Discovering subjectively interesting multigraph patterns

Kapoor, S.; Saxena, D.K.; Leeuwen, M. van

Machine Learning (2020) 109:1669–1696
https://doi.org/10.1007/s10994-020-05873-9

Discovering subjectively interesting multigraph patterns

Sarang Kapoor[1] · Dhish Kumar Saxena[2] · Matthijs van Leeuwen[3]

Received: 18 May 2019 / Revised: 14 October 2019 / Accepted: 13 February 2020 /
Published online: 16 March 2020

Abstract

Over the past decade, network analysis has attracted substantial interest because of its potential to solve many real-world problems. This paper lays the conceptual foundation for an application in aviation, through focusing on the discovery of patterns in *multigraphs* (graphs in which multiple edges can be present between vertices). Our main contributions are twofold. Firstly, we propose a novel *subjective interestingness measure for patterns in both undirected and directed multigraphs*. Though this proposal is inspired by our previous related research for simple graphs (having only single edges), the properties of multigraphs make this transition challenging. Secondly, we propose a greedy algorithm for subjectively interesting pattern mining, and demonstrate its efficacy through several experiments on synthetic and real-world examples. We conclude with a case study in aviation, which demonstrates how the departure from an analyst's prior beliefs captured as subjectively interesting patterns could help improve an analyst's understanding of the data and problem at hand.

Keywords Multigraph · Subjective interestingness · Maximum entropy principle · Exploratory data mining

1 Introduction

Over the past decade, researchers have realised that network analysis can be used to address many real-world problems. Examples include problems related to computer network infrastructure, co-authorship (scientific or other), co-actors (e.g., in movies), transport (road,

Editors: Kee-Eung Kim and Jun Zhu.

✉ Sarang Kapoor
skapoor@cs.iitr.ac.in

Dhish Kumar Saxena
dhish.saxena@me.iitr.ac.in

Matthijs van Leeuwen
m.van.leeuwen@liacs.leidenuniv.nl

[1] Department of Computer Science and Engineering, Indian Institute of Technology, Roorkee, India

[2] Department of Mechanical and Industrial Engineering, Indian Institute of Technology, Roorkee, India

[3] Leiden Institute of Advanced Computer Science, Leiden University, Leiden, The Netherlands

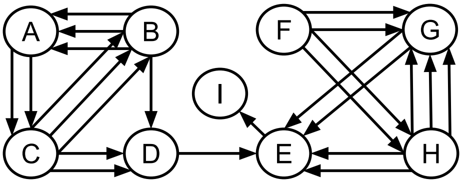


Fig. 1 An airline transportation network modelled as directed multigraph

airline, ...), and even tax evasion (Newman 2010). This has led to research on several types of networks, typically modelled as *simple graphs* (graphs having at most one edge between any pair of vertices) and *weighted graphs* (simple graphs but with weights on edges). A type of network that, to the best of our knowledge, has not yet been widely considered in the data mining literature[1] is one that needs to be modelled as a *multigraph* (graph in which multiple edges can be present between any pair of vertices). Motivated by an application in aviation, this paper lays the conceptual foundations for the discovery of subjectively interesting multigraphs patterns (SIMPs). SIMPs are defined as those subgraphs that are unexpected and/or contradict an analyst's prior beliefs or background knowledge (van Leeuwen et al. 2016). The rationale for the representation of an airline network as a multigraph and targeting of SIMPs vis-à-vis alternative approaches are discussed below.

In an airline network, symbolically depicted in Fig. 1, there can be several flights (edges in a graph) between a pair of airports (vertices in a graph), which explains as to why this network could be modelled as a multigraph.[2] Arguably, an airline network could also be studied as a *multilayer* graph, where multiple sets of edges are defined on the same set of vertices. In that setting, each set of edges acts as a unique layer, and different layers are characterised by different data properties. For instance, between a pair of airports, multiple flights from different airlines might operate, and each airline's flights may constitute a layer, differing from other layers. Notably, multigraphs may constitute building blocks for multilayer graphs [so far investigated only through simple graphs (Papalexakis et al. 2013; Qi et al. 2012)]. To avoid the added complexity of multilayer graphs, in this stage the multigraph representation of a network will form the basis for analysis in this paper.

Flight delays have punitive implications for airlines. Intuitively, and based on historical evidence, it is often believed that flight congestion between a pair of airports make them vulnerable to delays. Yet, delays are a reality, hence, it is critically important to mine the network data and facilitate scientifically informed assessment and decision making. Efforts in this direction have been made but they are limited in scope and practical relevance. For instance, finding *objectively* dense patterns (where density is defined through k-cores, cliques, k-plex, maximum average degree, etc.) is a commonly studied problem (Batagelj and Zaversnik 2003; Charikar 2000; Khuller and Saha 2009; McClosky and Hicks 2012; Palla et al. 2005; Tsourakakis et al. 2013). However, simple graphs do not suitably model an airline network in the first place. This paper attempts to overcome this limitation by focusing on multigraphs. Furthermore, it builds on the premise that capturing events (say, in terms of delays) which depart from an analyst's prior beliefs and may be referred as unexpectedly dense, *relative to what the analyst already knows* (van Leeuwen et al. 2016),

[1] Note that the term multigraph was used before Papalexakis et al. (2013) and Dong et al. (2012), but those works employ an alternative definition; see next section for details.

[2] At this formative stage, our endeavour is to analyse 'static' multigraphs (for fixed time intervals), though the longer-term goal would be to analyse dynamic multigraphs.

may be more revealing (say, in terms of source of delay), interesting, and practically useful. This justifies our focus on SIMPs dedicatedly in multigraph settings, besides the fact that this conceptual foundation could be useful in several other applications, including coauthorship analysis.

The structure of the paper is as follows. Following a description of related work in Sect. 2, our proposed approach is presented in Sect. 3. In particular, we formalise the conceptual contributions on SIMPs in Sect. 3.3, and present a greedy algorithm for the discovery of SIMPs in Sect. 3.5. Section 4 demonstrates the efficacy of the proposed algorithm; discusses the properties of the discovered SIMPs; compares our approach to existing methods; and presents a case study in aviation, highlighting how our approach could help improve an analyst's understanding of the problem. The paper concludes with key observations and future directions in Sect. 5.

2 Related work

Given that we are not aware of any previous work on mining multigraph patterns, this section briefly discusses related work on similar problems, dominantly in the context of simple graphs. In that, significant effort has been on finding dense patterns based on average degree (Charikar 2000; Khuller and Saha 2009), k-cores (Batagelj and Zaversnik 2003), cliques (Palla et al. 2005), quasi-cliques (Uno 2010; Tsourakakis et al. 2013), or k-plex (McClosky and Hicks 2012). For weighted graphs, the notion of average degree has been extended in Andersen and Chellapilla (2009). Structural partitioning of simple—unweighted and weighted—graphs, often based on modularity, has been actively utilised for community detection (Papadopoulos et al. 2012; Newman 2006; Girvan and Newman 2002; Pons and Latapy 2005; Clauset et al. 2004; Leicht and Newman 2008; Blondel et al. 2008).

Multilayer graphs are widely studied for finding patterns or clusters in the data; dense pattern discovery (Dong et al. 2012; Papalexakis et al. 2013); and community detection (Qi et al. 2012; Zhou et al. 2009; Xu et al. 2012; Silva et al. 2012; Ruan et al. 2013), by use of matrix factorisation, cluster expansion, pattern mining, etc.

Notably, the interestingness of a pattern is often defined as the departure from the expectations. In the case when expectations are objectively defined [say, through modularity (Clauset et al. 2004; Newman 2006) or edge surplus (Tsourakakis et al. 2013)], it is termed objectively interesting; and if expectations are derived subjectively (say, from the prior beliefs of an analyst), it is termed subjectively interesting. Arguably, the most fundamental contribution in the context of the latter has been made by De Bie (2011), where a generic framework based on maximum entropy principle was proposed to allow modelling of prior beliefs, laying the basis for subjective interestingness. Lijffijt et al. (2016) defined subjective interestingness for structured n-ary relational patterns. In this, it is assumed that the prior belief on the number of entities of a specific type to which a given entity is related is known. Drawing a parallel, it can be observed that this type of belief is apt for multilayer graphs such that each layer is a simple graph. However, it is different from our claim that a layer can also be a multigraph. Most importantly, van Leeuwen et al. (2016) defined subjectively interesting patterns for simple graphs and introduced a heuristic algorithm for mining those. Here, though the expectations were computed using the prior beliefs, the background distribution was assumed to be the product of independent Bernoulli distributions, given which the generalisation of this work to the multigraph setting is a non-trivial and challenging task.

In the context of objective interestingness, it has been noted that the modularity measure (Clauset et al. 2004; Newman 2006), originally proposed for unweighted simple graphs, can be trivially extended to weighted simple graphs. Although the resulting expected edge calculation is similar to one of our proposed type of beliefs, weighted (simple) graphs are inherently different from multigraphs: an edge weight in a weighted graph can be any real number, while an edge 'weight' in a multigraph is necessarily a natural number. In addition, the semantics are crucially different, which leads to different formalisations and possibilities. To demonstrate this we will empirically compare our proposed approach to the algorithm by Clauset et al. (2004).

3 Proposed approach

In this section, we formally introduce *multigraphs* and *subjective interestingness of a multigraph pattern*, based on the maximum entropy (MaxEnt) framework given by De Bie (2011). As in De Bie's framework, we compute the probability or background distribution, P of the data using the maximum entropy principle, treating the prior beliefs of the analyst as constraints. This also facilitates an iterative exploratory data mining process, implying that the background distribution can be updated upon presentation of subjectively interesting patterns. We will also discuss the method for updating the background distribution. Finally, we present an efficient greedy algorithm for mining subjectively interesting multigraph patterns.

3.1 Preliminaries

A *multigraph* is denoted by $G = (V, E)$, where V is a set of n vertices (usually indexed using symbol u or v) and E is a multiset of edges, where each edge $e \in E$ is an element of $V \times V$. In contrast to the common *simple graph* setting, there can be multiple edges between any pair of vertices. The adjacency matrix for the graph is denoted by $\mathbf{A} \in \mathbb{N}_0^{n \times n}$, with $a_{u,v} \in \mathbb{N}_0$ equal to the number of edges between u and v. For example, $a_{u,v} = 0$ means that there are no edges between u and v. This *undirected* definition can be straightforwardly extended to a *directed multigraph* by letting $a_{u,v}$ represent the number of edges from u to v. For the sake of simplicity, in this paper we focus the exposition on multigraphs without self-edges, for which it holds that $(u, v) \in E \Rightarrow u \neq v$, but if desired this restriction could simply be dropped.

We build on the premise that an analyst knows (or has direct access to) the list of vertices V in the graph, and is interested in improving self's knowledge and understanding of the edges. Thus, the data to be mined is the edge multiset E, and the domain of this data is $\mathbb{N}_0^{n \times n}$ (further constrained by exclusion of self-loops, implying that the diagonal values of $\mathbf{A}$ have to be 0).

The framework by De Bie (2011) suggests that prior knowledge (modelled as constraints) can be represented as a probability distribution P over the data domain. As the constraints typically leave many of such distributions possible, the maximum entropy principle is leveraged to argue that the distribution having the largest entropy should be used. The framework then quantifies the subjective interestingness of a pattern as the ratio of *information content* to *description length*, where information content is the negative logarithm of the probability of the pattern given the background distribution, and description length is the code length required to communicate the pattern to the user. In the following, we will build on this framework for multigraph patterns, albeit with a different definition of subjective interestingness.

