Generalizing Design of Support Measures for Counting Frequent Patterns in Graphs

Jinghan Meng
University of South Florida
Tampa, FL
jmeng@mail.usf.edu

Napath Pitaksirianan
University of South Florida
Tampa, FL
napath@mail.usf.edu

Yi-Cheng Tu
University of South Florida
Tampa, FL
tuy@mail.usf.edu

ABSTRACT

Frequent subgraph mining (FSM) from graphs is an active subject in data mining research, and is gaining momentum with the popularity of applications such as social networks and the web. One major challenge in FSM is the development of support measures, which are basically functions that map a pattern to its frequency count in a database. Current state-of-the-art in this topic features a hypergraph-based framework for modeling pattern occurrences which unifies the two main flavors of support measures: the overlap-graph based maximum independent set measure (MIS) and minimum image/instance based (MNI) measures. For the purpose of exploring the middle ground between these two groups and guiding the development of new support measures, we present sufficient conditions for designing new support measures in hypergraph framework. Furthermore, we utilize the sufficient conditions to generalize MI for designing user-defined linear-time measures. Furthermore, we introduce a new polynomial-time support measure, called maximum independent subedge set (MISS) measure to fill the gap between MIS and MI in terms of computation complexity and support count. To the best of our knowledge, MISS is the first non-relaxation polynomial-time support measure that is close to MIS in the counts returned. We further show MISS's advantage over other support measures derived from linear programming relaxations. Experiments conducted on real-world graph datasets show that MISS returns significantly smaller counts than MI, and similar or smaller counts as other relaxation-based measures. Bounding theorems among all relevant support measures are also presented in this paper.

CCS CONCEPTS

• **Information systems** → **Graph-based database models**; • **Mathematics of computing** → *Graph algorithms*;

KEYWORDS

Data mining, graph mining, support measure

ACM Reference Format:
Jinghan Meng, Napath Pitaksirianan, and Yi-Cheng Tu. 2018. Generalizing Design of Support Measures for Counting Frequent Patterns in Graphs. In *Proceedings of Technical Report, Department of Computer Science and Engineering, University of South Florida (CSE-TR18-002).* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

CSE-TR18-002, January 2018, Tampa, Florida

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The field of graph mining has attracted a lot of attention with applications in social networks, the web, and aviation maps. One of the important problems is frequent subgraph mining (FSM), which aims to identify frequent patterns for the purpose of studying trend or locating hotspots. In such efforts, an essential component is the support measure of the given pattern, i.e., the frequency the pattern appears in the graph database. In this paper, we report our recent work related to support measures in the *single-graph* setting, in which the database is only one (often large) graph.

Defining support measures is a non-trivial matter. First of all, a support measure has to satisfy the so-called *anti-monotonicity* property, which states that the frequency of a pattern should not exceed that of any of its subpatterns. This rule is essential to the majority of FSM algorithms because they rely on it to safely prune infrequent patterns from a tree structure to save time. Other facts to consider include computational cost and intuitive frequency counts.

The first anti-monotonic support measure is called the *maximum independent set based* (MIS) support [25]. The main idea of MIS is to count maximum number of independent instances of a pattern from a data structure named *overlap graph*. While providing an intuitive count, MIS shows a major drawback in its lack of efficient algorithms – it is proved to be NP-hard that makes MIS infeasible to compute in even moderately sized graph datasets. Based on the technique of vertex images, another type of support measure named the minimum-image-based (MNI) support [10] was designed. As another anti-monotonic support, MNI is linear-time computable. However, MNI falls short in *intuitiveness* - it could arbitrarily overestimate the frequency of a pattern. This is due to its lack of consideration of topological structure of the query pattern.

While the above measures are developed under different mathematical underpinnings, it is natural to ask the question whether there is an universal foundation for designing and profiling support measures in FSM. In addition to satisfying mathematical curiosity, it carries high practical value to develop a framework, within which we can compare existing measures and see different levels of tradeoff between efficiency and intuitiveness. One urgent task is to develop new support measures that sacrifice efficiency in exchange of higher intuitiveness. It is also necessary to develop new support measures to cater to the heterogeneous needs of FSM applications. Different users may have different types of experimental data and treat data features and pattern significance differently. Therefore, it is highly desirable if there is a framework that allows users define

their own support measures while enjoying same/similar guarantees on counts and computing time.

Our recent work [20] introduced a general framework based on the concept of *occurrence and instance hypergraph* to model many support measures. In this framework, MIS is shown to be equivalent to a *maximum independent edge set* (MIES) support, and MNI can be directly interpreted as a special case of the so-called *minimum-instance-based measure* (MI). Under this framework, MNI and MIS/MIES can be naturally seen as the upper and lower bounds of the frequency spectrum among state-of-the-art measures. Moreover, another measure called the *minimum vertex cover measure* (MVC) is derived as an ultimate version of MNI. MVC is also connected with MIS through the concepts of dual hypergraph. Polynomial-time linear programming relaxations of MIES and MVC, denoted as RMIES and RMVC, stand in-between MIS and MVC in terms of count value. Bounding theorems that compare the frequency counts returned by aforementioned support measures (given the same inputs) are given as:

$$MIS = MIES \leq RMIES = RMVC \leq MVC \leq MI \leq MNI \quad (1)$$

Contributions: In this paper, we report important new discoveries that further completes the hypergraph-based approach.

First, for thorough investigation of various support measures in the hypergraph framework, we study basic principles for designing support measures. Specifically, we present sufficient conditions for any function to be anti-monotonic so that it can be admitted as a support measure. Such conditions provide a strong guideline in developing new support measures.

Second, as applications of the sufficient conditions, we present: (1) a general version of the MI-flavored support measures, and (2) a new support measure to showcase the effectiveness of such theoretical guidelines. The discovery of the new support measure, which we named **maximum independent subedge set (MISS) support**, is an exciting breakthrough by itself. Note that computing MIS/MIES and MVC are NP-hard, and that for MI and MNI are linear. Naturally, one would ask if there exist support measures between MIES/MVC and MI/MNI in terms of computational complexity and counts. Such measures would be highly valuable as they provide another level of tradeoff between intuitiveness and efficiency. RMIES and RMVC are such measures however their computational complexity is still of high-order polynomial. MISS fills the aforementioned gap: it is proved to be anti-monotonic, returns counts between MIS and MI, and can be computed in low-order polynomial time. To the best of our knowledge, it is the first non-relaxation polynomial-time support measure that is close to MIS. Experiments conducted on different real-world graph datasets verify our theoretical conclusions and show that MISS returns significantly smaller counts than RMVC in many cases.

Other Related Work: A number of research work investigated support measures in a single graph via various approaches: overlap-graph based measures [11, 14, 19, 25, 26, 28] and minimum-image-based measure [10]. A sufficient and necessary condition was given in overlap-graph framework [25], however it is fundamentally different from our hypergraph approach. In particular, it cannot describe minimum image/instance measures such as MNI and MI.

2 PRELIMINARIES

In this section, we introduce basic notations and concepts to describe the problem and the necessary background.

2.1 Labeled Graphs

The graph considered in this paper is vertex-labeled graph, which is simply referred to as *graph* hereafter. In all figures of this article, we represent the label of a vertex by the shade.

Definition 1. *A (undirected)* ***labeled graph***

$$G = (V_G, E_G, \lambda_G)$$

consists of a set of vertices V_G, a set of edges $E_G \subseteq V_G \times V_G := \{(u, v) \mid u, v \in V_G,\ u \neq v\}$ and a labeling function $\lambda_G : V_G \to \Sigma$ that maps each vertex of the graph to an element of the alphabet Σ.

In this paper, we write graph $G = (V_G, E_G, \lambda_G)$ as $G = (V_G, E_G)$ when there is no ambiguity.

Definition 2. *A graph $S = (V_S, E_S, \lambda_S)$ is a* ***subgraph*** *of $G = (V_G, E_G, \lambda_G)$, if V_S is a subset of V_G and E_S is a subset of E_G and for all $v \in V_S$, $\lambda_S(v) = \lambda_G(v)$.*

Definition 3. *A* ***pattern*** *$p = (V_p, E_p, \lambda_p)$ is a labeled graph we use as a query against another graph.*

Definition 4. *Let P be a graph pattern, and p a subgraph of P, denoted by $p \subseteq P$. We call p a* ***subpattern*** *of P, and likewise, we call P a* ***superpattern*** *of p.*

After knowing copies of pattern p exist in data graph G, we are focused on how many times it appears in G. For that, we first need to define the concepts of *occurrence* and *instance* of the pattern in the data graph.

Definition 5. *Given a pattern p and a data graph G, an* ***occurrence*** *f is an isomorphism (an edge-preserving bijection between the vertex sets of two graphs) between pattern p and a subgraph of G. That is to say f is also a subgraph isomorphism from p to G.*

Definition 6. *Given a pattern p and a data graph G, a subgraph S of G is an* ***instance*** *of pattern p in G when there exists an isomorphism between p and S.*

2.2 Overlap Concepts and Support Measure

The purpose of defining support measure is to count the appearances of a pattern p in a data graph G. The definition of support measure is given below:

Definition 7. *A* ***support measure*** *of pattern p in data graph G is a function $\sigma : G \times G \to \mathbb{R}^+$, which maps (p, G) to a non-negative number $\sigma(p, G)$.*

Naturally, one would pick the number of occurrences or instances of a pattern as support measure. Unfortunately, such choices violate the *anti-monotonicity* rule [19, 25]. For example, in Fig. 1, pattern p has four occurrences f_1, f_2, f_3, f_4 in data graph, but its superpattern P_1 has five occurrences $f'_{1,1}, f'_{1,2}, f'_{1,3}, f'_2, f'_3$.

Definition 8. *A support measure σ of pattern p in G is* ***anti-monotonic*** *if for any pattern p and its superpattern P, we have $\sigma(p, G) \geq \sigma(P, G)$.*

Definition 9. ***Vertex overlap**: A vertex overlap of occurrences f_1 and f_2 of pattern $P = (V_P, E_P)$ in data graph $G = (V, E)$ exists if vertex sets $f_1(V_P)$ and $f_2(V_P)$ intersect, that is, $f_1(V_P) \cap f_2(V_P) \neq \emptyset$ where $f_i(V_P) = \{f_i(v) : v \in V_P\}$, $i = 1, 2$. A vertex overlap of instances $S_1 = (V_{S_1}, E_{S_1})$ and $S_2 = (V_{S_2}, E_{S_2})$ of pattern P exists if vertex sets of S_1 and S_2 intersect, that is, $V_{S_1} \cap V_{S_2} \neq \emptyset$.*

In this paper we consider two occurrences "overlap" when they overlap at one or more than one vertex.

MIS [25] is the first non-trivial anti-monotonic support measure. It is essentially the count of non-overlapping (independent) pattern occurrences (instances) .

