a new overlap based support measure S , which unlike existing overlap based support measures can be computed efficiently. We have shown that this overlap based measure is anti-monotonic and normalized. The experimental results demonstrate that it is practical to use and effective in pruning the search space.

In this paper, we also argue that a support measure needs to take into account the overlap between the embeddings of the patterns. In particular, if the overlap corresponds to some type of statistical dependency, then the S measure can be interpreted as a measure of the statistical power of the sample of embeddings. Hence, this work is the first to connect support measures with statistical power. The significance of this result lies in the possibility it provides for future research to build sound statistical models on networked data, which is inherently not identically and independently drawn. Of course, this paper only takes a first step in this direction and more research is needed.

Our proposed measure is flexible, in the sense that it is possible for a user to plug in his own definition of overlap. It also makes it possible to measure the frequency of a subgraph pattern in a more sound statistical way. There are, however, other challenges related to subgraph pattern mining in networks. The major one in our experiments was subgraph pattern matching. However, we anticipate that here too we will be able to make considerable advances in finding tractable solutions. In particular, we intend to integrate our approach with recent results concerning efficient subgraph pattern matching operators based on arithmetic circuits (Kibriya and Ramon 2012).

Several interesting problems remain. First, our current approach relies on first finding all embeddings of a pattern, even though there are indications that for approximate results this may not be necessary. It would be interesting to investigate whether a sample of the embeddings would allow us to compute the S measure efficiently and accurately. Second, closer analysis of the statistical power of samples could produce approaches which relieve the assumptions made about independence of the embeddings and the node properties in Sect. 6.2. Finally, now that we have established a connection between sets of embeddings and statistical power; we can start thinking about exploiting this connection by developing statistically sound learning algorithms on networked data. and providing learning guarantees in a similar way as is common in the propositional setting.

Acknowledgements This work was supported by ERC Starting Grant 240186 “MiGrANT: Mining Graphs and Networks: a Theory-based approach”.

References

- Agrawal R, Imielinski T, Swami A (1993) Mining association rules between sets of items in large databases. In: Proceedings of SIGMOD'93, Washington DC, pp 207–216
- Barabási A, Albert R (1999) Emergence of scaling in random networks. *Science* 286:509–512
- Berlingiero M, Bonchi F, Bringmann B, Gionis A (2009) Mining graph evolution rules. In: Proceedings of ECML/PKDD'09, Bled, pp 115–130
- Boyd S, Vandenberghe L (2004) *Convex optimization*. Cambridge University Press, Cambridge
- Bringmann B, Nijssen S (2008) What is frequent in a single graph? In: Proceedings of PAKDD'08, Osaka, pp 858–863
- Calders T, Ramon J, Dyck DV (2011) All normalized anti-monotonic overlap graph measures are bounded. *Data Min Knowl Discov* 23(3):503–548
- Chakrabarti D, Faloutsos C (2006) Graph mining: laws, generators, and algorithms. *ACM Comput Surv* 38(1):1–69
- Chan T, Chang KL, Raman R (2009) An SDP primal-dual algorithm for approximating the Lovsz-theta function. In: Proceedings of the IEEE ISIT'09, pp 2808–2812
- Diestel R (2010) *Graph theory*. Springer, Heidelberg
- Dreweke A, Wörlein M, Fischer I, Schell D, Meinel Th, Philippsen M (2007) Graph-based procedural abstraction. In: Proceedings of the international symposium on code generation and optimization'07, San Jose, pp 259–270
- Fagin R (1976) Probabilities on finite models. *J Symb Logic* 41(1):50–58
- Fiedler M, Borgelt C (2007) Support computation for mining frequent subgraphs in a single graph. In: Proceedings of the workshop on mining and learning with graphs (MLG'07), Firenze
- Feige U, Goldwasser S, Lovász L, Safra S, Szegedy M (1991) Approximating clique is almost NP-complete. In: FOCS IEEE Computer Society, pp 2–12
- Garey MR, Johnson DS (1979) *Computers and intractability, a guide to the theory of NP-completeness*. W. H. Freeman and Company, New York
- Gjoka M, Kurant M, Butts C, Markopoulou A (2010) Walking in facebook: a case study of unbiased sampling of OSNs. In: Proceedings of IEEE INFOCOM'10, San Diego, pp 1–9
- Iyengar G, Phillips DJ, Stein C (2011) Approximating semidefinite packing programs. *SIAM J Optim* 21(1):231–268
- Kibriya A, Ramon J (2012) Nearly exact mining of frequent trees in large networks. In: Proceedings of ECML-PKDD 2012, Bristol, pp 426–441
- Klein PN, Lu H (1996) Efficient approximation algorithms for semidefinite programs arising from MAX CUT and COLORING. In: Proceedings of ACM STOC'96, pp 338–347
- Knuth DE (1994) The sandwich theorem. *Electron J Comb* 1:1–48
- Kuramochi M, Karypis G (2005) Finding frequent subgraph patterns in a large sparse graph. *Data Mining Knowl Discov* 11(3):243–271
- Lovász L (1979) On the Shannon capacity of a graph. *IEEE Trans Inf Theory* 25(1):1–7
- Luigi P, Pasquale F, Carlo S, Mario V (2004) A subgraph isomorphism algorithm for matching large graphs. *IEEE Trans Pattern Anal Mach Intell* 26(10):1367–1372
- Schrijver A (1979) A comparison of the Delsarte and Lovász bounds. *IEEE Trans Inf Theory* 25:425–429
- Vanetik N, Gudes E, Shimony SE (2002) Computing frequent graph subgraph patterns from semistructured data. In: Proceeding of the IEEE international conference on data mining (ICDM'02), Maebashi, pp 458–465
- Vanetik N, Shimony SE, Gudes E (2006) Support measures for graph data. *Data Mining Knowl Discov* 13(2):243–260
- Wang Y, Ramon J (2012) An efficiently computable support measure for frequent subgraph pattern mining. In: Proceedings of ECML-PKDD 2012, Bristol, pp 362–377