3.2 Prior beliefs

We here consider and model the following three different types of prior beliefs that an analyst may have:

1. *Total number of edges (Belief-c)* The analyst here is assumed to have a prior belief concerning (only) the total number of edges in the network, e.g., on the total number of flights in case of airline data. This follows:

$$\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{u,v\in V} a_{u,v} = |E|. \tag{1}$$

   The MaxEnt distribution with constraint Eq. 1 results in a product of independent uniform geometric distributions, one for each random variable $a_{u,v} \in \mathbb{N}_0$ [cf. De Bie (2011)], where $P(\mathbf{A}) = \prod_{u,v\in V} \exp(2\lambda \cdot a_{u,v}) \cdot (1 - \exp(2\lambda))$. Here, each distribution represented as $P_{u,v}(a_{u,v})$ has a probability of success equal to $[1 - \exp(2\lambda)]$, where λ is a Lagrangian multiplier corresponding to the constraint in Eq. 1.
2. *Number of edges per vertex (Belief-i)* In this case, the analyst is assumed to have prior beliefs on the row and/or column marginals of the adjacency matrix, denoted by d_u^r and d_v^c respectively. In the airline case, this corresponds to knowing the total number of flights leaving from (d_u^r) or arriving (d_v^c) at each airport. This belief is represented by

$$\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{v\in V} a_{u,v} = d_u^r,\ (\forall u);\ \sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{u\in V} a_{u,v} = d_v^c.\ (\forall v) \tag{2}$$

   We observe that the MaxEnt distribution with constraints in Eq. 2 results in a product of independent geometric distributions given by $P(\mathbf{A}) = \prod_{u,v\in V} \exp((\lambda_u^r + \lambda_v^c) \cdot a_{u,v}) \cdot (1 - \exp((\lambda_u^r + \lambda_v^c))$, for each random variable $a_{u,v} \in \mathbb{N}_0$. This corresponds to the ‘geometric’ case in De Bie (2011), where each distribution $P_{u,v}(a_{u,v})$ has a probability of success equal to $[1 - \exp(\lambda_u^r + \lambda_v^c)]$. Here, λ_u^r and λ_v^c are Lagrangian multipliers following the constraints in Eq. 2.
3. *Number of neighbours per vertex (Belief-m)* In the third and final case, the analyst is assumed to have a prior belief about the number of unique neighbours of each vertex, referred to as m_u. In an airline case, this could be considered as the total number of unique routes on which an airline operates from any airport. This prior belief is represented as

$$\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{v\in V} 1_{a_{u,v}} = m_u^r, (\forall u);\ \sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{u\in V} 1_{a_{u,v}} = m_v^c,\ (\forall v) \tag{3}$$

   where $1_{a_{u,v}}$ is the indicator function, which equals 1 if $a_{u,v}$ is a non-zero value and 0 otherwise. This case is a multigraph-specific belief, as in case of a simple graph d_u would be equal to m_u, intuitive of the fact that at most one edge can exist between any two vertices. Hence, we will use this belief to complement the previous two types of belief. In this paper, we consider the case where this type of belief is combined with Belief-i. The MaxEnt distribution $P(\mathbf{A})$ for the data with constraints in Eqs. 2 and 3 reduces to a product of independent probability distributions $P(\mathbf{A}) = \prod_{u,v \in V} P_{u,v}(a_{u,v})$ for each random variable $a_{u,v} \in \mathbb{N}_0$, where $P_{u,v}(a_{u,v}) = \frac{[1-\exp(\lambda_u^r+\lambda_v^c)]}{[1-\exp(\lambda_u^r+\lambda_v^c)(1-\exp(\mu_u^r+\mu_v^c))]} \cdot \exp\left(\lambda_u^r + \lambda_v^c\right)^{a_{u,v}} \cdot \exp\left(\mu_u^r + \mu_v^c\right)^{1_{a_{u,v} \neq 0}}$. Here λ_u^r, λ_v^c, μ_u^r and μ_v^c are Lagrangian multipliers corresponding to the constraints in Eqs. 2 and 3 respectively. For completeness, a proof of the MaxEnt distribution $P(\mathbf{A})$ for this case is given in "Appendix 1".

The above-mentioned constraints are described for directed multigraphs represented by $\mathbf{A}$, however for undirected multigraphs $u < v$ should be added as an additional constraint. In this paper, the above three types of prior beliefs or knowledge will be evaluated. However, other types of prior beliefs could also be considered, for example, details about different airline carrier's flights arriving or departing from an airport. Though it is beyond the scope of this paper, such cases would also lead to a product of independent probability distributions, which can be used to compute the expected number of edges between any vertex pair.

3.3 Subjective interestingness for multigraph patterns

Given the prior beliefs of the analyst, the background distribution of the data can be derived as the MaxEnt distribution (De Bie 2011). We now establish a subjective interestingness measure for multigraph patterns given the background distribution and the data.

As multigraphs do not have a strict limit on the maximum number of edges that can occur between any pair of vertices, existing work on simple graphs by van Leeuwen et al. (2016) cannot be directly extended to multigraphs. We, therefore, introduce a new definition of interestingness based on the *expectation matrix* $\mathcal{E}$. In this matrix, of size $|V| \times |V|$, each entry $\mathcal{E}_{u,v}$ is defined as the number of expected edges—based on the prior beliefs—between vertices u and v.

The expectation of any geometric distribution of the form $(1-p)^x \cdot p$ for random variable $x \in \mathbb{N}_0$, where p is the probability of success, is given as $E(x) = \frac{1-p}{p}$. The probability distributions for Belief-c and Belief-i are represented in the natural form of a geometric distribution. Thus, we have expectation $\mathcal{E}_{u,v} = \frac{\exp(2\lambda)}{1-\exp(2\lambda)} = \rho$ and $\mathcal{E}_{u,v} = \frac{\exp(\lambda_u^r+\lambda_v^c)}{1-\exp(\lambda_u^r+\lambda_v^c)}$ for Belief-c and Belief-i, respectively. Here, ρ is the density[3] of a graph.

The probability distribution for Belief-m, however, cannot be represented in the natural form of a geometric distribution. Hence, the expected number of edges between vertices u and v is computed as

$$\mathcal{E}_{u,v} = \frac{\exp(\lambda_u^r + \lambda_v^c) \cdot \exp(\mu_u^r + \mu_v^c)}{\left[1 - \exp(\lambda_u^r + \lambda_v^c)\right]\left[1 - \exp(\lambda_u^r + \lambda_v^c)\left(1 - \exp(\mu_u^r + \mu_v^c)\right)\right]}. \tag{4}$$

Next, we quantify the interestingness of a vertex-induced subgraph pattern by *the difference between the actual and the expected number of edges*. For this, we derive what we call the *gulf matrix* $\mathcal{G}$, which is computed as the difference between the adjacency matrix and

[3] For undirected graphs $\rho = \frac{2*|E|}{|V|\cdot(|V|-1)}$, for directed graphs $\rho = \frac{|E|}{|V|\cdot(|V|-1)}$.

expectation matrix, i.e., $\mathcal{G} = \mathbf{A} - \mathcal{E}$. A value $\mathcal{G}_{u,v}$ is positive if the expected number of edges between u and v is lower than the actual number of edges, and negative in the opposite case. Without loss of generality, we assume that only positive differences are of interest; one could reverse the signs to discover 'sparse subgraphs'.

For a given pattern, we sum the deviations over all vertex pairs it contains, and define this sum as the aggregate deviation of the pattern, as follows.

Definition 1 (*Aggregate Deviation*) Given multigraph $G = (V, E)$ and gulf matrix $\mathcal{G}$, the *aggregate deviation AD* of a subgraph $H = (W, E')$, where $W \subseteq V$ and $E' \subseteq E$, is given by $AD(H, \mathcal{G}) = \sum_{u,v \in W} \mathcal{G}_{u,v}$.

One might be inclined to mine subgraphs that maximise AD, but in practice, this is likely to lead to large subgraphs. This is problematic because large subgraphs may not be interesting for and/or comprehensible to the analyst. Similar to existing subjective interestingness approaches (De Bie 2011; Lijffijt et al. 2016; van Leeuwen et al. 2016), we, therefore, penalise a pattern's deviation with its description length, i.e., its 'complexity'.

Definition 2 (*Description Length*) Given multigraph $G = (V, E)$, subgraph $H = (W, E')$, and parameter q, the cost required to describe a subgraph to the analyst—in terms of its vertices—is given by *description length* DL, defined as

$$\begin{aligned} DL(H) &= -\sum_{u \in W} \log(q) - \sum_{u \notin W, u \in V} \log(1-q) \\ &= |W| \cdot \log\left(\frac{1-q}{q}\right) + |V| \cdot \log\left(\frac{1}{1-q}\right), \end{aligned}$$

where $-\log(q)$ is the cost of a vertex included in W and $-\log(1-q)$ is the cost of a vertex excluded from W.

Definition 2 uses Shannon-optimal codes to describe the pattern, using a vertex probability, i.e., parameter q, that is set by the analyst in advance. The smaller the analyst believes the size of an interesting pattern to be, the smaller the q and the smaller the exclusion cost of a vertex, and the other way around. Once q is fixed then the description length increases with the size of the pattern as for each added vertex in a pattern a cost equal to $\log((1-q)/q)$ is added to the description length. Thus, q can be interpreted as the expected probability that a vertex is included in a random pattern and is set by the analyst based on expected/desired pattern size. Description length can be used to penalise larger patterns, for which it is easier to have a large AD.

Ideally, a pattern is considered to be interesting if it is highly informative (quantified in terms of aggregate deviation, AD) and can be encoded with a short code (measured in terms of description length, DL). Thus, we next define subjective interestingness of a pattern as the ratio of its aggregate deviation to its description length.

Definition 3 (*Subjective Interestingness*) Given multigraph $G = (V, E)$, subgraph H, and gulf matrix $\mathcal{G}$, the *subjective interestingness SI* of H is given by $SI(H, \mathcal{G}) = \frac{AD(H,\mathcal{G})}{DL(H)}$.

Note that in the previous we considered *any* vertex-induced subgraph, but this includes subgraphs that consist of multiple components, i.e., subgraphs that are not connected. As an analyst will expect patterns to be connected, we add the constraint that each subgraph has to be connected.[4] This leads to the following problem for finding the subjectively most interesting multigraph pattern.

Problem 1 (*SIMP: Subjectively Interesting Multigraph Pattern*) Let $G = (V, E)$ be a multigraph and $\mathcal{G}$ a gulf matrix. Find a set of vertices $W \subset V$ and its corresponding vertex-induced subgraph H that maximises $SI(H, \mathcal{G})$ such that H is a (weakly) connected component.

3.4 Updating the background distribution

When a new pattern is found, it is presented to the analyst, which then transforms the knowledge of the analyst, who learns from the information contained in the pattern. Hence, these newly learned information should be reflected in the background distribution. More specifically, in the updated background distribution $P'(\mathbf{A})$ the expectation of the number of edges in the pattern should be equal to the actual number of edges found. The rationale behind this is that, by updating the background distribution in this manner, the aggregate deviation of the pattern becomes (almost) zero and hence the pattern is no longer interesting.

Let $H = (W, E')$ be the communicated pattern, then the updated MaxEnt distribution is calculated using the following convex optimisation problem, which is the *I-projection* of the preceding background distribution onto the set of distributions that are consistent with the communicated pattern (De Bie 2011). Thus, the problem is formulated as

$$P'(\mathbf{A}) = \underset{Q}{\operatorname{argmin}} \sum_{\mathbf{A}} Q(\mathbf{A}) \log \left(\frac{Q(\mathbf{A})}{P(\mathbf{A})} \right), \tag{5}$$

$$s.t. \sum_{\mathbf{A} \in \mathbb{N}_0^{n \times n}} Q(\mathbf{A}) \sum_{u,v \in W} a_{u,v} = |E'|; \sum_{\mathbf{A} \in \mathbb{N}_0^{n \times n}} Q(\mathbf{A}) = 1, \tag{6}$$

where the constraint in Eq. 6 represents the acquired belief of the analyst on the data. That is, the vertex-induced subgraph H, with the set of vertices W, contains $|E'|$ edges. Using this updating procedure, we can perform an iterative exploratory data mining process: we can mine the subjectively most interesting multigraph pattern from the data, update the background distribution, and repeatedly perform these two steps to mine multiple SIMPs.