An anti-monotonic and linear-time computable support, called *minimum image based support* (MNI) is shown in [15]. The main concept behind MNI is *image*, which is an existence of a vertex in the pattern (called *node* hereafter) in the data graph, and the minimum distinct vertex image count is used as the support measure.

Our recent work [20] developed a framework that unifies all existing measures (including both MIS and MNI), and provides a foundation for introducing and analyzing new support measures. This framework is built on the concept of *hypergraph*.

Definition 10. *Assume that pattern $p = (V_p, E_p)$ has m occurrences $\{f_i\}_{1 \le i \le m}$, the **occurrence hypergraph** of p in G is defined as $H_p = (V, E)$ where $V = \cup_{1 \le i \le m} f_i(V_p)$, and $E = \{e_i\}_{1 \le i \le m}$, each $e_i = f_i(V_p)$. In other words, hypergraph vertex set V is the collection of all pattern node images, and each edge e_i is a collection of pattern node images mapped by occurrence f_i. For each e_i, a label f_i is given to distinguish it from others.*

For example, in Fig. 1, occurrence hypergraphs of p and P are shown as H_p and H_P. Each edge e_i in H_p corresponds to occurrence f_i, $i = 1, 2, 3, 4$. In H_P, edges $e'_{1,1}$, $e'_{1,2}$, $e'_{1,3}$, e'_2, e'_3 correspond to $f'_{1,1}$, $f'_{1,2}$, $f'_{1,3}$, f'_2, f'_3 respectively. Note that an edge in a hypergraph (called *hyperedge*) could contain more than two vertices.

Although defined in the overlap graph, MIS can be mapped to the MIES support defined in the hypergraph space [20], which uses the maximum independent occurrences as frequency count.

Another important measure defined in [20] is the minimum instance based support measure MI.

Definition 11. *Given a pattern $p = (V_p, E_p)$, a data graph $G = (V, E)$, let T be a transitive node subset in a subgraph of pattern p, the collection of all such T is denoted as $\mathcal{T} = \{T\}$. The **minimum instance based support (MI)** of p in G is defined as*

$$\sigma_{MI}(p, G) = \min_{T \in \mathcal{T}} c(T),$$

where $c(T)$ is the count of images of node set T under all occurrences of p in G.

3 A SUFFICIENT CONDITION FOR ANTI-MONOTONICITY

As mentioned earlier, sufficient conditions can reveal key factors of support measures and help design desired new support measures. In this section, we give sufficient conditions for verifying the anti-monotonicity of a candidate support measure. The idea is that, if we can find a series of operations on occurrence hypergraph H_p of pattern p which transform it to occurrence hypergraph H_P of its

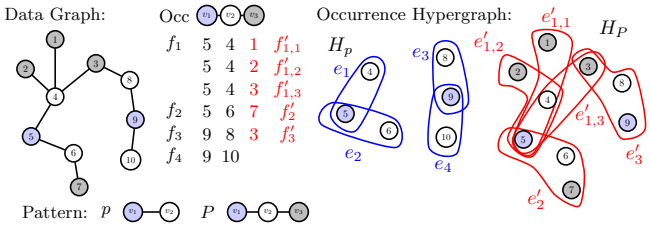

Figure 1: An example of occurrence hypergraphs generated from two patterns over a data graph. Occurrences and hyperedges of the superpattern are marked red

superpattern P, and if a non-negative function σ is non-increasing under these operations, then this function is anti-monotonic and thus admissable as a support measure.

In this paper, we propose three operations to convert occurrence hypergraph of a pattern $p = (V_p, E_p)$ to that of its superpattern $P = (V_P, E_P)$. Without loss of generality, we consider pattern p and its superpattern P in the case where P has one more edge than p. There are two scenarios: **(i)** $V_P = V_p$ and $E_P = E_p \cup \{u, v\}$ where $u, v \in V_p = V_P$. The extra edge of P connects existing pattern nodes u, v. **(ii)** $V_P = V_p \cup z$ and $E_P = E_p \cup \{u, z\}$ where $u \in V_p, z \in V_P$. The extra edge of P connects existing pattern node u and extra node z. For a pattern p and its superpattern P, we say an occurrence f of p can **extend** if in scenario (i) there exists an edge $\{f(u), f(v)\}$ in data graph; or in scenario (ii) there exists a vertex z' in data graph, which has the same label as node z and edge $\{f(u), z'\}$ is in data graph, hence there exists an occurrence f' of P that is identical to f on V_P. We say an edge e in H_p can extend to e' in H_P if the corresponding occurrence f can extend to f'. For instance, in Fig. 1, f_2 and e_2 can extend. For a node set $T \subseteq V_p$, the **image** of T under f and f's corresponding edge e is the vertex set $f(T) = \{f(v)\}_{v \in T}$.

Conversion Procedure: We have the following procedure to convert H_p into H_P.

Step (1): Delete all edges that cannot extend.

Assume that $H_p = (V, E)$, $E' \subseteq E$ is the set of edges that cannot extend, we delete each edge $e \in E'$ from hypergraph H_p. After this operation, we obtain a new hypergraph named H^{Del} and every edge e in it can extend. For example, in Fig. 2, edge e_4 cannot extend and it is deleted from H_p.

In scenario (i) mentioned earlier, p and P have the same set of pattern nodes, each e can represent a superpattern occurrence f', we shall label e with f' then the conversion is completed. Otherwise, in scenarios (ii) the conversion is still not accomplished, because P has an additional node z than pattern p. Each edge e shall extend to one or more edges e' with an additional vertex.

Step (2): Replace each e by extended edges, which do not overlap at newly added vertex in them.

Assume that $H^{Del} = (V, E)$, $E = \{e_i\}_{1 \le i \le n}$, each e_i extends to n_i edges $\{e'_{i,j}\}_{1 \le j \le n_i}$, where $n_i \ge 1$. For each $1 \le j \le n_i$, we create a vertex $z_{i,j}$ with the same label as pattern node z, and get extended edge $e^*_{i,j} = e_i \cup z_{i,j}$. Next, we replace e_i by $E^*_i = \{e^*_{i,j}\}_{1 \le j \le n_i}$. Note that $e^*_{i,j}$ is different from $e'_{i,j}$ in vertex $z_{i,j}$. All $z_{i,j}$ created are mutually different in vertex number but have the same label as z

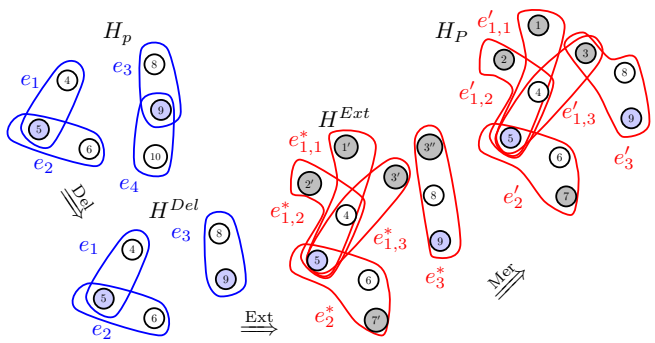

Figure 2: Three operations transform H_p to H_P as shown in Fig. 1

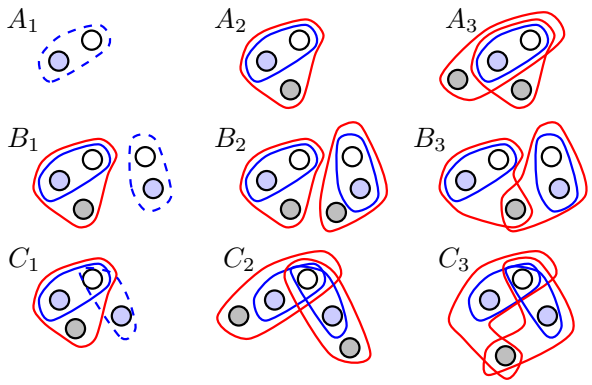

Figure 3: Different scenarios in which pattern p extends to superpattern P. Blue edge denotes an occurrence of pattern p and red edge represents an occurrence of superpattern P

since they are images of z. The purpose is to ensure extended edges do not overlap at newly added vertices. After this operation, we obtain a new hypergraph named H^{Ext}.

For instance, in Fig. 2, pattern node z is v_3, edge $e_1 = \{4, 5\}$ is replaced by $\{e^*_{1,1} = \{4, 5, 1'\}, e^*_{1,2} = \{4, 5, 2'\}, e^*_{1,3} = \{4, 5, 3'\}\}$, vertices $1', 2', 3'$ are distinct but have the same label (color gray) as v_3.

Step (3): Replace newly added vertices by z's images in data graph G.

For $1 \le i \le l$, $1 \le j \le n_i$, edge $e^*_{i,j}$ corresponds to $f'_{i,j}$, and the image of z under $f'_{i,j}$ is $f'_{i,j}(z)$ in G. We replace $z_{i,j}$ by $f'_{i,j}(z)$ in edge $e^*_{i,j}$ to get $e'_{i,j}$. After this operation, we obtain H_P.

In Fig. 2, v_3's image under $f'_{1,1}$ is 1 in data graph, hence in $e^*_{1,1}$ we replace vertex $1'$ by 1 to obtain $e'_{1,1}$. We also replace $3'$ and $3''$ by 3 for the same reason. In Step (2), distinct vertices $1', 2', 3', 3''$ are temporarily created to avoid "overlap." Now some of them are replaced by the same vertex, it is visualized as we "merge" vertices together, e.g., $3'$ and $3''$ are merged into one vertex 3. Now we give formal definition of these operations and notions as follows.

Definition 12. *For a pattern p, **Pattern Hypergraph Space** $\mathcal{H}(p)$ is defined as the collection of H_p and the derived hypergraphs H^{Del} and H^{Ext} in the Conversion Procedure from H_p to an H_P.*

Definition 13. ***Edge Deletion*** *operation $Del(H_p, p, P, G)$ deletes edge e that cannot extend from hypergraph H_p.*

Assume the following operations are for p and P in scenario (ii) when $V_P = V_p \cup z$.

Definition 14. ***Edge Extension*** *operation $Ext(H^{Del}, p, P, G)$ replaces every edge e_i in H^{Del} by $E^*_i = \{e^*_{i,j}\}_{1 \le j \le n_i}$, as defined in Step (2) of the conversion procedure.*

Definition 15. ***Vertex Merge*** *operation $Mer(H^{Ext}, p, P, G)$, in each edge of H^{Ext}, replaces the image of z by the image of z in data graph G under the corresponding occurrence, as defined in Step (3) of the conversion procedure.*

Hence the Conversion Procedure follows three operations in the strict order of *Del*, *Ext* and *Mer*. Now let us show that the above procedure always works for any p, P, and G.