Theorem 1 *Let* $P(\mathbf{A})$ *be a product of independent probability distributions over data* $\mathbf{A} \in \mathbb{N}_0^{V \times V}$, *then the optimal solution to the problem defined by Eqs. 5–6 is also a product of an independent probability distributions* $P'(\mathbf{A})$, *such that:*

1. *if* $P(\mathbf{A}) = \prod_{u,v \in V} (1 - p_{u,v})^{a_{u,v}} \cdot p_{u,v}$, *then* $P'(\mathbf{A}) = \prod_{u,v \in V} (1 - p'_{u,v})^{a_{u,v}} \cdot p'_{u,v}$,
2. *if* $P(\mathbf{A}) = \prod_{u,v \in V} \frac{1 - R_{u,v}}{1 - R_{u,v}(1 - S_{u,v})} \cdot R_{u,v}^{a_{u,v}} \cdot S_{u,v}^{1_{a_{u,v}}}$, *then* d$P'(\mathbf{A}) = \prod_{u,v \in V} \frac{1 - R'_{u,v}}{1 - R'_{u,v}(1 - S_{u,v})} \cdot (R'_{u,v})^{a_{u,v}} \cdot S_{u,v}^{1_{a_{u,v}}}$,

[4] For directed multigraphs the constraint is relaxed to weakly connected component, i.e., the undirected equivalent of the directed graph is a connected graph.

where

$$p'_{u,v} = \begin{cases} 1-(1-p_{u,v})\exp(\lambda_H), & \text{if} \quad (u,v) \in W, \\ p_{u,v}, & \text{otherwise,} \end{cases}$$

$$R'_{u,v} = \begin{cases} R_{u,v} \cdot \exp(\lambda_H), & \text{if} \quad (u,v) \in W, \\ R_{u,v}, & \text{otherwise.} \end{cases}$$

Here λ_H *is a Lagrangian multiplier and a unique real number such that* $(1-p_{u,v}) \cdot \exp(\lambda_H) \in (0\,1) \subset \mathbb{R}$ *and* $R_{u,v} \in (0\,1) \subset \mathbb{R}$.

It is observed that background distribution $P(\mathbf{A})$ can be updated using Theorem 1. For Belief-c and Belief-i claim 1 is followed, while for Belief-m we follow claim 2, where $R_{u,v} = \exp(\lambda^r_u + \lambda^c_v)$ and $S_{u,v} = \exp(\mu^r_u + \mu^c_v)$. Both claims in Theorem 1 follow the same principle, hence for brevity, only the proof of claim 2 is given in “Appendix 2”.

For the computation of aggregate deviation AD, we require to compute the expected number of edges between two vertices given the background distribution. It is inefficient to update and store all the expectations every time the background distribution is updated. It is therefore recommended to only store the λ_H and compute the expectation whenever required. After a series of patterns $H = (W, E')$ are presented to the user $p'_{u,v}$ is given by $1-(1-p_{u,v}) \cdot \exp\left(\sum_{H:u,v \in W} \lambda_H\right)$, and $R'_{u,v}$ is given by $R_{u,v} \cdot \exp\left(\sum_{H:u,v \in W} \lambda_H\right)$.

```
Algorithm 1: HillClimber($G$, $H$, $\mathcal{G}$, $I$)
Input  : Graph dataset $G = (V, E)$, seed subgraph $H = (W, E')$, gulf matrix $\mathcal{G}$,
         and interestingness of seed subgraph $I$
Output: Multigraph pattern $H$, a heuristic solution to Problem 1, together with
         its interestingness $I$
begin
    $H_a, I_a \leftarrow$ CheckGraphExtension($G$, $H$, $\mathcal{G}$, $I$)
    if $I_a > I$ then  $H \leftarrow H_a$, $I \leftarrow I_a$      return HillClimber($G$, $H$, $\mathcal{G}$, $I$) ;
    else
        $H_r, I_r \leftarrow$ CheckGraphReduction($G$, $H$, $\mathcal{G}$, $I$)
        if $I_r > I$ then  $H \leftarrow H_r$, $I \leftarrow I_r$      return HillClimber($G$, $H$, $\mathcal{G}$, $I$) ;
        else  return $H$, $I$ ;
```

3.5 Algorithm

To exhaustively solve Problem 1, we would have to consider all $2^{|V|}$ possible subsets of V, for each subset determine its vertex-induced subgraph, check if it is connected, and compute its interestingness. As there are hardly any possibilities for pruning this would lead to very large run-times and we resort to a greedy hill-climber, which was shown to give good solutions in little time in the simple graph setting (van Leeuwen et al. 2016).

As input Algorithm 1 takes a multigraph G, seed subgraph $H = (W, E')$, gulf matrix $\mathcal{G}$, and—for efficiency—corresponding interestingness I (i.e., $I = SI(H, \mathcal{G})$). For directed multigraphs, each vertex is (virtually) split into two, one having in-degree equal to zero and the other having out-degree equal to zero, based on which corresponding concepts *Predecessors* & *OutNode* and *Successors* & *InNode*, respectively, are defined. Hence, a directed (sub-)graph has two lists of vertices one of *OutNodes*, W_{out} and the other of *InNodes*, W_{in}, thus, $W = W_{in} \cup W_{out}$.

3.5.1 Description

Algorithm 1 initially tries to add neighbouring vertices to the current subgraph (Lines 2–3). If the addition of any neighbour vertex results in improved interestingness, the addition is consolidated and the method recurses (L3). Otherwise, the algorithm eliminates, one by one, vertices from the current subgraph and checks whether this improves interestingness (L5–6). When no improvement can be made in any iteration, the procedure stops (L7).

Algorithm 2: CheckGraphExtension(G, H, $\mathcal{G}$, I)

```
1  begin
2      $H^* \leftarrow H$, $I^* \leftarrow I$
3      if $type(G) = Undirected$ then
4          for $u \in Neighbors(H, G) \setminus W$ do
5              $W' \leftarrow W \cup \{u\}$, $H' \leftarrow (W', E'_H)$, $I' \leftarrow I(H', \mathcal{G})$
6              if $I' > I^*$ then $H^* \leftarrow H'$, $I^* \leftarrow I'$;
7      else
8          for $u \in Predecessors(H, G) \setminus W_{out}$ do
9              $W'_{out} \leftarrow W_{out} \cup \{u\}$, $W' \leftarrow [W_{in}, W'_{out}]$, $H' \leftarrow (W', E'_H)$,
                 $I' \leftarrow I(H', \mathcal{G})$
10             if $I' > I^*$ then $H^* \leftarrow H'$, $I^* \leftarrow I'$;
11         for $v \in Successors(H, G) \setminus W_{in}$ do
12             $W'_{in} \leftarrow W_{in} \cup \{u\}$, $W' \leftarrow [W'_{in}, W_{out}]$, $H' \leftarrow (W', E'_H)$, $I' \leftarrow I(H', \mathcal{G})$
13             if $I' > I^*$ then $H^* \leftarrow H'$, $I^* \leftarrow I'$;
14     return $H^*, I^*$
```

Algorithm 2 and 3 are two subroutines that return the best addition or removal step possible respectively. Function *type*(G) determines the type of graph; if the graph is undirected then vertices are added (Algorithm 2, L3–6) or removed (Algorithm 3, L3–6) one by one without distinguishing the type of neighbour as *predecessor* or *successor*, unlike in the case of directed graphs (Algorithm 2, 3; L7–13).

Algorithm 3: CheckGraphReduction(G, H, $\mathcal{G}$, I)

```
1  begin
2      $H^* \leftarrow H$, $I^* \leftarrow I$
3      if $type(G) = Undirected$ then
4          for $u \in W$ do
5              $W' \leftarrow W \setminus \{u\}$, $H' \leftarrow (W', E'_H)$, $I' \leftarrow I(H', \mathcal{G})$
6              if $I' > I^*$ then $H^* \leftarrow H'$, $I^* \leftarrow I'$;
7      else
8          for $u \in W_{out}$ do
9              $W'_{out} \leftarrow W_{out} \setminus \{u\}$, $W' \leftarrow [W_{in}, W'_{out}]$, $H' \leftarrow (W', E'_H)$,
                 $I' \leftarrow I(H', \mathcal{G})$
10             if $I' > I^*$ then $H^* \leftarrow H'$, $I^* \leftarrow I'$;
11         for $v \in W_{in}$ do
12             $W'_{in} \leftarrow W_{in} \setminus \{u\}$, $W' \leftarrow [W'_{in}, W_{out}]$, $H' \leftarrow (W', E'_H)$, $I' \leftarrow I(H', \mathcal{G})$
13             if $I' > I^*$ then $H^* \leftarrow H'$, $I^* \leftarrow I'$;
14     return $H^*, I^*$
```

The proposed hill-climber, which is a greedy heuristic, may experience problems due to locally converging to a sub-optimal solution. This largely depends on the choice of seed (initial subgraph) provided to the algorithm. To overcome this pitfall, we propose to independently run the hill-climber for k different seeds and choose the best solution among the k returned patterns. The seeds can be chosen on the basis of different criteria; we consider the following three:

1. *Degree* Select the top-k vertices having the highest degrees in the graph, where each individual vertex is used as a seed once.
2. *Uniform* Select k different vertices at random, where each individual vertex is used as a seed once.
3. *Interest* Use the k most interesting vertices and use each of those individually as seed. The interestingness of a vertex is calculated as the subjective interestingness SI of the vertex-induced subgraph of the vertex together with its immediate neighbours.

It is intuitively beneficial but cost-inefficient to evaluate all possible seeds (i.e., to use each vertex in a graph as an independent seed). We demonstrate the effectiveness of the above-described seed selection strategies in Sect. 4.

3.5.2 Complexity

In a single iteration of the hill-climber, interestingness computation is the most costly part of the computation and has complexity $O(|W|^2)$, as aggregate deviation computation requires to sum elements in the gulf matrix. We can, however, maintain a list of potential vertices that can be added to the current subgraph, along with the potential gain in aggregate deviation associated with each candidate vertex. These potential gains are updated upon addition or removal of a vertex from the current subgraph, which has complexity $O(|V|)$. As the complexity of the search procedure is identical, the resulting overall complexity is $O(|V|)$.

4 Experiments

In this section, we evaluate our proposed approach and compare it to related methods. To distinguish the results obtained using different types of prior beliefs, we denote our proposed approach using the background distribution given by Belief-c as SIMP-c; by Belief-i as SIMP-i; and by Belief-m as SIMP-m. For the initial experiments we use both synthetic and real multigraphs; later we present a case study on an airline dataset.

4.1 Datasets

We generate synthetic datasets in two steps. First, a simple, undirected graph is generated using the preferential attachment method by Barabási and Albert (1999). Second, a randomly generated sequence is used to add parallel edges to make it a multigraph. This sequence has a length equal to the number of edges in the simple graph, and is a combination of a Bernoulli (parameterised by the probability of success p_b) and geometric distribution (parameterised by p_g). The former determines whether parallel edges are added, while

Table 1 Properties of the multigraph datasets: number of vertices (|*V*|), number of edges ($|E^m|$), number of edges in a simple graph projection ($|E^s|$), probabilities of success for generating multigraph sequences (p_b and p_g), and Barabási-Albert model parameter (l)

DS	p_b	p_g	l	|*V*|	$\|E^m\|$	$\|E^s\|$
SYN1	0.2	0.40	10	200	2628	1900
SYN2	0.2	0.65	10	1000	12,977	9900
SYN3	0.4	0.80	10	10,000	149,729	99,900
SYN4	0.2	0.65	10	50,000	653,821	499,900
DBLP1	–	–	–	5271	19,888	16,847
DBLP2	–	–	–	6956	23,879	20,837
DBLP3	–	–	–	18,466	98,493	78,699
DBLP4	–	–	–	65,074	230,006	202,642
IMDB	–	–	–	4644	13,416	12,702

the latter determines how many parallel edges are added to the vertex pair indicated by the index in the sequence (if any). For the Barabási-Albert model, parameter l is used to define the maximum number of vertices to which a newly inserted vertex should be connected. Parameter values and properties of the resulting four synthetic datasets are shown in Table 1, where superscripts s and m refer to the initial *s*imple graph and the final *m*ultigraph, respectively.

From the DBLP[5] data, we generate a co-author graph, where authors are represented as vertices and co-authored publications as undirected edges. Due to its large size, we have created multiple datasets from the data using different queries: (1) all conference publications of October 2017 (DBLP1) and July 2017 (DBLP2); (2) all publications of the top-20[6] conferences of Data Mining, Machine Learning and Artificial Intelligence in 2016–2017 (DBLP3); and (3) all journal publications of May 2017 (DBLP4). To obtain the IMDB[7] dataset we build a co-actor graph, where actors are represented as vertices and common movies as undirected edges. For each dataset, we only consider the largest connected component.

4.2 Evaluation criteria

We characterise the results using several commonly used subgraph properties: the number of vertices |*V*|; the number of edges |*E*|; density ρ, given by $(2 \times |E|)/(|V| \times (|V| - 1))$; average degree η, given by $2 \times |E|/|V|$; and diameter d. Further, to demonstrate the benefits of considering multigraphs over simple graphs, we 'project' the multigraph patterns, indicated by superscript m, to their simple graph counterparts, indicated by superscript s, by removing any 'parallel' edges between each vertex pair. We then define a new measure, denoted γ, to quantify the number of parallel edges in a subgraph relative to the number of vertex pairs with at least one edge: $(|E^m| - |E^s|)/|V|$.

[5] Source: https://dblp.uni-trier.de/.

[6] Source: https://scholar.google.co.in/citations?view_op=top_venues&hl=en&vq=eng.

[7] Source: https://www.kaggle.com/carolzhangdc/imdb-5000-movie-dataset.

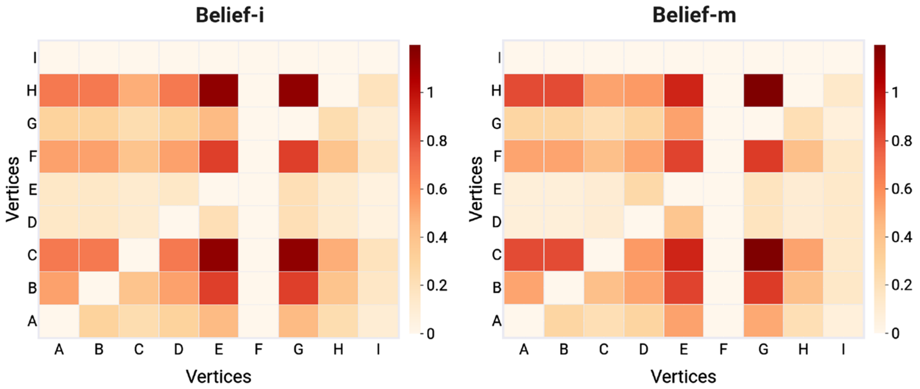


Fig. 2 Heatmap showing the expected number of edges between all pairs of vertices ($\mathcal{E}$) for the toy example (Fig. 1) w.r.t. Belief-i and Belief-m (dark colour represents higher expectations)

4.3 Prior beliefs and interestingness evaluation

The different types of prior belief that we defined reflect different types of knowledge an analyst may have. Here we demonstrate the different effects of the proposed types of prior beliefs. The expectation on the number of edges between two vertices (or the probability distribution) varies with the prior knowledge as quantified using the maximum entropy principle (shown earlier).

Belief-c results in a uniform distribution with equal expectation for all pairs of vertices. Thus, a subgraph with high average vertex degree would be considered most interesting under this type of belief, which is confirmed by the co-occurrence of high values of both interestingness (SI-c) and average degree (η) in Table 3.

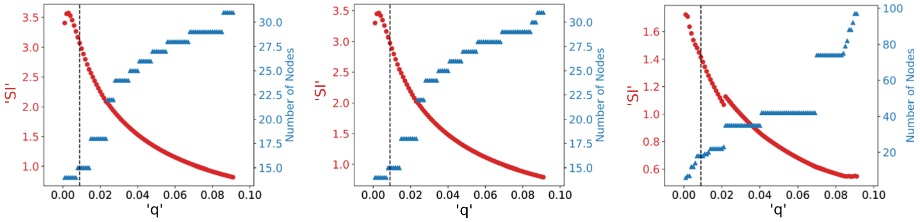
Belief-i and Belief-m represent more extensive forms of prior knowledge than Belief-c. Using the toy data set from Fig. 1, the expectation between all pairs of vertices is shown in Fig. 2 for both Belief-i and Belief-m. With Belief-i, it can be seen that the highest expectation on the number of edges is for vertices C and E, as C has the highest number of outgoing edges and E has the maximum number of incoming edges in the graph. As subjective interestingness is defined as the positive deviations from the expectation, this type of belief usually leads to dense patterns (as can be witnessed from Table 3). Belief-m is more profound than Belief-i, as here the analyst has additional information on the number of unique neighbours for each vertex. With the addition of a new constraint, the expectation between vertices C & E decreases, as C has only two successors, which is compensated for by an increased expectation for the number of edges between vertex pairs C & A and C & B. In this particular case, these expectations are much closer to the actual values.

4.4 Description length and seeding strategy evaluation

In this subsection, we empirically demonstrate the effect of the value of parameter q as used in the description length. For most of the datasets, including the larger graphs, a value of 0.01 was found to be robust as it results in moderately sized patterns. Note that this corresponds to a belief that a pattern is expected to consist of 1% of all vertices in a graph. For the DBLP1 dataset, the effect of varying q is shown in Fig. 3. The plots demonstrate how

Table 2 Mean subjective interestingness (*SI*) of the best pattern found using SIMP-c, SIMP-i, and SIMP-m, for ‘Interest’, ‘Degree’ and ‘Uniform’ seed selection strategies, with corresponding runtimes (in seconds)

Belief	*k*	1		10		50		All	
	Seed Type	SI	Time	SI	Time	SI	Time	SI	Time
SIMP-c	Interest	1.799	3.84	1.911	23.03	1.915	108.47	1.919	2758
	Degree	1.304	2.02	1.911	19.35	1.916	118.27		
	Uniform	0.844	2.62	1.453	26.75	1.456	124.69		
SIMP-i	Interest	1.592	2.26	1.602	3.50	1.602	9.41	1.607	412
	Degree	0.781	0.13	1.511	1.03	1.602	6.32		
	Uniform	0.439	0.33	0.720	1.66	1.156	7.54		
SIMP-m	Interest	1.015	2.50	1.170	5.27	1.170	13.47	1.173	591
	Degree	0.628	0.22	1.150	1.68	1.163	9.98		
	Uniform	0.449	0.24	0.717	4.80	1.094	13.67		



(a) Belief: SIMP-c **(b)** Belief: SIMP-i **(c)** Belief: SIMP-m

Fig. 3 Parameter q versus the number of vertices (triangles) versus subjective interestingness (*SI*, circles), for subgraphs found on DBLP1. The vertical dashed line indicates $q = 0.01$

q can be used to influence pattern size as desired by the analyst. For the remainder of the paper, we fix q to 0.01.

Next, we perform experiments on datasets SYN1, SYN2, DBLP1 and DBLP2, for different number of independent runs (represented by k) and for each type of seeding strategy. The results, aggregated over the four mentioned datasets, are shown in Table 2 (mean interestingness score and sum of the runtimes). We can observe that, in general, the highest mean subjective interestingness (*SI*) was found using the *interest*-based seed selection strategy, followed by the *degree* based strategy, for all three types of belief. Further, we observe that the extra runtime needed for using all individual vertices as seeds is substantially larger than the improvement in subjective interestingness. The results show that $k = 10$ provides an adequate trade-off, saving substantially on runtime while hardly giving in on subjective interestingness. Hence, for all remaining experiments, we will use the *interest*-based seeding strategy with $k = 10$ independent runs.

4.5 Quantitative evaluation

In this subsection, we demonstrate that (1) our proposed subjective interestingness measure is different from existing measures designed for simple and multigraphs, and (2) the

Table 3 Properties (see text) of the best pattern found by each method

DS	Method	$\|V\|$	$\|E^m\|$	$\|E^s\|$	SI-c	SI-i	SI-m	ρ	η	d	γ
SYN1	SIMP-c	31.66	400.50	205.00	**1.51**	0.78	0.94	1.441	**24.77**	2.90	6.34
	SIMP-i	5.42	40.46	7.22	1.03	**0.94**	1.21	5.468	15.31	2.38	**6.50**
	SIMP-m	4.72	35.18	5.98	0.98	0.76	**1.26**	**5.964**	15.00	2.08	6.39
	SSG-c	10.98	64.23	48.90	0.77	0.21	0.20	1.170	11.69	1.88	1.39
	SSG-i	3.70	6.24	4.58	0.68	0.32	0.28	1.290	3.30	**1.36**	0.46
	CNM	30.54	159.83	98.95	0.32	0.43	0.39	0.360	10.34	3.40	2.00
SYN2	SIMP-c	90.74	1123.52	775.00	**1.65**	0.47	0.48	0.279	**24.75**	3.02	**3.85**
	SIMP-i	18.84	101.44	39.60	1.31	**0.61**	0.59	0.752	10.79	3.88	3.41
	SIMP-m	23.02	147.16	66.96	1.39	0.54	**0.65**	0.691	12.51	3.56	3.56
	SSG-c	24.92	216.14	167.86	1.40	0.18	0.17	0.730	17.25	**2.00**	1.93
	SSG-i	6.22	11.60	9.00	1.07	0.21	0.22	**0.770**	3.61	2.30	0.40
	CNM	116.19	610.35	371.32	0.37	0.52	0.50	0.095	10.45	4.91	2.06
SYN3	SIMP-c	381.60	5806.46	3626.30	**2.09**	0.51	0.54	0.080	**30.44**	3.92	**5.72**
	SIMP-i	175.50	1135.00	546.70	1.45	**0.67**	0.61	0.075	12.92	5.06	3.35
	SIMP-m	161.46	1414.60	735.78	1.41	0.64	**0.77**	0.111	17.55	4.36	4.21
	SSG-c	79.78	956.40	703.62	1.65	0.61	0.43	**0.304**	23.98	**3.00**	3.16
	SSG-i	30.84	58.72	44.48	1.02	0.36	0.31	0.130	3.79	6.36	0.46
	CNM	903.89	4480.05	2589.45	0.38	0.52	0.48	0.010	9.89	6.85	2.09
SYN4	SIMP-c	1052.20	14422.60	9951.30	**1.80**	0.74	0.68	0.030	**27.42**	4.00	4.25
	SIMP-i	324.30	2864.42	541.68	1.43	**0.91**	0.88	0.055	17.59	9.11	6.70
	SIMP-m	418.45	3918.32	898.45	1.66	0.87	**0.99**	0.045	18.73	8.12	**6.81**
	SSG-c	280.36	3535.70	2705.70	1.35	0.14	0.17	**0.090**	25.21	**3.02**	2.96
	SSG-i	164.08	267.60	207.06	1.08	0.15	0.18	0.020	3.28	12.00	0.37
	CNM	4303.55	21044.65	12138.40	0.41	0.51	0.42	0.002	9.73	9.45	2.07
DBLP1	SIMP-c	15	524	105	**2.98**	2.89	1.31	**4.990**	**69.87**	**1.00**	**27.93**
	SIMP-i	15	524	105	2.98	**2.89**	1.31	**4.990**	**69.87**	**1.00**	**27.93**
	SIMP-m	18	406	125	2.91	2.87	**1.38**	2.654	45.11	2.00	15.61
	SSG-c	20	192	190	0.92	0.89	0.76	1.011	19.20	**1.00**	0.10
	SSG-i	20	190	190	0.91	0.90	0.78	1.000	19.00	**1.00**	0.00
	CNM	532	2010	1696	0.45	0.49	0.42	0.014	7.56	7.00	0.59
DBLP2	SIMP-c	30	448	435	**1.49**	1.46	1.42	1.030	29.87	1.00	0.43
	SIMP-i	30	448	435	1.49	**1.46**	1.42	1.030	29.87	1.00	0.43
	SIMP-m	30	448	435	1.49	1.46	**1.42**	1.030	29.87	1.00	0.43
	SSG-c	30	448	435	**1.49**	1.46	1.42	1.030	29.87	1.00	0.43
	SSG-i	30	448	435	1.49	**1.46**	1.42	1.030	29.87	1.00	0.43
	CNM	307	998	856	0.42	0.45	0.41	0.021	6.50	12.00	**0.58**
DBLP3	SIMP-c	140	14626	9692	**12.23**	9.48	10.31	1.503	**208.94**	2.00	**35.24**
	SIMP-i	142	14780	9843	12.22	**9.49**	10.30	1.476	208.17	2.00	34.77
	SIMP-m	140	14626	9692	12.23	9.48	**10.31**	1.503	**208.94**	2.00	**35.24**
	SSG-c	104	8215	5356	8.58	9.40	9.39	**1.534**	157.98	**1.00**	27.40
	SSG-i	139	14488	9591	12.19	9.45	10.29	1.511	208.46	1.00	35.23
	CNM	369	18320	13283	6.72	5.22	5.04	0.270	99.30	5.00	13.68