Theorem 1. *The occurrence hypergraph H_p is transformed into H_P by following the above Conversion Procedure.*

Proof. We shall show that every edge in H_p is converted, and every edge in H_P is generated and generated once.

First, let us take a look at different scenarios when the occurrences of pattern p extend to occurrences of P (Fig. 3).

1) Consider a single occurrence, there are two scenarios:
   Case A_1: it cannot extend,
   Case A_2, A_3: it extend to one or more occurrences of P.
2) Consider pairwise occurrences, there are six scenarios that are categorized by "overlap":
   I : Occurrences are **independent** (do not mutually overlap) before extension:
   Case B_1: some of them cannot extend
   Case B_2: they remain independent after extension,
   Case B_3: some of them overlap at newly added vertices.
   II : Occurrences **overlap** at one or more than one vertex before extension:
   Case C_1: some of them cannot to extend,
   Case C_2: they do not overlap at newly added vertices.
   Case C_3: some of them overlap at newly added vertices.

Although there are many scenarios, we shall show how effective the proposed three operations can handle them all.

The *Del* operation will handle the cases A_1, B_1, C_1, it deletes occurrences that cannot extend. All the remaining edges can extend.

There are multiple cases of "overlap" before and after extension. The proposed *Ext* operation handles cases A_2, A_3, B_2, C_2, in which extended edges do not overlap at new vertices.

For the cases in which extended edges overlap at new vertices the *Mer* operation transforms B_2, C_2 to B_3, C_3 by replacing distinct vertices with the same vertex.

Hence we conclude that the proposed procedure is suitable for covering all extension scenarios and every edge in H_p is deleted or replaced. On the other hand, because we extend each edge e_i in H^{Del} in all possible ways (n_i ways) and every edge in H_P must be an extension from some edge in H_p, thus every edge in H_P is obtained. If there are two different edges e_1, e_2 in H_p extend to the same e' in H_P, then their corresponding occurrences f_1, f_2, as restriction function of the same f' on V_p, must be identical. Hence

we get $e_1 = e_2$, this is a contradiction. Therefor every edge $e' \in H_P$ is generated only once. This completes the proof. □

Now we present our main theorem, which becomes obvious once we established the above results.

Theorem 2. ***(Sufficient Conditions)*** *In hypergraph framework, for a pattern p, a superpattern P who has one more edge that p, and data graph G, a non-negative function $\sigma(p, G)$, as a restriction of function $\sigma(H, p, G)$(where $H \in \mathcal{H}(p)$) on H_p, is an anti-monotonic support measure if $\sigma(H, p, G)$ is non-increasing under of these three operations Del, Ext, Mer in the sense that:*

(1) $\sigma(H_p, p, G) \geq \sigma(H^{Del}, p, G)$, $\sigma(H^{Del}, p, G) \geq \sigma(H^{Del}, P, G)$,
(2) $\sigma(H^{Del}, p, G) \geq \sigma(H^{Ext}, P, G)$,
(3) $\sigma(H^{Ext}, P, G) \geq \sigma(H_P, P, G)$.

Proof. In the Conversion Procedure, H_p is transformed to H^{Del} by *Del*, then to H^{Ext} by *Ext*, then to H_P by *Mer*. If $\sigma(H, p, G)$ is non-increasing under these operations, then in scenario (i) $\sigma(p, G) = \sigma(H_p, p, G) \geq \sigma(H^{Del}, p, G) \geq \sigma(H^{Del}, P, G) = \sigma(P, G)$; in scenario (ii) $\sigma(p, G) = \sigma(H_p, p, G) \geq \sigma(H^{Del}, p, G) \geq \sigma(H^{Ext}, P, G) \geq \sigma(H_P, P, G) = \sigma(P, G)$. Hence $\sigma(p, G)$ is anti-monotonic. □

Intuitively, the sufficient conditions say that when the number of occurrences decreases, or occurrences extend, or occurrences overlap at more vertices, a function should not increase to qualify as a support measure.

3.1 Verification of Sufficient Conditions

In fact, all existing support measures in hypergraph framework are defined in such a boarder sense like $\sigma(H, p, G)$, that means they can work on not only one H_p, but also on uniform hypergraphs derived from H_p, e.g., H^{Del}, H^{Ext}. All of them can be verified to satisfy the sufficient conditions in Theorem 2. Such measures include MI (see section 4), MVC, MIS/MIES, RMVC and RMIES.

We first define MVC, MIES, RMVC, and RMIES in hypergraph framework, then show that they are non-increasing under these three operations, so that we obtain their anti-monotonicity according to Theorem 2.

Definition 16. *Given a pattern p and a data graph G, in hypergraph framework, for any $H \in \mathcal{H}(p)$, a function $\sigma(H, p, G)$ is defined as the size of minimum vertex cover of H and a* ***minimum vertex cover (MVC)*** *support of p in G, as a restriction of σ on H_p, is defined as the size of minimum vertex cover of H_p.*

Theorem 3. *The MVC support measure is anti-monotonic.*

Proof. We shall verify that σ is non-increasing under *Del*, *Ext*, and *Mer* in Theorem 2.

Denote the collection of minimum vertex cover of a hypergraph H by $\mathcal{MVC}(H)$. The size of a $M \in \mathcal{MVC}(H)$ is denoted as $a(H)$.

After *Del*, $H^{Del} \subseteq H_p$, edges in H^{Del} are also in H_p, thus any $M \in \mathcal{MVC}(H_p)$ that covers all edges in H_p can cover all edges in H^{Del}. That is to say the size of M is greater or equal to that of any $M' \in \mathcal{MVC}(H^{Del})$. Hence the first inequality in (1) of Theorem 2 is satisfied. In scenario (i) because $V_p = V_P$, the edges are the same vertex sets. Hence the second inequality in (1) of Theorem 2 is satisfied.

After *Ext*, because any edge in H^{Ext} contains an edge in H^{Del} as a subset, we have any $M \in \mathcal{MVC}(H^{Del})$ that covers all edges in H^{Del} can cover all edges in H^{Ext}. Hence we obtain the inequality (2) in Theorem 2.

Because $V_P = V_p \cup z$ and $z \notin V_p$, the operation *Mer* only replace vertices that are related to z's images which are not part of images of V_p, we conclude that after *Mer*, the images of V_p under edges in H_P are the same of that of V_p under H^{Ext}. Thus any $M \in \mathcal{MVC}(H^{Ext})$ that covers all edges in H^{Ext} can cover all edges in H_P. It implies that the size of M is greater or equal to that of any $M' \in \mathcal{MVC}(H_P)$. Therefore the inequality (3) in Theorem 2 is satisfied. Now we obtain the anti-monotonicity of $MVC(p, G)$. □

Definition 17. *Given a pattern p and a data graph G, in hypergraph framework, for any $H \in \mathcal{H}(p)$, a function $\sigma(H, p, G)$ is defined as the size of maximum independent edge set of H and a* ***maximum independent edge set (MIES)*** *support of p in G, as a restriction of σ on H_p, is defined as the size of maximum independent edge set of H_p.*

Theorem 4. *The MIES support measure is anti-monotonic.*

Proof. We shall check if σ is non-increasing under *Del*, *Ext*, and *Mer* in Theorem 2.

Denote the collection of maximum independent edge set of a hypergraph H by $\mathcal{M}(H)$. The size of an $M \in \mathcal{M}(H)$ is denoted as $mies(H)$. Thus we have $MIES(p, G) = \sigma(H_p, p, G) = mies(H_p)$.

After *Del*, $H^{Del} \subseteq H_p$, edges in H^{Del} are also in H_p. For any $M \in \mathcal{M}(H^{Del})$, it is also an independent edge set in H_p. It implies that $mies(H_p) \geq mies(H^{Del})$. Hence we get $\sigma(H_p, p, G) = mies(H_p) \geq mies(H^{Del}) = \sigma(H^{Del}, p, G)$, the first inequality in (1) of Theorem 2 is obtained. In scenario (i) because $V_p = V_P$, after *Del*, every $e \in H^{Del}$ can represent an occurrence f of p and at the same time f's extension occurrence f' of P. Therefore $\sigma(H^{Del}, P, G) = mies(H^{Del}) = \sigma(H^{Del}, p, G)$, which means the second inequality in (1) of Theorem 2 is satisfied.

After *Ext*, any edge e' in H^{Ext} is extended from an edge e in H^{Del} as a subset. Therefore for an $M \in \mathcal{M}(H^{Del})$, say $M = \{e'_i\}_{1 \leq i \leq n}$, each e'_i is extended from e_i in H^{Del}. Apparently $\{e_i\}_{1 \leq i \leq n}$ is an independent edge set in H^{Del}, which implies that $mies(H^{Del}) \geq mies(H^{Ext})$. Hence $\sigma(H^{Del}, p, G) = mies(H^{Del}) \geq mies(H^{Ext}) = \sigma(H^{Ext}, P, G)$, we obtain the inequality (2) in Theorem 2.

Assume $M \in \mathcal{M}(H_P)$ and $M = \{e'_i\}_{1 \leq i \leq n}$. In each edge e'_i, if we replace image of z by the image of z under e_i in H^{Ext}, it is also an independent edge set in H^{Ext}. It implies that $mies(H^{Ext}) \geq mies(H_P)$. Hence $\sigma(H^{Ext}, P, G) = mies(H^{Ext}) \geq mies(H_P) = \sigma(H_P, P, G)$. Therefore the inequality (3) in Theorem 2 is satisfied. Now we obtain the anti-monotonicity of $MIES(p, G)$. □

We now formally define the standard linear programming relaxation versions of the MVC and MIES measures. We shall show that they are both anti-monotonic in the sense of the three operations.

Definition 18. *Given a pattern p and a data graph G, in hypergraph framework, for any $H \in \mathcal{H}(p)$, where $V = \{v_i\}_{1 \leq i \leq n}$ and $E = \{e_i\}_{1 \leq i \leq m}$, a function $\sigma(H, p, G)$ is defined as the solution of*

linear programming relaxation of minimum vertex cover problem on H:

$$\sigma_{RMVC}(p,G) = \min \sum_{v\in V} x(v) \tag{2}$$
$$\text{subject to} \quad \sum_{v\in e_i} x(v) \geq 1 \qquad e_i \in E$$
$$0 \leq x(v) \leq 1 \qquad v \in V$$

*and the **polynomial-time relaxed MVC (RMVC)** support measure of pattern P in graph G, as a restriction of σ on H_p, that is*

$$\sigma_{RMVC}(p,G) = \min \sum_{v\in V_p} x(v) \tag{3}$$
$$\text{subject to} \quad \sum_{v\in e_i} x(v) \geq 1 \qquad e_i \in E_p$$
$$0 \leq x(v) \leq 1 \qquad v \in V_p$$

Theorem 5. *The RMVC support measure is anti-monotonic.*

Proof. We shall verify that σ is non-increasing under *Del*, *Ext*, and *Mer*.

After *Del*, $H^{Del} = (V_D, E_D)$, $H^{Del} \subseteq H_p$, edges in H^{Del} are also in H_p, and $V_D \subseteq V_p$. Let x^* be the solution of equation (3) for H_p. Since $V_D \subseteq V_p$, if we let $x^{**} = x^*$ for $v \in V_D$ and $x^{**} = 0$, for $v \in V_p - V_D$ then $\sum_{v\in e_i} x^{**}(v) \geq 1$, $e_i \in E_D$. Hence we have x^{**} is greater or equal to the solution of

$$\sigma(H^{Del},p,G) = \min \sum_{v\in V_D} x(v) \tag{4}$$
$$\text{subject to} \quad \sum_{v\in e_i} x(v) \geq 1 \qquad e_i \in E_D$$
$$0 \leq x(v) \leq 1 \qquad v \in V_D$$

Hence the first inequality in (1) of Theorem 2 is satisfied. In scenario (i) because $V_p = V_P$, it is obviously that the second inequality in (1) of Theorem 2 is satisfied.

After *Ext*, because any edge in $H^{Ext} = (V_E, E_E)$ contains an edge in H^{Del} as a subset, we have $V_D \subseteq V_E$, and for $e \in E_D$, there exists $e' \in E_E$ such that $e \subseteq e'$. We have

$$\sigma(H^{Ext},P,G) = \min \sum_{v\in V_E} x(v) \tag{5}$$
$$\text{subject to} \quad \sum_{v\in e_i} x(v) \geq 1 \qquad e_i \in E_E$$
$$0 \leq x(v) \leq 1 \qquad v \in V_E$$

Let x' be the solution of equation (4) for H^{Del}. Since $V_D \subseteq V_E$, if we let $x'' = x'$ for $v \in V_D$ and $x'' = 0$, for $v \in V_E - V_D$ then $\sum_{v\in e_i} x''(v) \geq 1$, $e_i \in E_E$. Hence we have x' is greater or equal to the solution of equation (5). Hence we obtain the inequality (2) in Theorem 2.

Let x^* be the solution of equation (4) for H^{Ext}. After *Mer*, for $1 \leq i \leq l$, $1 \leq j \leq n_i$, edge $e^*_{i,j}$ corresponds to $f'_{i,j}$, and the image of z under $f'_{i,j}$ is $f'_{i,j}(z)$ in G. We replace $z_{i,j}$ by $f'_{i,j}(z)$ in edge $e^*_{i,j}$ to get $e'_{i,j}$. Let $A = \{f'_{i,j}(z) : 1 \leq i \leq l, 1 \leq j \leq n_i\}$, for $u \in A$, and let B_u be the collection of all vertices that are replaced by u, and $u^* = \max\{x^*(v) : v \in B_u \cup u\}$. For $v \in V_P$ and , define $x^{**}(v) = u^*$ for $v \in A$ and $x^{**}(v) = x^*(v)$ for $V_P - A$. Hence x^{**} satisfies all inequalities in (4), and $\sum_{v\in V_E} x^*(v) \geq \sum_{v\in V_P} x^{**}(v)$ $\min \sum_{v\in V_E} x(v)$. It gives rise to $\sigma(H^{Ext},P,G) \geq \sigma(H_P,P,G)$ Therefore the inequality (3) in Theorem 2 is satisfied. Now we obtain the anti-monotonicity of $RMVC(p,G)$. □

Likewise, the integrability conditions of maximum independent edge set problem can be relaxed to a linear programming formulation and another polynomial-time support.

Definition 19. *Given a pattern p and a data graph G, in hypergraph framework, for any $H \in \mathcal{H}(p)$, where $V = \{v_i\}_{1\leq i\leq n}$ and $E = \{e_i\}_{1\leq i\leq m}$, a function $\sigma(H,p,G)$ is defined as the solution of linear programming relaxation of*

$$\sigma(H,p,G) = \max \sum_{e\in E} y(e) \tag{6}$$
$$\text{subject to} \quad \sum_{e \text{ contains } v_i} y(e) \leq 1 \qquad \forall v_i \in V$$
$$0 \leq y(e) \leq 1 \qquad \forall e \in E.$$

*the **polynomial-time MIES** support measure of pattern p in graph G is defined as*

$$\sigma_{RMIES}(p,G) = \max \sum_{e\in E_p} y(e) \tag{7}$$
$$\text{subject to} \quad \sum_{e \text{ contains } v_i} y(e) \leq 1 \qquad \forall v_i \in V_p$$
$$0 \leq y(e) \leq 1 \qquad \forall e \in E_p.$$

Theorem 6. *The RMIES support measure is anti-monotonic.*

Proof. After *Del*, $H^{Del} = (V_D, E_D)$, we have $H^{Del} \subseteq H_p$ and $E_D \subseteq E_p$. Let y^* be the function that achieves $\sigma(H^{Del},p,G)$,

$$\sigma(H^{Del},p,G) = \max \sum_{e\in E_D} y(e) \tag{8}$$
$$\text{subject to} \quad \sum_{e \text{ contains } v_i} y(e) \leq 1 \qquad \forall v_i \in V_D$$
$$0 \leq y(e) \leq 1 \qquad \forall e \in E_D.$$

Since $E_D \subseteq E_p$, if we let $y^{**}(e) = y^*(e)$ for $e \in E_D$ and $y^{**}(e) = 0$, for $e \in E_p - E_D$ then $\sum_{e \text{ contains } v_i} y^{**}(e) \leq 1$, $v_i \in V_p$. Thus we have $\sigma_{RMIES}(p,G) \geq y^*$. Hence the first inequality in (1) of Theorem 2 is satisfied. In scenario (i) because $V_p = V_P$, it is obviously that the second inequality in (1) of Theorem 2 is satisfied.

After *Ext*, because any edge in $H^{Ext} = (V_E, E_E)$ contains an edge in H^{Del} as a subset, we have $V_D \subseteq V_E$, and for $e' \in E_E$, there exists only one $e \in E_D$ such that $e \subseteq e'$. We let y' be the function that achieves $\sigma(H^{Ext},p,G)$, then

$$\sigma(H^{Ext},p,G) = \max \sum_{e\in E_E} y(e) \tag{9}$$
$$\text{subject to} \quad \sum_{e \text{ contains } v_i} y(e) \leq 1 \qquad \forall v_i \in V_E$$
$$0 \leq y(e) \leq 1 \qquad \forall e \in E_E.$$

Since $e' \in E_E$, there exists only one $e \in E_D$ such that $e \subseteq e'$, if we let $y''(e) = \sum_{e\subset e'} y'(e')$ for $e \in E_D$ then $\sum_{e \text{ contains } v_i} y''(e) \leq 1$, $v_i \in V_D$. Hence we have $\sigma(H^{Del},p,G)$ is greater or equal to the solution of equation (9). Hence we obtain the inequality (2) in Theorem 2.

Let y^* be the function that achieves $\sigma(H_P, P, G)$,

$$\sigma(H_P, P, G) = \max \sum_{e \in E_P} y(e) \tag{10}$$

$$\text{subject to} \sum_{e \text{ contains } v_i} y(e) \leq 1 \qquad \forall v_i \in V_P$$

$$0 \leq y(e) \leq 1 \qquad \forall e \in E_P.$$

After Mer, for $1 \leq i \leq l$, $1 \leq j \leq n_i$, edge $e^*_{i,j}$ corresponds to $f'_{i,j}$, and the image of z under $f'_{i,j}$ is $f'_{i,j}(z)$ in G. We replace $z_{i,j}$ by $f'_{i,j}(z)$ in edge $e^*_{i,j}$ to get $e'_{i,j}$. For $z_{i,j}$, we have $\sum_{e \text{ contains } z_{i,j}} y(e) \leq 1$ is $y(e^*_{i,j}) \leq 1$. After replacement there could be more than one edge e that contains $f'_{i,j}(z)$. Hence $\sum_{e \text{ contains } f'_{i,j}(z)} y(e) \leq 1$ implies a smaller solution $y(e)$. This gives rise to $\sigma(H^{Ext}, P, G) \geq \sigma(H_P, P, G)$ Therefore the inequality (3) in Theorem 2 is satisfied. Now we obtain the anti-monotonicity of $RMIES(p, G)$. □

3.2 Discussions

The Del, Ext and Mer operations give us deep insights on support measures. In Fig.2, the example illustrates how H_p transforms into H_P. We think they reveal critical characters of support measures.

- When converting H_p to H_P, Del deletes occurrences that cannot extend. We shall see that the count of edges in H^{Del} is smaller than that of H_p, which coincides with our intuition that **less occurrences will have less frequency count**.
- In scenarios (ii), one edge e can extend to one or multiple edges. The number of edges increases, but support measure should be non-increasing under Ext operation because the extended edges overlap at vertices of e. Another intuition is that they still have the **same "overlap", count should not increase**.
- In scenarios (ii) a new pattern node is added, the extension may result in more "overlap", hence support measure should be non-increasing under Mer operation. Hence support measure is related to "overlap," **the more "overlap" among occurrences, the smaller the count is**.

Compared with overlap-graph framework sufficient and necessary condition, our sufficient condition is straightforward and intuitive. In the overlap-graph framework, clique traction and edge removal operations handle overlap between occurrences while in hypergraph framework we have Ext, and Mer handles overlap and partial overlap between occurrences. In order to transform overlap graphs, a five-step algorithm is designed. Our method of transformation from H_p to H_P only involves three steps. Ext extends occurrence without "overlap" at new vertices. Mer operation deals with the "overlap" later. We shall point out that occurrence hypergraph as uniform hypergraph is not a general hypergraph. A k-uniform hypergraph framework is able to show that MVC measure is k-approximable, it also shows that any edge cannot intersect with more than k mutually independent edges. Hence the technique used in overlap graph to prove the necessary condition of add vertex $add()$ operation in [25] cannot be applied here.

In addition to the results in Section 3.1, we can also show that invalid measures such as number of occurrences violate some of the sufficient conditions. For example, the number of occurrences is not anti-monotonic because it increases under Ext operation –

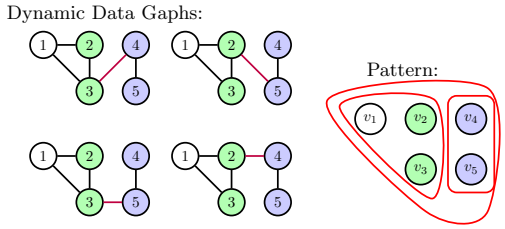

Figure 4: Four possible ways interactions among five molecules (nodes) can occur in a particular molecular interaction network. Connections between the triangular subgraph (v_1, v_2, v_3) and the one-edge subgraph (v_4, v_5) could change over time. To interpret such patterns, hypergraph models are adopted [22] as shown on the right: two node subsets are formed to capture the stable interactions

one occurrence of pattern p can extend to multiple occurrences of its superpattern P, e.g., f_1 extends to $f'_{1,1}, f'_{1,2}, f'_{1,3}$ as shown in Fig. 1. With such, it becomes natural to ask whether the sufficient conditions we developed are also necessary. Our conjecture is that they are also necessary, yet it is still an open problem whether we can prove necessary conditions for monotonicity in the hypergraph framework.