Table 3 (continued)

DS	Method	$\|V\|$	$\|E^m\|$	$\|E^s\|$	SI-c	SI-i	SI-m	ρ	η	d	γ
DBLP4	SIMP-c	55	1495	1485	**1.14**	0.94	0.91	**1.007**	**54.36**	**1.00**	0.18
	SIMP-i	71	1663	1653	1.05	**1.17**	1.13	0.669	46.85	2.00	0.14
	SIMP-m	71	1663	1653	1.05	1.17	**1.13**	0.669	46.85	2.00	0.14
	SSG-c	55	1495	1485	**1.14**	0.94	0.91	**1.007**	**54.36**	**1.00**	0.18
	SSG-i	55	1495	1485	1.14	0.94	0.91	**1.007**	**54.36**	**1.00**	0.18
	CNM	3905	13455	11255	0.44	0.46	0.43	0.002	6.89	19.00	**0.56**
IMDB	SIMP-c	137	1037	837	**1.05**	0.46	0.43	0.111	**15.14**	4.00	1.46
	SIMP-i	85	560	425	0.91	**0.57**	0.53	0.157	13.18	4.00	**1.59**
	SIMP-m	86	543	451	0.91	0.56	**0.54**	0.149	12.63	**3.00**	1.07
	SSG-c	72	480	410	0.87	0.44	0.43	0.188	13.33	**3.00**	0.97
	SSG-i	11	18	16	0.13	0.13	0.12	**0.327**	3.27	4.00	0.18
	CNM	657	2397	2113	0.42	0.31	0.26	0.011	7.30	7.00	0.43

For each combination of dataset and property the best value obtained by any method is highlighted in bold

hill-climber finds subgraphs with large subjective interestingness scores. We empirically compare to (1) the modularity-based approach by Clauset et al. (2004) and (2) subjective interestingness for subgraphs (SSG) (van Leeuwen et al. 2016), as those are the closest to our approach and representative for the classes of methods they belong to. Note that neither is designed for mining patterns from multigraphs; we compare to these methods nevertheless to demonstrate that the task of mining patterns from multigraphs is very different from mining patterns from simple (unweighted or weighted) graphs in important ways, and therefore deserves the attention it gets in this paper.

Since SSG is designed for simple, unweighted graphs, the datasets are converted to simple graph by removing parallel edges. For fair comparison on the task of mining multigraphs, the evaluation criteria are computed on the original multigraph. For the method by Clauset et al. (2004), to which we will also refer as CNM, we use its implementation in iGraph,[8] which supports weighted graphs. We transform each multigraph to a simple, weighted graph by replacing each ‘multi-edge’ with a single edge, with the number of edges as weight. Further, to be able to designate a ‘most interesting pattern’ for CNM, the pattern giving the highest mean score according to SIMP-c, SIMP-i and SIMP-m is used. *Note that this comparison is very favourable for CNM’s method*, as we consider all patterns that the method generates, versus only the top-1 pattern discovered by SIMP (!). For synthetic data, we present averages over the most interesting patterns found on 50 different multigraphs, obtained using different seeds for multigraph generation.

Table 3 presents the results. The SI-c, SI-i and SI-m columns show that our proposed hill-climber, by optimising our multigraph interestingness measure on the multigraph data, was able to find subgraphs with higher scores than SSG and CNM, for all prior beliefs. The patterns found by SSG, however, are much smaller and have very few parallel edges, as witnessed by low values for γ. In general, all three of the proposed method—SIMP-c, SIMP-i, and SIMP-m—discover patterns with more parallel edges than the two baseline methods. For DBLP2 and DBLP4; CNM found patterns with the largest γ, but those

[8] https://igraph.org/.

Table 4 Properties of the top-10 patterns found by SIMP-c, SIMP-i, and SIMP-m, indicating the total computation time, the fraction of the vertices of the multigraph covered by all patterns combined, and the average Jaccard distance between all pairs of vertex sets

DataSet	Time (s)			Coverage			AvgJaccard		
	SIMP-c	SIMP-i	SIMP-m	SIMP-c (%)	SIMP-i (%)	SIMP-m (%)	SIMP-c	SIMP-i	SIMP-m
SYN1	6.93	6.51	7.49	32.77	21.82	24.17	0.95	0.90	0.96
SYN2	312.5	61.3	108.6	27.97	16.01	18.16	0.93	0.95	0.97
SYN3	2674	2394	2462.9	11.34	8.78	9.89	0.97	0.98	0.99
SYN4	8634	8435	8876	8.57	6.54	7.12	0.94	0.97	0.98
DBLP1	871.8	828.8	835.6	3.09	3.23	2.98	0.99	1.00	0.98
DBLP2	1025	1014	1024	3.16	3.08	3.18	1.00	1.00	1.00
DBLP3	7443	7828	7522	2.66	2.53	2.58	0.97	0.94	0.98
DBLP4	12,659	11,765	11,828	1.08	1.04	1.05	1.00	1.00	1.00
IMDB	493.8	215.1	276.5	12.64	6.54	6.98	0.91	0.94	0.90

patterns are very large and sparse, indicating that these are hardly informative. For some of the DBLP and IMDB datasets, the advantage of SIMP is quite large in terms of γ. Finally, the patterns found by SIMP-c, SIMP-i, and SIMP-m do not typically have a high density (ρ), which demonstrates that the proposed measure is different from ('objective') density.

Overall, it is shown that although SSG and SIMP are built on the same principles, they clearly quantify subjective interestingness of patterns differently, which leads to the identification of different patterns. While SIMP focuses on the occurrence of parallel edges, SSG only focuses on patterns with a smaller diameter. CNM provides similar results to SIMP-i, yet it yields large pattern as partitioning the dataset does not provide the user with an option to control the size of the patterns. Moreover, CNM's modularity measure necessarily always assign all vertices to a pattern, while SIMP-i can easily find few patterns containing only part of the graph.

It is also interesting to compare the results obtained by SIMP-c, SIMP-i, and SIMP-m. For almost all datasets, SIMP-c finds the pattern with the largest average multigraph degree, i.e., η, which is as expected since only a prior belief on the total number of edges in the network is assumed; all information on individual vertex degrees is assumed unknown. As expected, η is smaller for SIMP-i and SIMP-m results, and on the synthetic data SIMP-i and SIMP-m typically finds smaller subgraphs with larger densities and diameters. However, there is a trade-off among SIMP-c, SIMP-i, and SIMP-m for the measures ρ, η, $\mathbf{d}$ and γ, which demonstrates the flexibility of our proposed approach, where plugging in different prior beliefs lead to different results.

4.6 Qualitative evaluation

In this subsection, we first demonstrate how iterative pattern mining results different yet partially overlapping patterns, and then present an external validation of the patterns found on the IMDB dataset.

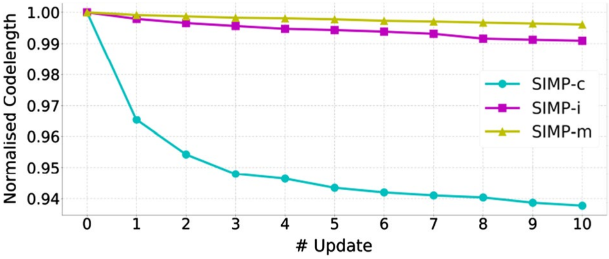


Fig. 4 Normalised code length of the IMDB dataset after each performed update, showing how each consecutive pattern adds new information to the set of patterns that is mined and therefore results in a shorter code for the data

4.6.1 Iterative pattern mining

As discussed in Sect. 3.4, our approach can be naturally utilized for iterative exploratory data mining to identify the top-K patterns in a multigraph. Table 4 shows the properties of the top-10 patterns found using SIMP-c, SIMP-i and SIMP-m. The patterns are evaluated based on total computation time taken to find the ten patterns, coverage (i.e., the percentage of all vertices in a multigraph dataset covered by the union of the found 10 patterns), and average Jaccard (AvgJaccard) distance among the found patterns. The total computation time is mainly dependent on the size of the dataset and the expected size of the pattern by the analyst (altered with the supplied parameter 'q' used in description length; not shown). The results show that the proposed approach can be easily used on moderately large datasets, with around two hours of computation time needed to find the top 10 patterns in the most densely connected graph, SYN4. This time includes the initial computation of the background distribution, searching for the most interesting pattern with ten independent runs (seeds) of the hill-climber, and updating the background distribution after each iteration. The coverage values indicate that the proposed method finds patterns in different regions of the graph; the exact coverage varies depending on the dataset, its topology and its size. At the same time, the high AvgJaccard value indicates that overlap is largely avoided but small overlaps among vertex sets do occur.

In terms of runtime, updating the background distribution hardly affects the performance of the algorithm. The main factor affecting this step is the computation of a Lagrangian multiplier corresponding to the found pattern, which is computed using the bisection method—in practice this method is very fast compared to the overall runtime of the algorithm. Updating the background distribution in every iteration is essential to the process, as we can demonstrate empirically. That is, by updating the background distribution, the code length of the data—i.e., the number of bits required to encode the data under the background distribution—is expected to decrease; this can be regarded to represent the effect of learning based on the found patterns.