That being said, in this paper we focus on current sufficient condition and in the next section, we shall show how to use this sufficient condition as guidance to construct new anti-monotonic support measures.

4 NEW POLYNOMIAL SUPPORT MEASURES

After obtaining sufficient conditions for anti-monotonicity of support measures, in this section we show how we utilize it for finding new support measures that are efficient and effective.

4.1 Generalized Linear Time Support Measures

One aspect of studying support measures is that users could emphasize application/data-specific information (e.g., pattern structure, dataset features) in developing their graph mining applications. Therefore, it is reasonable to include such information in developing support measures suitable for various needs. One example (Fig. 4) comes from the field of systems biology, where hypergraphs are used to capture the uncertainty (dynamic changes) that is inherent in gene-gene networks [22]. In such cases, researchers should have the option of forming appropriate node subsets (instead of a fixed one such as the transitive node subset used in MI) to obtain reasonable pattern frequency. After investigating the MI measure with respect to the three operations in Theorem 2, our conclusion is that we can design various linear-time support measures in a general framework that ensures anti-monotonicity.

We first revisit the MI measure by mapping the design rationale to the proposed sufficient conditions. Note that MI uses the minimum count of node set images under occurrences as the frequency count - the function $c(T)$ returns the number of set T's images under occurrences. For $H \in \mathcal{H}(p)$, we can define $c_H(T)$ as the number of images of T under all edges in H. It is non-increasing under Del which deletes edges and images of set T. It is also non-increasing under Mer which merges vertices in images of set T. Therefore,

the count function $c(T)$ makes MI anti-monotonic under *Del* and *Mer*. However, itself does not guarantee anti-monotonicity under *Ext* (this is why the number of occurrences or instances cannot be used as support measures). Since occurrences can usually extend to multiple superpattern occurrences, if we let MI take the minimum among counts of node set T's images, and any T of pattern p should be considered when calculating its superpattern P support. In such a way, we ensure the support is non-increasing under *Ext*. Following the above reasoning, we can generalize MI-flavored support measure as follows.

Definition 20. *Given a pattern p and a data graph G, in hypergraph framework, for any $H \in \mathcal{H}(p)$, a function $\sigma(H, p, G)$ is defined as $\sigma(H, p, G) = \min_{T \in \mathcal{T}_p} c_H(T)$, and a **general minimum instance based (GMI)** support of p in G, as a restriction of σ on H_p, is defined as*

$$GMI(p, G) = \min_{T \in \mathcal{T}_p} c_{H_p}(T), \tag{11}$$

where $\mathcal{T}_p$ is a collection of node subsets in pattern p, $c_H(T)$ and $c_{H_p}(T)$ are the counts of images of T under edges in H and H_p, respectively.

Theorem 7. ***(Sufficient condition for GMI)** GMI is anti-monotonic if the following condition*

$$\mathcal{T}_p \subseteq \mathcal{T}_P, \quad \text{for any } p \subseteq P, \tag{12}$$

is satisfied.

Proof. After *Del*, $H^{Del} \subseteq H_p$, H^{Del} has fewer edges than H_p thus for any $T \in \mathcal{T}_p$ we have $c_{H_p}(T) \geq c_{H^{Del}}(T)$. Hence the first inequality in (1) of Theorem 2 is satisfied. In scenario (i) because $\mathcal{T}_p \subseteq \mathcal{T}_P$, we get $\min_{T \in \mathcal{T}_p} c_{H^{Del}}(T) \geq \min_{T \in \mathcal{T}_P} c_{H^{Del}}(T)$. Hence the second inequality in (1) of Theorem 2 is satisfied.

After *Ext*, for any $T \in \mathcal{T}_p$, $c_{H^{Del}}(T) = c_{H^{Ext}}(T)$. Because $\mathcal{T}_p \subseteq \mathcal{T}_P$ we have $\min_{T \in \mathcal{T}_p} c_{H^{Del}}(T) \geq \min_{T \in \mathcal{T}_P} c_{H^{Ext}}(T)$, which implies the inequality (2) in Theorem 2.

After *Mer*, for any $T \in \mathcal{T}_P$ that does not contain z, their images remain the same hence $c_{H^{Ext}}(T) = c_{H_P}(T)$. For $T \in \mathcal{T}_P$ that contains z, $c_{H^{Ext}}(T) \geq c_{H_P}(T)$. Thus we have $\min_{T \in \mathcal{T}_P} c_{H^{Ext}}(T) \geq \min_{T \in \mathcal{T}_P} c_{H_P}(T)$. Therefore the inequality (3) in Theorem 2 is satisfied. Now we obtain the anti-monotonicity of $GMI(p, G)$. □

Going back to the example in Figure 4, for any superpattern P of triangular subgraph (v_1, v_2, v_3), according to Theorem 7, we can have $T = \{v_1, v_2, v_3\}$ included in $\mathcal{T}_P$ and make sure $\mathcal{T}_p \subseteq \mathcal{T}_P$ to design a GMI that is suitable for counting these types of patterns. Such techniques are meaningful in many other applications (e.g., social network, protein structures, and image classification) that deal with noisy data, dynamic dataset, or graphs generated from fast approximate algorithms [8, 9, 12, 18].

To sum up, we extend the minimum image approach into hypergraph subedge approach to develop new MI variants. We believe that hypergraph subedge approach is a valuable method for such explorations, because it is flexible and can be tailored for pattern nodes and other features.

4.2 New Low-Order Polynomial Time Support

Within the hypergraph framework we have defined/redefined a series of support measures including the MIS/MIES and MNI [20]. MIS and MNI represent the two main categories of support measures yet they stand on far ends of computing efficiency and overestimation of pattern frequency. Without considering application-specific requirements, users prefer the intuitive MIS that returns the number of independent occurrences (smallest counts among all measures mentioned in this paper). However, it is not practical to compute the NP-hard MIS for even moderately sized inputs. On contrary to that, the MNI has linear complexity but can return an arbitrarily large count for a pattern [10]. According to Eq. (1), the MVC and MI we developed fill in between MIS and MNI, providing more options for the efficiency/intuitiveness tradeoff. Linear programming relaxations of MIS and MVC further reduce the frequency counts and only require polynomial time to compute. Our goal is to design a highly-scalable support measure that returns frequency counts closer to MIS as compared to the relaxed version of MVC (RMVC). Note that RMVC is still not scalable - the best solution is an interior-point-method algorithm which runs at $O(n^{3.5}L)$ complexity, where n is the number of vertices and L is the number of bits in the input [17].

Our new measure is derived from an in-depth investigation of the sufficient conditions together with GMI and MIS, MIES, MVC measures. Revisit the analysis of GMI = $\min_{T \in \mathcal{T}_p} c_{H_p}(T)$, in which count function $c(T)$ and the min function ensure GMI is non-increasing under operations *Del* and *Mer*; the design of $\mathcal{T}_P$ makes GMI non-increasing under *Ext*. We can find a function whose count is closer to MIES than $c(T)$. Consider the fact that MIES counts only independent occurrences and $MIES(T) \leq MVC(T) \leq c(T)$, we shall choose MIES as the starting point. This is important, because it means that the subedge version of MIES will most likely be bounded by GMI/MI/MNI which are subedge version of MVC.

In the next step, we need to design $\mathcal{T}_p$. The maximum independent edge set problem (i.e., maximum matching problem) under k-uniform hypergraphs is NP-hard in general. However, under k = 2, the occurrence hypergraph edges contain only two vertices, the problem of counting MIES is equivalent to the famous maximum matching problem which is solvable in polynomial time [21]. Note that for $k \geq 3$, the problem becomes NP-hard again [16]. Our solution is to break the hypergraph edges into two-vertex subedges to reduce complexity. Therefore we choose $\mathcal{T}_p = \{T\}$, where $T = \{u, v\}, u, v \in V_p$, and this meets the requirement of Theorem 7. As a result, the designed measure should be non-increasing under *Ext*. In addition, the reason we use the set of all node pairs $\{u, v\}$ instead of E_p is that sometimes occurrences can overlap on vertices that are not connected by any pattern edges (see Fig. 5).

Following the above ideas, we propose a maximum independent subedge set (MISS) measure. This new support measure is also supported by the following observation: if two hypergraph edges are independent to each other (i.e., no vertex intersections), any subedges (subsets) within these two edges should also be independent. We hereby develop a formal definition of the MISS measure and conduct rigorous analysis on its features.

Definition 21. *Given a pattern p and a data graph G, in hypergraph framework, for any $H \in \mathcal{H}(p)$, a function $\sigma(H, p, G)$ is defined as $\sigma(H, p, G) = \min_{w \in W_p} d_H(w)$, and the **maximum independent subedge set (MISS) support measure**, as a restriction*

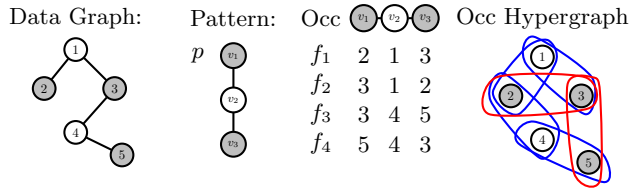

Figure 5: An example of FSM problem, in which MISS = MIS = 1, and any GMI ≥ 2

of σ on H_p, is defined as

$$\sigma_{MISS}(p, G) = \min_{w \in W_p} d_{H_p}(w), \tag{13}$$

where $W_p = \{\{u, v\} : u, v \in V_p\}$, $d_{H_p}(w)$ and $d_H(w)$ are the size of maximum independent edge set of images of w under edges of H_P and H respectively.

An example of MISS is given in Fig.5. The red edges are images of node pair $\{v_1, v_3\}$ (which is not a pattern edge), blue edges represent images of node pairs (pattern edges) $\{v_1, v_2\}$ and $\{v_2, v_3\}$. $d(\{v_1, v_3\}) = 1$. $d(\{v_1, v_2\}) = d(\{v_2, v_3\}) = 2$, thus MISS = min{1, 2, 2} = 1. On the other hand, we go through all combinations of node subsets, all their image counts ≥ 2, thus any GMI is at least 2.

Anti-monotonicity: We shall show that this new support measure satisfies the anti-monotonicity requirement.

Theorem 8. *The MISS support measure is anti-monotonic.*

Proof. After *Del*, for $w \in W_p$, we use $\{I_H(w)\}$ and $\{M_H(w)\}$ to denote the collections of independent edge sets and *maximum* independent edge sets of images of w under edges in $H \in \mathcal{H}(p)$, respectively. Because $H^{Del} \subseteq H_p$, for any $w \in W_p$, we conclude that any $M_{H^{Del}}(w)$ is also an $I_{H_p}(w)$. Hence $d_{H_p}(w) \geq d_{H^{Del}}(w)$. Now we obtain the first inequality in (1) of Theorem 2. In scenario (i) because $W_p = W_P$, we have $\min_{w \in W_p} d_{H^{Del}}(w) \geq \min_{w \in W_P} d_{H^{Del}}(w)$. Hence the second inequality in (1) of Theorem 2 is satisfied.

After *Ext*, for any $w \in W_p$, we get $d_{H^{Del}}(w) = d_{H^{Ext}}(w)$. Since $W_p \subseteq W_P$, we have $\min_{w \in W_p} d_{H^{Del}}(w) \geq \min_{w \in W_P} d_{H^{Ext}}(w)$, which implies that the inequality (2) in Theorem 2 is satisfied.

After *Mer*, for any $w \in W_P$ that does not contain z, $d_{H^{Ext}}(w) = d_{H_P}(w)$. For $w \in W_P$ that contains z, $d_{H^{Ext}}(w) \geq d_{H_P}(w)$. Thus we have $\min_{w \in W_P} d_{H^{Ext}}(w) \geq \min_{w \in W_P} d_{H_P}(w)$, and it gives rise to inequality (3) in Theorem 2.

To conclude, the sufficient conditions of Theorem 2 are satisfied, MISS is anti-monotonic.

Another proof that follows similar intuition and ideas behind Theorem 2 but does not explicitly depend on the sufficient conditions is given below:

Given pattern $p = (V_p, E_p)$ and its superpattern $P = (V_P, E_P)$ in data graph G. Assume that there are m occurrences $\{f_i\}_{1 \leq i \leq m}$ of pattern p, and they can be extended to l occurrences $\{f'_i\}_{1 \leq i \leq l}$ of P. Since $V_p \subseteq V_P$, we have $W_p \subseteq W_P$ by definition. For any $w \in W_p \subseteq W_P$, because each f'_i is extension of some f_j, we have $\{f'_i(w)\}_{1 \leq i \leq l} \subseteq \{f_i(w)\}_{1 \leq i \leq m}$. Hence if we denote the cardinalities of maximum independent edge set of $\{f'_i(w)\}_{1 \leq i \leq l}$ and $\{f_i(w)\}_{1 \leq i \leq m}$ as $d_{f'}(w)$ and $d_f(w)$ respectively, we have $d_{f'}(w) \leq d_f(w)$.

$$\begin{aligned}
\sigma_{MISS}(P, G) &= \min_{w \in W_P} d_{f'}(w) && \text{by definition} \\
&\leq \min_{w \in W_p} d_{f'}(w) && \text{because of } W_p \subseteq W_P \\
&\leq \min_{w \in W_p} d_f(w) \\
&= \sigma_{MISS}(p, G) && \text{by definition.}
\end{aligned}$$

Hence we obtain the anti-monotonicity of MISS. □

Computational Complexity: As mentioned earlier, we expect MISS to be polynomial-time computable, here is a formal proof.

Theorem 9. *The MISS support measure is of polynomial time complexity in the number of pattern occurrences.*

Proof. By definition, $\sigma_{MISS}(p, G) = \min_{w \in W_p} d_{H_p}(w)$. Assume p has m occurrences $\{f_i\}_{1 \leq i \leq m}$. For $w \in W_p$, the images of w are $\{f_i(w)\}_{1 \leq i \leq m}$, because each $f_i(w)$ contains two vertices, they can be viewed as general graph edges. According to [21], there exists fast $O(\sqrt{n}m)$ algorithm for finding maximum independent edge set (maximum matching) in general graphs, where n is the number of vertices and m is the number of edges in general graphs. For p that has k nodes, and H_p that has n vertices and m hyperedges, there are $\frac{k(k-1)}{2}$ node pairs in W_p. Thus the complexity of finding $d_{H_p}(w)$ is $O(\sqrt{n}m)$ for each $w \in W_p$. The number of $w \in W_p$ is fixed (in terms of k), hence MISS is also polynomial-time computable. □

An important note is, MISS can be implemented in $O(\sqrt{n}m)$ time, as compared to the $O(n^{3.5}L)$ time needed for computing RMVC.

Bounding Theorems: We study the frequency counts returned by MISS in comparison to other support measures. We first compare MISS with MIES, which is equivalent to MIS.

It is clear that if two hypergraph edges e and e' do not overlap then subedges (subsets) of e do not overlap with that of e'. Hence MIES is a lower bound of MISS, that is, $\sigma_{MIES}(p, G) \leq \sigma_{MISS}(p, G)$. The MISS measure is derived from maximum independent edge set, while GMI, MI and MNI are rooted at the minimum image based idea. This confirms the finding in our previous work [20] that MIES/MIS is the natural lower bound of support measures defined within the hypergraph framework. On the other hand, it is critical to study whether MISS is bounded by GMI support measures - the value of MISS will be questionable if it finds larger counts than GMI as the latter is of linear complexity.

Theorem 10. *For pattern p in data graph G, we have*

$$\sigma_{MISS}(p, G) \leq \sigma_{GMI}(p, G).$$

Proof. Assume that pattern p has m occurrences $\{f_i\}_{1 \leq i \leq m}$ in data graph G. We shall show that this inequality is true for any $\sigma_{GMI}(p, G) = \min_{T \in \mathcal{T}} c_{H_p}(T)$ with $\mathcal{T} = \{T\}$, where T is a subset of V_p.

Case I: If T contains only one node u. Since in this paper we consider patterns containing more than one node, we can find another node v to form a node pair $w' = \{u, v\}$. Assume a maximum independent edge set of w' images under all occurrences is $I = \{f_i(w')\}_{1 \leq i \leq l} = \{\{f_i(u), f_i(v)\}\}_{1 \leq i \leq l}$, where $l \leq m$. It follows that

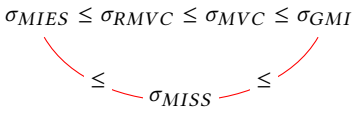

Figure 6: Bounding theorems among relevant measures

$\{f_i(u)\}_{1 \leq i \leq l}$ are a set of distinct vertices. Hence $d_{H_p}(w') \leq c_{H_p}(T)$, and $\sigma_{MISS}(p, G) = \min_{w \in W_p} d_{H_p}(w) \leq d_{H_p}(w') \leq c(T)$.

Case II: If T contains two or more nodes, for any two nodes $u, v \in V_p$, let $w' = \{u, v\}$. Assume a maximum independent edge set of their images is $I = \{f_i(w')\}_{1 \leq i \leq l}$, where $l \leq m$. Because $f_i(w') \subseteq f_i(T)$, $1 \leq i \leq l$, and $f_i(w') \cap f_j(w') = \emptyset$, for $i \neq j$, we obtain that $f_i(T)$ and $f_j(T)$ are pairwise different for $1 \leq i, j \leq l$ and $i \neq j$. Hence, we have

$$\sigma_{MISS}(p, G) = \min_{w \in W_p} d_{H_p}(w) \leq d_{H_p}(w') = l = |\{f_i(T)\}_{1 \leq i \leq l}|$$
$$\leq |\{f_i(T)\}_{1 \leq i \leq m}| = c_{H_p}(T).$$

Putting all together, we have $\sigma_{MISS}(p, G) \leq c_{H_p}(T)$ for any $T \in \mathcal{T}$. To conclude, $\sigma_{MISS}(p, G) \leq \min_{T \in \mathcal{T}} c_{H_p}(T) = \sigma_{GMI}(p, G)$. □

Theorem 10 tells us that no matter how the node subsets in a pattern are formed in a GMI, the MISS count is always smaller. This also means that MISS provides a new level of efficiency-intuitiveness trade-off in the domain of anti-monotonic support measures - MISS will not overestimate pattern frequency as much as MI does under the cost of longer computing time. With such results, the current spectrum of support counts with all measures mentioned in this paper is shown in Fig. 6.

Note that there is no bound between MISS and RMVC/MVC. However, MISS is found to be very close to MIS (thus smaller than RMVC) in most cases. First, in special cases where the pattern only contains one edge (two nodes), MISS values are found to be identical to MIS/MIES and smaller than RMVC.

Theorem 11. *When pattern p is a one-edge pattern, MISS is identical to MIS, hence MISS ≤ MVC.*

Proof. When pattern p is a one-edge pattern, say the only edge is $w' = \{u, v\}$, then $\sigma_{MISS}(p, G) = \min_{w \in W_p} d_{H_p}(w) = d_{H_p}(w') = \sigma_{MIES}(p, G)$. Because $\sigma_{MIES}(p, G) \leq \sigma_{RMVC}(p, G)$, we conclude that $\sigma_{MISS}(p, G) \leq \sigma_{RMVC}(p, G)$ □

Second, MISS are smaller than RMVC in most scenarios, as shown by experiments running on real graph datasets (Section 5).

In summary, we believe the discovery of MISS is a major milestone in the exploration of support measures: MISS is the first non-relaxation support measure that is close to MIS yet requires only (low-order) polynomial time to compute.