To investigate this, Fig. 4 depicts the decrease in normalised code length of the IMDB dataset, for SIMP-c, SIMP-i, and SIMP-m, after each consecutive update of the background distribution. The code length of data $\mathbf{A}$ is given by $-\log_2 P(\mathbf{A})$, and in the plot, this is normalised by the code length of the data without any update, i.e., the length computed before learning but based only on the prior beliefs. We can observe that the negative loglikelihood of the data decreases over time, as the background distribution is updated using the found patterns. This clearly demonstrates how each consecutive pattern adds new information to the set of patterns that is mined. Further, the relative decrease in code length is larger for SIMP-c than for SIMP-i and SIMP-m, which is also completely in line with our expectations as Belief-i and Belief-m represent more elaborate forms of prior knowledge; hence there is less to learn from the data.

Table 5 Genres that are positively and significantly associated with the top-10 patterns found by SIMP-i, SIMP-m, SSG-i, and CNM, from the IMDB dataset, along with their respective Bonferroni corrected *p*-values (< 1e−4) (between brackets)

SN	SIMP-i	SIMP-m	SSG-i	CNM
1	Drama(0.0e+0), Crime(5.4e−10), Thriller(6.2e−16), Action(8.4e−6), Romance(2.2e−6)	Adventure(2.7e−12), Action(1.1e−5), Crime(2.8e−8)	Adventure(1.9e−7)	Adventure(2.1e−7), Drama(1.8e−14), Thriller(8.5e−8)
2	Adventure(0.0e+0), War(1.1e−12), Sci-Fi(5.9e−49), Action(1.1e−95), Family(2.0e−45), Thriller(0.0e+0), History(7.2e−10), Crime (9.0e−73), Romance(1.7e−96), Sport(8.7e−9), Biography(2.7e−20)	Sci-Fi(7.9e−12), Action(2.2e−8)	Drama(1.9e−6)	Comedy(1.3e−39)
3	Adventure(3.5e−61), Sport(6.3e−8), Sci-Fi(3.8e−36), Fantasy(2.4e−37), Family(3.6e−39), Action(1.3e−58), Crime(5.3e−52), Horror(1.5e−35), Thriller(1.8e−94)	Adventure(2.0e−5), Action(5.4e−9), Crime(1.2e−5)	—	Music(6.1e−11)
4	Romance(4.7e−10), Comedy(1.6e−11)	Adventure(1.7e−8), Fantasy(7.1e−9), Romance(2.0e−6)	—	—
5	Thriller(1.6e−22), Family(1.4e−7), Fantasy(6.0e−11), Sci-Fi(1.1e−11), Action(2.4e−13), Crime(7.3e−9), Comedy(1.4e−42), Adventure (4.2e−14)	Adventure(7.2e−5), Fantasy(2.0e−16), Family(3.3e−8)	Horror(6.3e−17)	—
6	Action(2.2e−11), Crime(6.0e−21), Sport(3.8e−13)	Sci-Fi(3.8e−15), Action(1.0e−5)	Adventure(2.2e−6), Action(8.4e−5)	—
7	History(8.6e−10), Crime(9.2e−5), Action(6.5e−15), Thriller(9.7e−12)	Adventure(1.1e−10)	Action(7.4e−13)	Action(1.4e−6)
8	Music(4.3e−11), Drama(1.7e−8)	Romance (6.1e−9)	Documentary (8.1e−7)	—
9	Action(4.3e−21), Horror(1.1e−7), Comedy(1.4e−40)	Action(5.3e−17), Crime(7.4e−7)	Western(5.8e−22)	—
10	Fantasy(1.2e−15)	Thriller(4.0e−6)	—	History(7.0e−11), Action(3.4e−12)

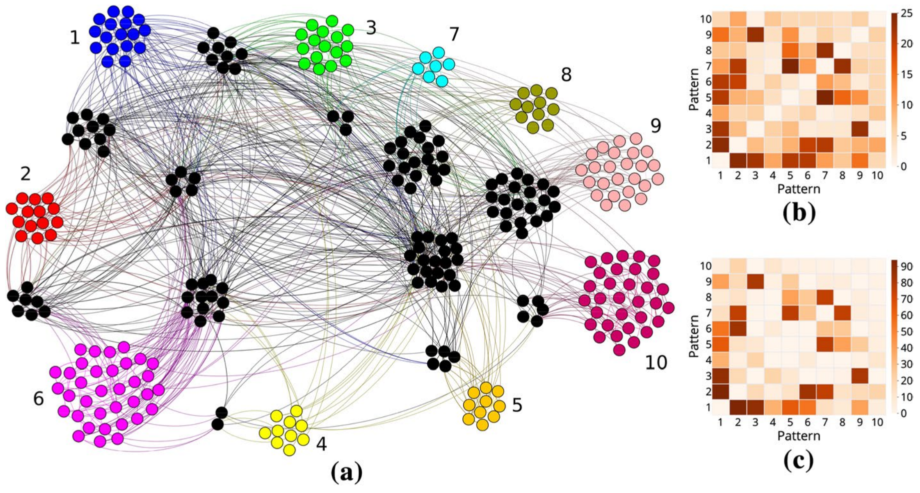


Fig. 5 Visualisation of the top-10 patterns (numbered as per Table 5) found by SIMP-m in the IMDB dataset: **a** network representation, with vertices present in more than one pattern shown in black colour (note that multiple edges between vertex pairs are depicted as a single edge to avoid cluttering the graph; see the other subfigures); **b** pattern overlap in terms of vertices: for each pair of the top-10 patterns, the heatmap shows the number of vertices that are part of both patterns, i.e., $|W_1 \cap W_2|$ for every two mined subgraphs H_1, H_2; and **c** pattern overlap in terms of edges: for each pair of the top-10 patterns, the heatmap shows the number of edges that are part of both patterns, i.e., $|E'_1 \cap E'_2|$ for every two mined subgraphs H_1, H_2

4.6.2 External validation

In Table 5 we investigate how the found patterns are different and whether they could be meaningful to a domain expert. In the IMDB co-actor network, each edge corresponds to a movie in which the two actors (represented by the vertex pair) have worked together. Clearly, this naturally fits the multigraph setting, as co-actors can work together in multiple movies and each movie can be of a different genre. Genre information is not considered in the construction of the dataset or the prior belief and we, therefore, use this attribute to externally and objectively validate the semantics of the found patterns. For validation, we consider 26 different genres and the top-10 patterns found by SIMP-i, SIMP-m, SSG-i, and CNM. For each combination of genre and pattern, we conduct a hypergeometric test to assess whether a genre is significantly associated with the pattern. We compute the corresponding p values and multiply them by the total number of tests per pattern, i.e., 26, as Bonferroni correction. All genres that are positively associated, i.e., have a p value smaller than the threshold of 1e−4, are shown for the top-10 patterns found by each of the four methods.

It is observed, in Table 5, mostly patterns found by SIMP-i and SIMP-m have more than one positively associated genre. This is mainly because of the presence of parallel edges that correspond to different genres; two actors can work together in numerous movies that belong to different genres. The patterns found by SSG-i are mostly associated with one or no genre. This is indicative of the fact that SSG, by definition, considers patterns with a smaller diameter as more interesting, which is different from the proposed approach for multigraphs. CNM, on the other hand, was able to find patterns with more than one significantly associated genre, but not every pattern was significantly associated with one or more genres. This might be explained by the fact that CNM partitions the entire graph into several communities, which results in relatively large patterns that do not correspond to certain genres. The results show that the patterns found by each method are different; both SIMP variants tend to find patterns that more strongly correspond to genres.

We further investigate the patterns found by SIMP-m by visualising the resulting patterns in Fig. 5. From the figures, we can observe that our approach succeeds in exploiting information about multiple edges between vertices, which results in the discovery of distinct yet partially overlapping patterns. From Table 5 we observe that patterns 1 and 3 are associated with the same set of genres, which might indicate that they are redundant or might be merged. Figure 5 shows that these patterns indeed share some vertices and edges, but are also different. Inspecting the data in more detail, we find that the actors with the highest degree in pattern 1 (but not in pattern 3) include Johnny Depp, Bruce Willis, Julia Roberts, and Robert Duvall. Similarly, actors present only in pattern 3 include Tom Hanks, John Ratzenberger, Delroy Lindo, and Sylvester Stallone. The overlapping region includes actors with very high degrees: Brad Pitt, J.K. Simmons, Morgan Freeman, and Kristen Dunst. Considering *actor's Facebook likes*, another feature present in the data, we find that the actors in pattern 1 (but not in pattern 3) have 8994 likes on average, versus 2973 on average for the actors in pattern 3 (not in pattern 1). The actors shared by both patterns on average have 104,53 likes. Further, we also find that the union of patterns 1 and 3 would give an SI-m of 0.503, which is clearly less than that of pattern 1, i.e., 0.538. All combined, the above analysis provides sufficient evidence to claim that pattern 1 and 3 indeed represent different, non-redundant 'actor communities', and are therefore rightfully considered to be two distinct patterns by our approach.

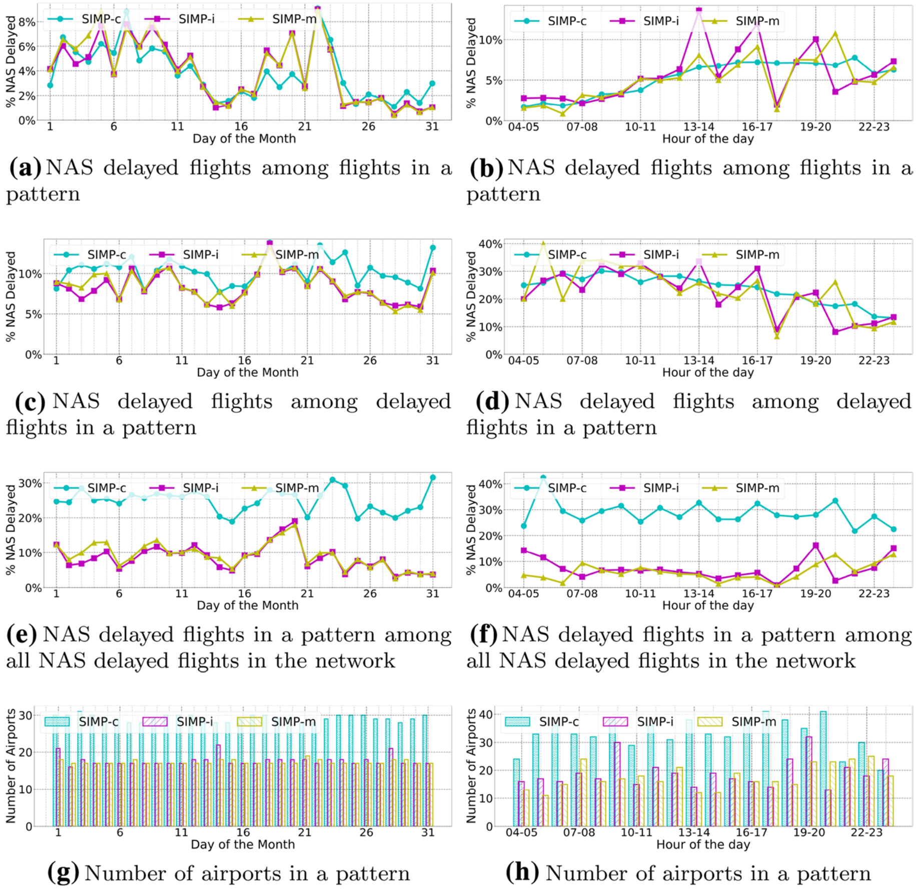
4.7 Airline case study

We now present a case study to showcase the application of SIMP in the aviation domain. More specifically, we use SIMP to analyze airline transport data taken from the Bureau of Transportation Statistics.[9] As discussed earlier in Sect. 1, such an airline dataset can be best represented as a directed multigraph. We focus on finding regions in the network that are likely to experience high delay due to heavy traffic, which is categorised in the data as NAS (National Aviation System) delay. There could be various factors for NAS delay, but heavy traffic is one of the major factors accounting for NAS delays.