5 EXPERIMENTS

As most of the findings in this paper are supported by rigorous proof, our experimental evaluations focus on the actual performance of MISS in comparison to other support measures, especially the RMVC measure that provides the best tradeoff between efficiency and frequency counts so far. MISS is of even lower complexity than RMVC but there exists no bounds between the frequency

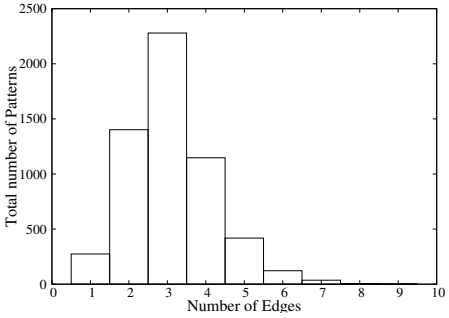

Figure 7: Distribution of the sizes (in terms of number of edges in a pattern) of patterns studied in all datasets

Table 1: Graph datasets used in our experiments

Dataset	Total Edges	Total Vertices	Description
Chicago [13]	1,467	1,298	Transportation
FAA [1]	2,615	1,226	Routes Database
Stelzl [23]	6,207	1,706	Protein network
Figeys [3]	6,452	2,239	Protein network
Vidal [4]	6,726	3,133	Protein network
Chess [2]	65,053	7,301	Results of games
Brightkite [5]	214,078	58,228	Friendship Graph
Facebook [6]	817,035	63,731	Friendship Graph

Table 2: Runtime parameters for processing the datasets in our experiments

Dataset	Frequency Cutoff	Total Patterns studied
Chicago	5	133
FAA	10	1,634
Stelzl	20	143
Figeys	40	230
Vidal	25	2323
Chess	270	562
Brightkite	1000	304
Facebook	2000	164

counts returned by those two. For that, we implemented a framework to compute relevant support measures mentioned in this paper. As input the framework takes a pattern, and a list of its occurrences in the data graph in the form of DFScode [29], which represents the DFS lexicographic order of the pattern. We obtain occurrences by using DistGraph Framework [24]. We implemented the following support measures: MNI, MI, MISS, RMVC, and MIS. In particular, we implemented RMVC by using the Interior Point Optimizer (IPOPT) [27], which is one of the most efficient codes of interior-point algorithms. We run all of our experiments in a workstation running Linux (Ubuntu 14.04 LTS) with an Intel Xeon E5-2640 v2 CPU and 64GB of DDR3 1333-MHz memory. Our source code is available on GitHub https://github.com/napath-pitaksirianan/GraphMining.

Data Graphs: We use 8 different datasets for our experiments as shown on Table 1. All datasets are collected from real-world applications and acquired from the well-known KONECT [7] website.

Patterns: To generate patterns, we set a frequency cutoff according to MNI (being the largest support measure) for the data graph. We

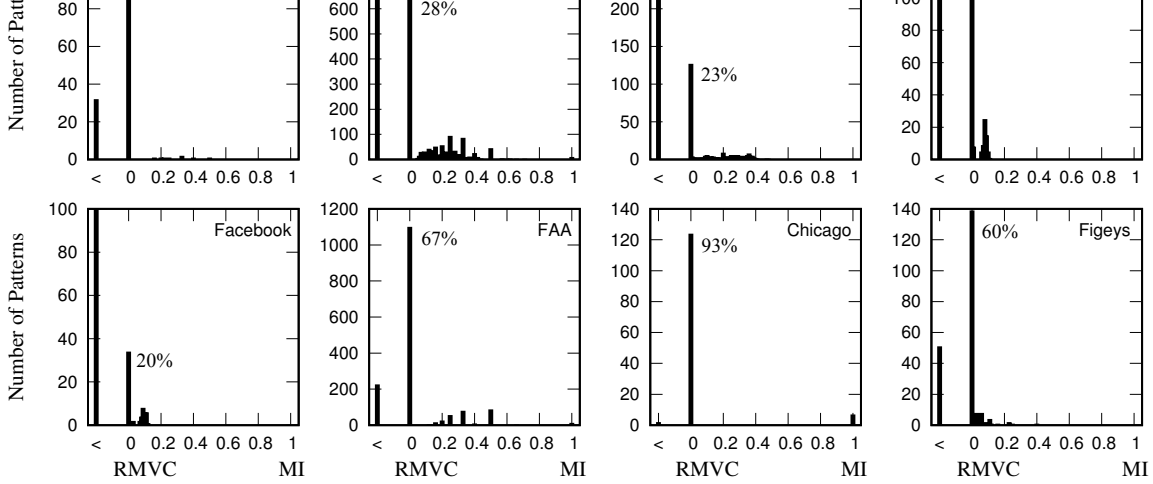

Figure 8: Distribution of the relative values of MISS to MI and RMVC for all patterns visited in different datasets

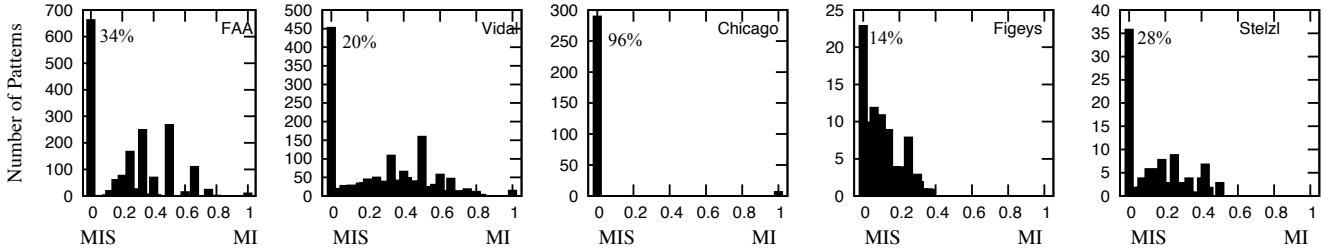

Figure 9: Distribution of the relative values of MISS to MI and MIS for all patterns visited in different datasets

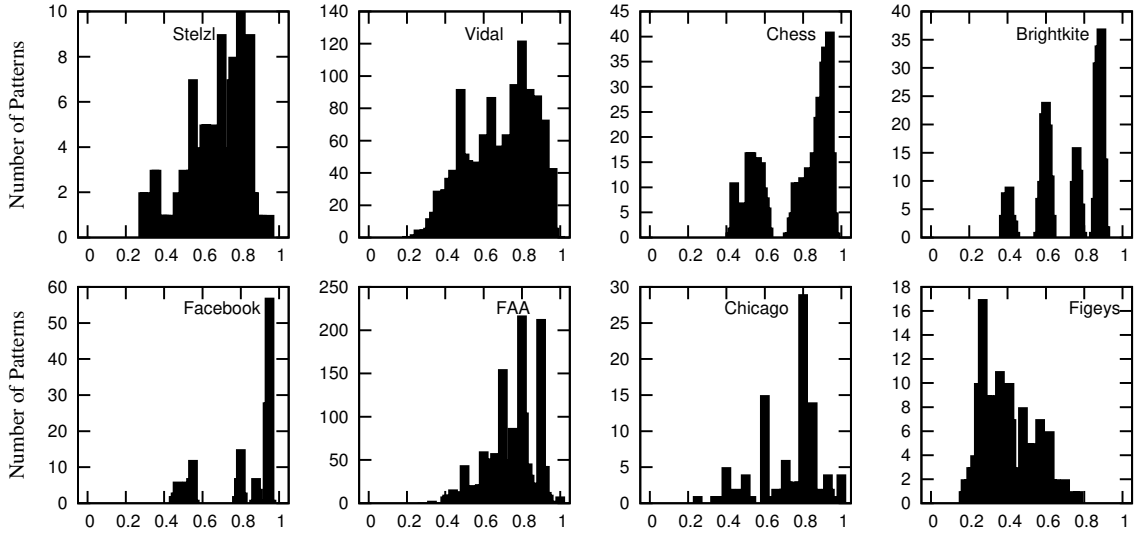

Figure 10: Distribution of the relative values of MISS to MI for all patterns visited in different datasets

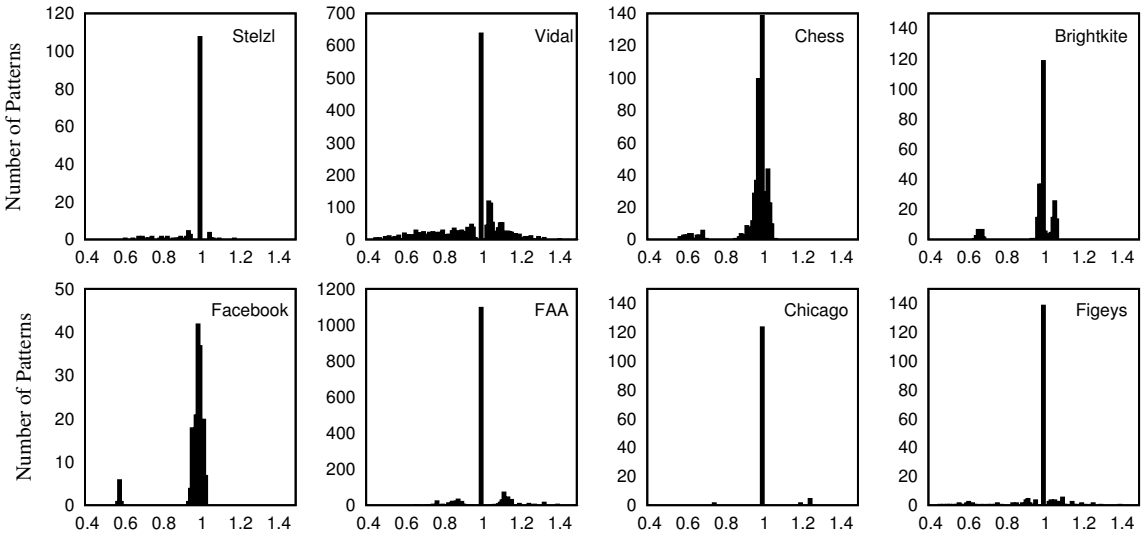

Figure 11: Distribution of the relative values of MISS to RMVC for all patterns visited in different datasets

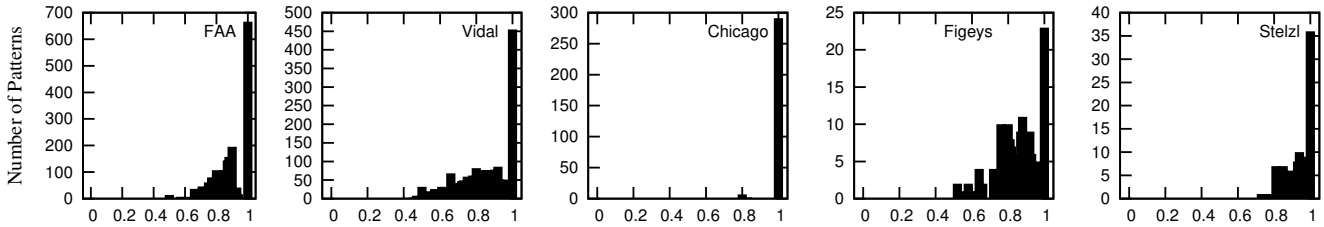

Figure 12: Distribution of the relative values of MIS to MISS for all patterns visited in different datasets

report four measures (i.e., MI, MISS, RMVC, and MIS) for EVERY pattern in the dataset with an MNI value higher than that cutoff. The cutoff is set to a relatively small number to ensure we study a large number of patterns therefore the workload is not biased towards particular patterns. Such cutoff numbers as well as the sizes of patterns we study in our experiments are shown in Table 2 and Figure 7.

5.1 Experimental Results

For comparison purposes, we present counts returned by MISS in relation to those returned by other measures as follows:

$$R = \frac{\sigma_{MISS} - \sigma_A}{\sigma_B - \sigma_A} \tag{14}$$

where A is another measure whose counts serve as the lower bound, and B the higher bound. Thus, an R value close to 1 (0) means that MISS value is close to that of B (A). We present the results for all frequent patterns we encountered in each dataset.