We consider 298 commercial airports with 450,017 flights that took place in January 2017. As a first case, we investigate the most interesting patterns for each day over the period of the month of January 2017. For each day, we construct the background distribution based on prior beliefs taken from the flight *schedule data*; note that this is a very realistic scenario, as the schedule informs our expectations and we look for deviations from these expectations in the actual flight data. As a second case, we build the background distribution from scheduled data for each hour of a specific day, i.e., 22nd of January 2017. That is, we consider flights are either arriving or departing from any airport in any time block on the day, we have 20 time blocks of 1 h (from 0400 to 2400 h, all converted to UTC -7). We exclude cancelled flights from the data, as these would have an infinite delay.

The most interesting patterns per time frame found by SIMP are shown in Fig. 6. Figure 6a, b show that the patterns found by SIMP have a fairly large number of NAS delayed flights in the set of flights present in the found pattern. This shows that the first patterns

[9] Source: https://www.transtats.bts.gov/.



(a) NAS delayed flights among flights in a pattern

(b) NAS delayed flights among flights in a pattern

(c) NAS delayed flights among delayed flights in a pattern

(d) NAS delayed flights among delayed flights in a pattern

(e) NAS delayed flights in a pattern among all NAS delayed flights in the network

(f) NAS delayed flights in a pattern among all NAS delayed flights in the network

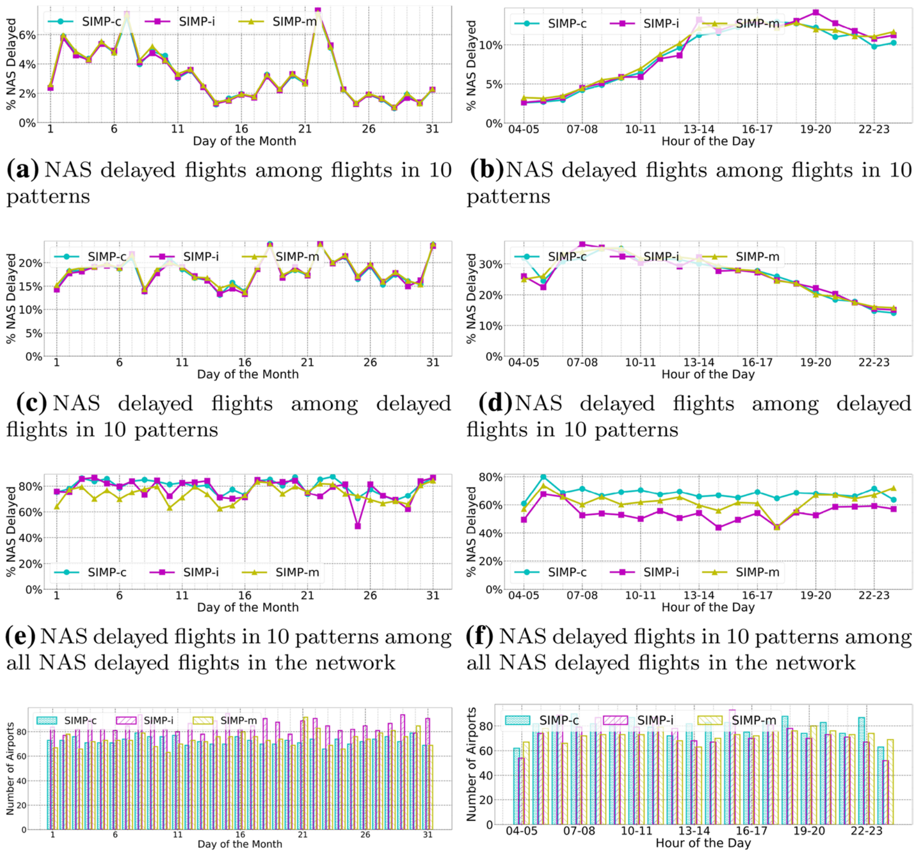
(g) Number of airports in a pattern

(h) Number of airports in a pattern

Fig. 6 Results of best pattern found by SIMP-c, SIMP-i and SIMP-m for two cases, i.e., (left) the entire month and (right) a single day

found by SIMP-i and SIMP-m have a fairly large 'precision', indicating that a fair number of the NAS delays occurs in these patterns. This is corroborated by Fig. 6c, d, which indicates that, among all delayed flights present in a pattern, a fair set of flights are categorised as NAS delayed. To verify that these patterns are the major source of NAS delay, we computed the 'recall' of the patterns in Fig. 6e, f, i.e., the number of NAS delayed flights present in the pattern among all NAS delayed flights in the current view of the network. It was found that SIMP-c has a fairly large recall, where around 25% of NAS delayed flights were present in around 10% of the airports of the network (see Fig. 6g, h). This is because of the large size of the patterns. Upon closely inspecting the patterns found by SIMP-i and SIMP-m, we found that these patterns all have a similar ratio of 'recall' to the percentage of airports in pattern, but have high 'precision', which supports our hypothesis that NAS delay is most likely to occur in the regions identified by SIMP.

Following the observations on the most interesting pattern per time frame, we analyse the top-10 patterns shown in Fig. 7. For this analysis the union of all top-10 patterns is considered, i.e., all the airports and flights that were present in any found pattern are taken together. Analysing the network over a period of a month, Fig. 7e shows that each day the



(a) NAS delayed flights among flights in 10 patterns

(b) NAS delayed flights among flights in 10 patterns

(c) NAS delayed flights among delayed flights in 10 patterns

(d) NAS delayed flights among delayed flights in 10 patterns

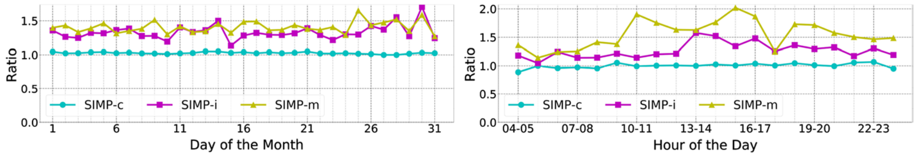
(e) NAS delayed flights in 10 patterns among all NAS delayed flights in the network

(f) NAS delayed flights in 10 patterns among all NAS delayed flights in the network

(g) Number of airports in all 10 patterns

(h) Number of airports in all 10 patterns

Fig. 7 Results of top 10 patterns found by SIMP-c, SIMP-i and SIMP-m for two cases, i.e., (left) the entire month and (right) a single day



(a) Ratio vs day of the month

(b) Ratio vs hour of the day

Fig. 8 Plots showing the ratio of % of NAS delays present in top-10 SIMP patterns to the % of NAS delays present in a baseline pattern having the same number of edges

top-10 patterns found by SIMP-c, SIMP-i and SIMP-m have a very high presence of NAS delayed flights among all the NAS delayed flights in the network on that day. A similar observation was made in Fig. 7f, while analysing the airline network, each hour for a single day. SIMP-c, SIMP-i and SIMP-m follow almost the same trend to account for NAS delayed flights in the top-10 patterns (Figs. 7a–d).

To further investigate this, we compute baseline patterns having the top-r airports with the highest multigraph degree, such that each such pattern has a number of edges (approximately) equal to the number of edges covered by the top-10 patterns found by SIMP. We then compute the ratio of the number of NAS delayed flights covered by the top-10 SIMP patterns to the number of NAS delayed flights covered by their respective baseline patterns, as shown in Fig. 8. This ratio is always close to one for SIMP-c, indicating that with this type of belief SIMP finds patterns with high densities, very similar to our constructed baseline patterns. SIMP-i and SIMP-m, on the other hand, have fairly high ratios, above 1 and sometimes close to 2, suggesting that these types of belief help in discovering patterns that correspond to NAS delays. These patterns may not always be structurally dense, i.e., their diameters may be high, but they encompass a large number of air routes with a larger number of flights. This shows the potential of using prior beliefs—such as the ones that we propose in this paper—for finding patterns that correspond to high traffic congestion, which may lead to NAS delays.

Overall, this exploratory case study shows that NAS delay is likely to occur in regions of the network that are *subjectively* interesting, i.e., relative to Belief-i and Belief-m. These patterns might provide strategic information to airliners in the context of flight scheduling.

5 Conclusions

We proposed a novel subjective interestingness measure for subgraphs in multigraphs, taking into account both the given multigraph and different types of prior beliefs that the analyst may have. For the background distributions we used existing ideas based on the maximum entropy principle, but to quantify interestingness for multigraph patterns we used the properties of the background distribution to derive an expected number of edges for each pair of vertices. Following this, we proposed an effective hill-climber algorithm for mining the most interesting pattern from the data. Our experiments demonstrated that our subjective interestingness measure for multigraphs is different from existing definitions for other types of graphs, highlighting the benefits of taking the specific properties of multigraphs into account. Further, our exploratory airline case study showed the potential relevance of the patterns and the advantage of being able to plug in background knowledge, such as flight schedule data. The proposed algorithm was naturally extended for iterative exploratory data mining process. Using this characteristic of the proposed algorithm a number of overlapping yet different patterns were shown to be found. Also, the proposed algorithm was found to be scalable and accurate in iteratively finding interesting patterns. A future direction is to extend our approach to dynamic multigraphs. We also anticipate to explore the application possibilities of the proposed algorithm in different domains.

Acknowledgements This research has been conducted as part of an Indo-Dutch project titled ‘A Systems approach towards Data Mining and Predictions in Airline Operation’ (SAPPAO), Grant No.: MIT-861-MID. We gratefully acknowledge the project sponsors MeitY (India), NWO (The Netherlands), and GE Aviation (India).

Appendix 1: Proof of probability distribution for belief-m

The problem of maximizing entropy under the user's belief about the number of edges per vertex and the number of neighbors per vertex is given as

$$\underset{P(\mathbf{A})}{\operatorname{argmax}} - \sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \log(P(\mathbf{A})), \tag{7}$$

$$\text{s.t.} \sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{v} a_{u,v} = d_u^r; \sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{u} a_{u,v} = d_v^c, \tag{8}$$

$$\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{v} 1_{a_{u,v}\neq 0} = m_u^r; \sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_{u} 1_{a_{u,v}\neq 0} = m_v^c, \tag{9}$$

$$\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) = 1. \tag{10}$$

Since this optimization problem is convex, we solve it using convex optimization methods (Boyd and Vandenberghe 2004). Let us introduce the Lagrangian multipliers λ_i^r and λ_i^c for constraints in Eq. 8, μ_i^r and μ_i^c for constraints in Eq. 9; and ψ for constraint 10. The Lagrangian of the Problems 7–10 is now given by

$$\begin{aligned} L(P(\mathbf{A}), \boldsymbol{\lambda}^r, \boldsymbol{\lambda}^c, \boldsymbol{\mu}^r, \boldsymbol{\mu}^c, \psi) = &- \sum_{\mathbf{A}} P(\mathbf{A}) \log P(\mathbf{A}) \\ &+ \sum_u \lambda_u^r \left(\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_v a_{u,v} - d_u^r \right) \\ &+ \sum_v \lambda_v^c \left(\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_u a_{u,v} - d_v^c \right) + \sum_u \mu_u^r \left(\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_v 1_{a_{u,v}} - m_u^r \right) \\ &+ \sum_v \mu_v^c \left(\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) \sum_u 1_{a_{u,v}} - m_v^c \right) + \psi \left(\sum_{\mathbf{A}\in\mathbb{N}_0^{n\times n}} P(\mathbf{A}) - 1 \right). \end{aligned} \tag{11}$$

The optimality conditions are achieved by equating the derivative of Eq. 11 w.r.t. $P(\mathbf{A})$ to 0. Hence, we get

$$P(\mathbf{A}) = \frac{1}{Z(\boldsymbol{\lambda}^r, \boldsymbol{\lambda}^c, \boldsymbol{\mu}^r, \boldsymbol{\mu}^c)} \exp\left(\sum_{u,v} a_{u,v}(\lambda_u^r + \lambda_v^c) + \sum_{u,v} 1_{a_{u,v}}(\mu_u^r + \mu_v^c) \right), \tag{12}$$

where $Z(\boldsymbol{\lambda}^r, \boldsymbol{\lambda}^c, \boldsymbol{\mu}^r, \boldsymbol{\mu}^c) = \exp(1-\psi)$ is a partition function. De Bie (2011) suggested that the choice of partition function is such to ensure the normalisation constraint 10. Similarly, here the partition function is also found to be the product of individual partition function represented by unique pair u and v, i.e., $Z(\boldsymbol{\lambda}^r, \boldsymbol{\lambda}^c, \boldsymbol{\mu}^r, \boldsymbol{\mu}^c) = \prod_{u,v} Z(\lambda_u^r, \lambda_v^c, \mu_u^r, \mu_v^c)$. Therefore, Eq. 12 now becomes

$$P(\mathbf{A}) = \prod_{u,v} \frac{1}{Z(\lambda_u^r, \lambda_v^c, \mu_u^r, \mu_v^c)} \exp\left(\lambda_u^r + \lambda_v^c\right)^{a_{u,v}} \cdot \exp\left(\mu_u^r + \mu_v^c)\right)^{1_{a_{u,v}}}. \tag{13}$$

This perfectly aligns with the proposition made by De Bie (2011), as here also $P(\mathbf{A})$ comes out to be the product of an exponential family distribution. Given the domain of $a_{u,v}$, the partition function is calculated as $Z(\lambda_u^r, \lambda_v^c, \mu_u^r, \mu_v^c) = \sum_{a_{u,v} \in \mathbb{N}_0} \exp(a_{u,v}(\lambda_u^r + \lambda_v^c) + 1_{a_{u,v}}(\mu_u^r + \mu_v^c))$ which results in $Z(\lambda_u^r, \lambda_v^c, \mu_u^r, \mu_v^c) = \frac{1-\exp(\lambda_u^r+\lambda_v^c)\left(1-\exp(\mu_u^r+\mu_v^c)\right)}{1-\exp(\lambda_u^r+\lambda_v^c)}$ such that $\lambda_u^r + \lambda_v^c < 0$. Finally, from Eq. 13 we get

$$\begin{aligned} P_{u,v}(a_{u,v}) = {} & \frac{\left[1 - \exp\left(\lambda_u^r + \lambda_v^c\right)\right]}{\left[1 - \exp\left(\lambda_u^r + \lambda_v^c\right)\left(1 - \exp\left(\mu_u^r + \mu_v^c\right)\right)\right]} \\ & \cdot \exp\left(\lambda_u^r + \lambda_v^c\right)^{a_{u,v}} \cdot \exp\left(\mu_u^r + \mu_v^c\right)^{1_{a_{u,v}}} \end{aligned}$$

□

Appendix 2: Proof of Theorem 1 (Claim 2)

The Lagrangian of Eqs. 5–6 is given as

$$L = \sum_{\mathbf{A}} Q(\mathbf{A}) \log\left(\frac{Q(\mathbf{A})}{P(\mathbf{A})}\right) + \lambda_H \left(|E'| - \sum_{\mathbf{A}} Q(\mathbf{A}) \sum_{u,v \in W} a_{u,v} \right) + \mu_H \left(1 - \sum_{\mathbf{A}} Q(\mathbf{A}) \right). \tag{14}$$

Thus, upon taking the derivative of L w.r.t. Q, such that $P'(\mathbf{A}) = Q(\mathbf{A})$ at $\frac{\partial L}{\partial Q} = 0$. Then, we get

$$\Rightarrow P'(\mathbf{A}) = \frac{P(\mathbf{A})}{Z'} \prod_{u,v \in W} \exp\left(\lambda_H\right)^{a_{u,v}}, \tag{15}$$

where $Z' = \exp\left(1 - \mu_H\right)$ where Z' is a new partition function. Now, using $P(\mathbf{A})$ as given in second part of Theorem 1, Eq. 15 becomes

$$P'(\mathbf{A}) = \frac{1}{Z'} \prod_{u,v \in W} \exp\left(\lambda_H\right)^{a_{u,v}} \cdot \prod_{u,v} \frac{1 - R_{u,v}}{1 - R_{u,v}(1 - S_{u,v})} R_{u,v}^{a_{u,v}} S_{u,v}^{1_{a_{u,v}}}. \tag{16}$$

Let, $R'_{u,v} = R \cdot \exp(\lambda_H)$, hence Eq. 16 is further bifurcated as

$$P'(\mathbf{A}) = \prod_{u,v \in W} \frac{1}{Z'} \frac{1 - R_{u,v}}{1 - R_{u,v}(1 - S_{u,v})} [R'_{u,v}]^{a_{u,v}} S_{u,v}^{1_{a_{u,v}}} \cdot \prod_{\neg u,v \in W} P_{u,v}(a_{u,v}). \tag{17}$$

Now, for partition function Z', we know that $\sum_{\mathbf{A}} P'(\mathbf{A}) = 1$ and also $a_{u,v} \in \mathbb{N}_0$. Thus, $Z' = \sum_{a_{u,v} \in \mathbb{N}_0} \frac{1-R_{u,v}}{1-R_{u,v}(1-S_{u,v})} [R'_{u,v}]^{a_{u,v}} S_{u,v}^{1_{a_{u,v}}} = \frac{1-R_{u,v}}{1-R_{u,v}(1-S_{u,v})} \cdot \frac{1-R'_{u,v}(1-S_{u,v})}{1-R'_{u,v}}$.

Now, putting in Eq. 17, we get

$$P'(\mathbf{A}) = \prod_{u,v\in W} \frac{1-R'_{u,v}}{1-R'_{u,v}(1-S_{u,v})}[R'_{u,v}]^{a_{u,v}} S_{u,v}^{1_{a_{u,v}}} \cdot \prod_{\neg u,v\in W} \frac{1-R_{u,v}}{1-R_{u,v}(1-S_{u,v})} R_{u,v}^{a_{u,v}} S_{u,v}^{1_{a_{u,v}}},$$

Hence, we can say $P'(\mathbf{A}) = \prod_{u,v\in V} \frac{1-R'_{u,v}}{1-R'_{u,v}(1-S_{u,v})} \cdot R'_{u,v}{}^{a_{u,v}} \cdot S_{u,v}^{1_{a_{u,v}}},$

where $R'_{u,v} = \begin{cases} R_{u,v} \cdot \exp(\lambda_H), & \text{if } (u,v) \in W \\ R_{u,v}, & \textit{otherwise} \end{cases}.$

Note The λ_H can be found using the bi-section method (Boyd and Vandenberghe 2004).

□

References

Andersen, R., & Chellapilla, K. (2009). Finding dense subgraphs with size bounds. In *International workshop on algorithms and models for the web-graph*, Springer, pp. 25–37.

Barabási, A. L., & Albert, R. (1999). Emergence of scaling in random networks. *Science*, *286*(5439), 509–512.

Batagelj, V., & Zaversnik, M. (2003). An o(m) algorithm for cores decomposition of networks. arXiv preprint arXiv:cs/0310049.

Blondel, V. D., Guillaume, J. L., Lambiotte, R., & Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, *2008*(10), P10008.

Boyd, S., & Vandenberghe, L. (2004). *Convex optimization*. Cambridge: Cambridge University Press.

Charikar, M. (2000). Greedy approximation algorithms for finding dense components in a graph. In *International workshop on approximation algorithms for combinatorial optimization*, Springer, pp. 84–95.

Clauset, A., Newman, M. E., & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, *70*(6), 066111.

De Bie, T. (2011). Maximum entropy models and subjective interestingness: an application to tiles in binary databases. *Data Mining and Knowledge Discovery*, *23*(3), 407–446.

Dong, X., Frossard, P., Vandergheynst, P., & Nefedov, N. (2012). Clustering with multi-layer graphs: A spectral perspective. *IEEE Transactions on Signal Processing*, *60*(11), 5820–5831.

Girvan, M., & Newman, M. E. (2002). Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, *99*(12), 7821–7826.

Khuller, S., & Saha, B. (2009). On finding dense subgraphs. In *International colloquium on automata, languages, and programming*, Springer, pp. 597–608.

Leicht, E. A., & Newman, M. E. (2008). Community structure in directed networks. *Physical Review Letters*, *100*(11), 118703.

Lijffijt, J., Spyropoulou, E., Kang, B., & De Bie, T. (2016). Pn-rminer: A generic framework for mining interesting structured relational patterns. *International Journal of Data Science and Analytics*, *1*(1), 61–76.

McClosky, B., & Hicks, I. V. (2012). Combinatorial algorithms for the maximum k-plex problem. *Journal of Combinatorial Optimization*, *23*(1), 29–49.

Newman, M. (2010). *Networks: An introduction*. Oxford: Oxford University Press.

Newman, M. E. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, *103*(23), 8577–8582.

Palla, G., Derényi, I., Farkas, I., & Vicsek, T. (2005). Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, *435*(7043), 814.

Papadopoulos, S., Kompatsiaris, Y., Vakali, A., & Spyridonos, P. (2012). Community detection in social media. *Data Mining and Knowledge Discovery*, *24*(3), 515–554.

Papalexakis, E. E., Akoglu, L., & Ience, D. (2013). Do more views of a graph help? Community detection and clustering in multi-graphs. In 2013 16th International conference on information fusion (FUSION), IEEE, pp. 899–905.

Pons, P., & Latapy, M. (2005). Computing communities in large networks using random walks. In *International symposium on computer and information sciences*, Springer, pp. 284–293.

Qi, G. J., Aggarwal, C. C., & Huang, T. (2012). Community detection with edge content in social media networks. In *2012 IEEE 28th International conference on data engineering*, IEEE, pp. 534–545.

Ruan, Y., Fuhry, D., & Parthasarathy, S. (2013). Efficient community detection in large networks using content and links. In *Proceedings of the 22nd international conference on World Wide Web*, ACM, pp. 1089–1098.

Silva, A., Meira, W, Jr., & Zaki, M. J. (2012). Mining attribute-structure correlated patterns in large attributed graphs. *Proceedings of the VLDB Endowment*, *5*(5), 466–477.

Tsourakakis, C., Bonchi, F., Gionis, A., Gullo, F., & Tsiarli, M. (2013). Denser than the densest subgraph: Extracting optimal quasi-cliques with quality guarantees. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, pp. 104–112.

Uno, T. (2010). An efficient algorithm for solving pseudo clique enumeration problem. *Algorithmica*, *56*(1), 3–16.

van Leeuwen, M., De Bie, T., Spyropoulou, E., & Mesnage, C. (2016). Subjective interestingness of subgraph patterns. *Machine Learning*, *105*(1), 41–75.

Xu, Z., Ke, Y., Wang, Y., Cheng, H., & Cheng, J. (2012). A model-based approach to attributed graph clustering. In *Proceedings of the 2012 ACM SIGMOD international conference on management of data*, ACM, pp. 505–516.

Zhou, Y., Cheng, H., & Yu, J. X. (2009). Graph clustering based on structural/attribute similarities. *Proceedings of the VLDB Endowment*, *2*(1), 718–729.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.