We first show the distribution of the R values among all considered patterns in relation to that of MI (higher bound) and RMVC (lower bound) in Figure 8. First of all, there is no case in which MISS is larger than MI, verifying Theorem 6. For a significant portion of the patterns, MISS is the same as (i.e., $R = 0$) RMVC. Furthermore, we see even more cases in which MISS is smaller than RMVC, as shown by a single column marked with "<" in Figure 8. For the cases in which MISS is larger than RMVC, the R value is small, with almost all of them under 0.5. We also show the percentage of pattern that MISS value is the same as MI value in the figure.

The relative values of MISS to MI and MIS are also plotted in the same way in Figure 9. Due to the exponential running time of MIS, we only obtained MIS values in 5 datasets (i.e., FAA, Vial, Chicago, Figeys, and Stelzl). According to Figure 9, for most of patterns in the Chicago, Figeys, and Stelzl datasets, MISS is very close to MIS while for FAA and Vial datasets, the R value spread out in a larger range. However, in a large portion of the patterns we observed that MISS equals to MIS (i.e., $R = 0$). Again, We show the percentage of pattern that MISS value is the same as MIS value in the figure.

Figures 8 and 9 show the relative values in the range of two baseline measures. While it demonstrates how close MISS is to either baseline, there is a lack of insights on how the absolute values of different measures differ. For that purpose, we also present the relative values of MISS to a target measure directly as follow:

$$R' = \frac{\sigma_A}{\sigma_B} \tag{15}$$

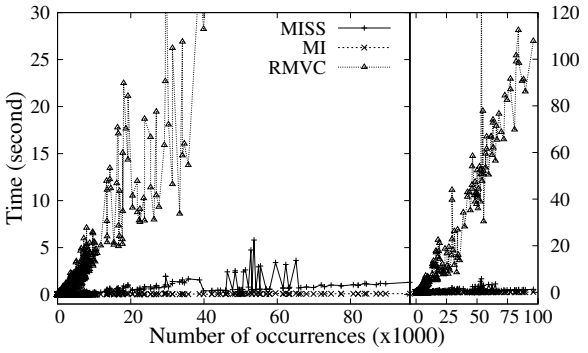

Figure 13: Time to compute different support measures

The R' value shows the relative value of A to B. R' can represent how far the values are different in terms of their absolute values.

We show the R' values between MISS to MI and MISS to RMVC in Figures 10 and 11, respectively. First, as shown in Figure 10, it is clear that MISS is not only smaller than MI in all cases, it differs from MI by a large margin (i.e., up to 80%, and 20% on average among all datasets). On the other hand, MISS is similar to RMVC in most cases, but there are more cases in which MISS is smaller than those in which RMVC is smaller. There are significant number of cases in which the MISS value is less than half of RMVC. In the rare cases of MISS > RMVC, the MISS value is never more than 40% larger than RMVC.

We also plot the R' value of MIS to MISS in Figure 12. As a general observation, the R' value is skewed towards larger values in the range $[0, 1.0]$, meaning that for most cases, MISS is not much larger than MIS. Note there are also many cases in which MIS equals MISS (i.e., $R' = 1$), especially in the Chicago dataset – this is the same as shown in Figure 9. Only in extremely rare cases MISS is twice as large as MIS (i.e., $R' = 0.5$). In short, all such results demonstrate that MISS is very close to MIS, which is generally regarded as the most intuitive measure (without application-specific considerations). As MIS/MIES represents the lower bound of all hypergraph-based measures, it is inevitable to have a gap between MIS/MIES and others. Therefore, our goal is to minimize that gap, and we have achieved that goal by showing a much smaller gap than that between MIS/MIES and RMVC. Here we have to again mention the major difficulty in dealing with MIS: we only obtained the MIS for a subset of the patterns visited due to its extremely long computing time.

Computational Efficiency: We also report the computational time of generating the support measures in Figure 13, with a single point representing a single pattern. We report time for all patterns with up to 100,000 occurrences each. We do not record the time for computing the pattern occurrences, as that is the same for all measures. As expected, the time for generating MISS is much shorter than that for RMVC. The difference between MISS and the linear-time MI in terms of running time is insignificant.

In summary, MISS outperforms RMVC in computational time and returns frequency counts that are closer to the intuitive MIS.

6 CONCLUSIONS AND FUTURE WORK

In this paper, we present sufficient conditions for support measures within a hypergraph framework. With insights gained from such sufficient conditions, we develop new time-efficient support measures and provide theoretical and experimental evaluations for them. In particular, we introduce and analyze GMI, a general form for all minimum instance based support measures. We point out the flexibility of this support for practical use cases in counting frequent patterns. We also introduce the first non-relaxation polynomial-time frequent support measure named MISS which fills the gap between the NP-hard MIES/MIS and linear-time computable MI and MNI measures. For that, we believe this is a significant breakthrough in the topic of support measures in FSM. Experiments on real-world datasets show the effectiveness of the new MISS support measure. By checking a large number of patterns in 8 different real graph databases, we found that MISS significantly outperforms the best-known polynomial measure RMVC in running time and returns counts closer to MIS.

There are still a lot of opportunities for future research under the hypergraph framework. Some ideas include: (1) find and prove necessary conditions for support measures; (2) more established graph combinatorics theorems can be incorporated into existing efforts in designing support measures; (3) methods other than hypergraph subedges can be developed for reducing time-complexity of current NP-hard support measures; (4) if certain support measures are absolutely needed while they have NP-hard or high-order polynomial time complexity, parallel computing becomes a viable solution; (5) The increasing interest in web and social networks has heightened the need for future research on support counting techniques that handle dynamic and streaming graph data sets.

ACKNOWLEDGMENTS

This work is supported by an award (IIS-1253980) from the National Science Foundation (NSF) of U.S.A.. Equipments used in this research were acquired via NSF grant CNS-1513126.

REFERENCES

[1] 2016. Air traffic control network dataset – KONECT. (Sept. 2016). http://konect.uni-koblenz.de/networks/maayan-faa
[2] 2016. Chess network dataset – KONECT. (Sept. 2016). http://konect.uni-koblenz.de/networks/chess
[3] 2016. Human protein (Figeys) network dataset – KONECT. (Sept. 2016). http://konect.uni-koblenz.de/networks/maayan-figeys
[4] 2016. Human protein (Vidal) network dataset – KONECT. (Sept. 2016). http://konect.uni-koblenz.de/networks/maayan-vidal
[5] 2017. Brightkite network dataset – KONECT. (April 2017). http://konect.uni-koblenz.de/networks/loc-brightkite_edges
[6] 2017. Facebook friendships network dataset – KONECT. (April 2017). http://konect.uni-koblenz.de/networks/facebook-wosn-links
[7] 2018. KONECT. (Feb. 2018). http://konect.uni-koblenz.de/
[8] Niusvel Acosta-Mendoza, Andrés Gago-Alonso, and José E Medina-Pagola. 2012. Frequent approximate subgraphs as features for graph-based image classification. *Knowledge-Based Systems* 27 (2012), 381–392.
[9] Pranay Anchuri, Mohammed J Zaki, Omer Barkol, Shahar Golan, and Moshe Shamy. 2013. Approximate graph mining with label costs. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 518–526.
[10] Björn Bringmann and Siegfried Nijssen. 2008. What is frequent in a single graph?. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 858–863.
[11] Toon Calders, Jan Ramon, and Dries Van Dyck. 2008. Anti-monotonic overlap-graph support measures. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 73–82.

[12] Chen Chen, Xifeng Yan, Feida Zhu, and Jiawei Han. 2007. gapprox: Mining frequent approximate patterns from a massive network. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*. IEEE, 445–450.

[13] R. W. Eash, K. S. Chon, Y. J. Lee, and D. E. Boyce. 1983. Equilibrium Traffic Assignment on an Aggregated Highway Network for Sketch Planning. *Transportation Research Record* 994 (1983), 30–37.

[14] Mathias Fiedler and Christian Borgelt. 2007. Support Computation for Mining Frequent Subgraphs in a Single Graph.. In *MLG*. Citeseer.

[15] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian Suchanek. 2013. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 413–422.

[16] Elad Hazan, Shmuel Safra, and Oded Schwartz. 2006. On the complexity of approximating k-set packing. *computational complexity* 15, 1 (2006), 20–39.

[17] Narendra Karmarkar. 1984. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. ACM, 302–311.

[18] Brian P Kelley, Roded Sharan, Richard M Karp, Taylor Sittler, David E Root, Brent R Stockwell, and Trey Ideker. 2003. Conserved pathways within bacteria and yeast as revealed by global protein network alignment. *Proceedings of the National Academy of Sciences* 100, 20 (2003), 11394–11399.

[19] Michihiro Kuramochi and George Karypis. 2005. Finding frequent patterns in a large sparse graph. *Data mining and knowledge discovery* 11, 3 (2005), 243–271.

[20] Jinghan Meng and Yi-Cheng Tu. 2017. Flexible Support Measures and Search Schemes for Mining Frequent Patterns in a Large Single Graph. In *Proceedings of the 2017 ACM International Conference on Management of Data*. ACM, 34–36.

[21] Silvio Micali and Vijay V Vazirani. 1980. An O (v| v| c| E|) algoithm for finding maximum matching in general graphs. In *Foundations of Computer Science, 1980., 21st Annual Symposium on*. IEEE, 17–27.

[22] Ahsanur Rahman, Christopher L Poirel, David J Badger, Craig Estep, and TM Murali. 2013. Reverse engineering molecular hypergraphs. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 10, 5 (2013), 1113–1124.

[23] U. Stelzl, et al. 2005. A Human Protein–Protein Interaction Network: A Resource for Annotating the Proteome. *Cell* 122 (2005), 957–968.

[24] Nilothpal Talukder and Mohammed J Zaki. 2016. A distributed approach for graph mining in massive networks. *Data Mining and Knowledge Discovery* 30, 5 (2016), 1024–1052.

[25] Natalia Vanetik, Ehud Gudes, and Solomon E. Shimony. 2002. Computing frequent graph patterns from semistructured data. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings*. 458–465.

[26] Natalia Vanetik, Solomon E. Shimony, and Ehud Gudes. 2006. Support measures for graph data. *Data Mining and Knowledge Discovery* 13, 2 (2006), 243–260.

[27] Andreas Wächter and Lorenz T. Biegler. 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106, 1 (01 Mar 2006), 25–57.

[28] Yuyi Wang and Jan Ramon. 2012. An efficiently computable support measure for frequent subgraph pattern mining. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 362–377.

[29] Xifeng Yan and Jiawei Han. 2002. gSpan: Graph-Based Substructure Pattern Mining. In *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan*. 721–724.