Dense subgraph mining with a mixed graph model

Anita Keszler*

Distributed Events Analysis Laboratory, Computer and Automation Research Institute (MTA SZTAKI), H-1111, Kende u. 13-17, Budapest, Hungary

Tamás Szirányi

Distributed Events Analysis Laboratory, MTA SZTAKI

Zsolt Tuza

Alfréd Rényi Institute of Mathematics, Hungarian Academy of Sciences
Department of Computer Science and Systems Technology, University of Pannonia

Abstract

In this paper we introduce a graph clustering method based on dense bipartite subgraph mining. The method applies a mixed graph model (both standard and bipartite) in a three-phase algorithm. First a seed mining method is applied to find seeds of clusters, the second phase consists of refining the seeds, and in the third phase vertices outside the seeds are clustered. The method is able to detect overlapping clusters, can handle outliers and applicable without restrictions on the degrees of vertices or the size of the clusters. The running time of the method is polynomial. A theoretical result is introduced on density bounds of bipartite subgraphs with size and local density conditions. Test results on artificial datasets and social interaction graphs are also presented.

Keywords: graph clustering, mixed graph model, dense subgraph mining, cluster seed mining, social graphs

*Corresponding author. Tel.: +36 12796106.
Email addresses: keszler.anita@sztaki.mta.hu (Anita Keszler), sziranyi.tamas@sztaki.mta.hu (Tamás Szirányi), tuza.zsolt@renyi.mta.hu (Zsolt Tuza)

1. Introduction

Data clustering is one of the most rapidly developing area of machine learning. Among the several main stream techniques (see Jain (2010) for a detailed introduction), graph based clustering methods have gained a lot of attention since the previous decades in numerous engineering applications (see for example Geva and Sharan (2011), Benchettara et al. (2010), Boykov and Kolmogorov (2004), Cousty et al. (2009), Du et al. (2008)), due to the modeling capabilities of graphs, and the large number of available theoretical results in this field (Schaeffer (2007)).

Considering the modeling, there are two major types of graph based clustering methods used in the field of pattern recognition: standard and bipartite graphs. Standard graphs model the objects to be clustered, bipartite graphs - with the two vertex classes - are eligible to model properties of the objects as well (Geva and Sharan (2011)). Some applications apply projection of the bipartite graph to standard graphs (e.g. Benchettara et al. (2010)).

Frequently applied methods using the standard model are graph partitioning and dense subgraph mining methods. Graph cuts (Boykov and Kolmogorov (2004), Danek et al. (2012), Cousty et al. (2010)), spectral partitioning, several MST-based clustering methods such as Zhou et al. (2011) belong to the partitioning methods. On the other hand, clique mining (Feige (2004)) is an example of density based methods.

In case of bipartite graph models, there also exist partitioning methods which divide one or both vertex classes into disjoint subsets (e.g. modularity-based

methods, such as Barber et al. (2008)), however dense subgraph mining methods - e.g. biclustering, dense bipartite subgraph mining (Du et al. (2008), Jancura and Marchiori (2010)) - are applied more often.

The advantage of the partitioning methods is their low computational cost (polynomial in the number of vertices). One of their drawbacks is that these algorithms are not able to deal with overlaps between clusters. Outliers cannot be handled either, therefore, pairwise similarities within a cluster cannot be ensured.

Density based methods are designed to overcome these drawbacks, but with exponential running time in the number of vertices - in general. In case of restrictions of vertex degrees, or limitations on the expected cluster sizes, there exist more efficient algorithms.

These methods are applied even if all the vertices are needed to be clustered. Dense subgraphs are considered as seeds of clusters, and the remaining vertices are clustered based on their similarities to the cluster seeds Du et al. (2008), Jancura and Marchiori (2010).

However, for bipartite graphs it is also proven that for a wide range of edge weights even finding good approximations of the maximum weight biclique in polynomial time is impossible (Tan (2008)). Due to computational complexity issues, methods based on random sampling have become popular (Mishra et al. (2003), Suzuki and Tokuyama (2005)), but there are severe restrictions on the size of the clusters in order to find them with high probability.

Despite the drawbacks, using bipartite graph based methods is important,

since besides clustering the objects, these have the potential of finding a subset of relevant properties as well, and with this gives a detailed description of the connection between the objects.

Our goal is to design an algorithm, that has the ability of detailed cluster descriptions as bipartite graph based methods, but with polynomial running time, without restrictions on the size of the clusters or the vertex degrees, and application of randomized methods. The capability of handling overlaps between clusters and outliers is also required. So the desired output is not only subsets of similar objects, but also subsets of properties, these objects (or a large fraction of them) agree on.

We accomplish this by a three-phase algorithm, where both standard and bipartite graphs are applied. The input is an object-property matrix, where each row represents an object, showing which properties it has. This matrix is converted into a standard weighted model (object distance graph), and a bipartite model (object-property graph). Phase 1 is a modified MSF-based clustering method on the standard weighted graph to find the seeds of the clusters. These seeds are only subsets of the real clusters. Phase 2 consists of two seed-refining step - one is carried out in the standard model, the other one in the bipartite model. The role of Phase 3 is the clustering of objects based on their similarities to the seeds.

The paper is organized as follows. In Section 2 some basic notations and definitions are presented. In Section 3 the steps of the proposed method are introduced. From Section 4 to 7 these steps are analyzed in details. Test results

of the algorithm are shown in Section 8. Section 9 presents the proof of a theoretical result for density bounds of subgraphs of bipartite graphs with size conditions.

2. Terminology and notation

Definition 1. *An undirected graph $G = (V_G, E_G)$ consists of the set of vertices or nodes (V_G), and E_G represents the edges.*

Definition 2. *A bipartite graph $G = (V, E) = (A, B, E)$ is a graph with two disjoint subset of vertices, such that $A \bigcup B = V$ and every edge connects a vertex in A to one in B.*

Definition 3. *Let G be a graph. If A is any subset of the vertex set, and v is any vertex, we denote by $N_A(v)$ the set of vertices adjacent to v in A.*

Definition 4. *Density of graphs. For a graph $G = (V, E)$ we define the* density *of G to be the quotient $\frac{|E|}{\binom{|V|}{2}}$. We also say that G has* local density *at least c (where c is any real number in the range $0 < c < 1$) if each vertex has degree at least $c(|V| - 1)$.*

Definition 5. *Density of bipartite graphs. For a bipartite graph $G = (V, E)$ with vertex bipartition $P \cup Q = V$, we define the* density *of G to be the quotient $\frac{|E|}{|P||Q|}$. We also say that G has* local density *at least c (where $0 < c < 1$) if each vertex $v \in P$ has at least $c|Q|$ neighbors in Q and each $v \in Q$ has at least $c|P|$ neighbors in P.*

Definition 6. *A connected component of a graph is a maximal subgraph such that any two vertices within are connected by a path (through a sequence of neighboring vertices).*

Definition 7. *An $F = (V_F, E_F)$ spanning tree of a $G = (V, E)$ is a spanning subgraph ($V_F = V$) and a tree (connected, cycle-free). A minimum weight spanning tree (MST) is a spanning tree with weight less than or equal to the weight*

of any other spanning tree. If the graph is not connected it contains a minimum spanning forest (MSF).

3. Steps of the proposed algorithm

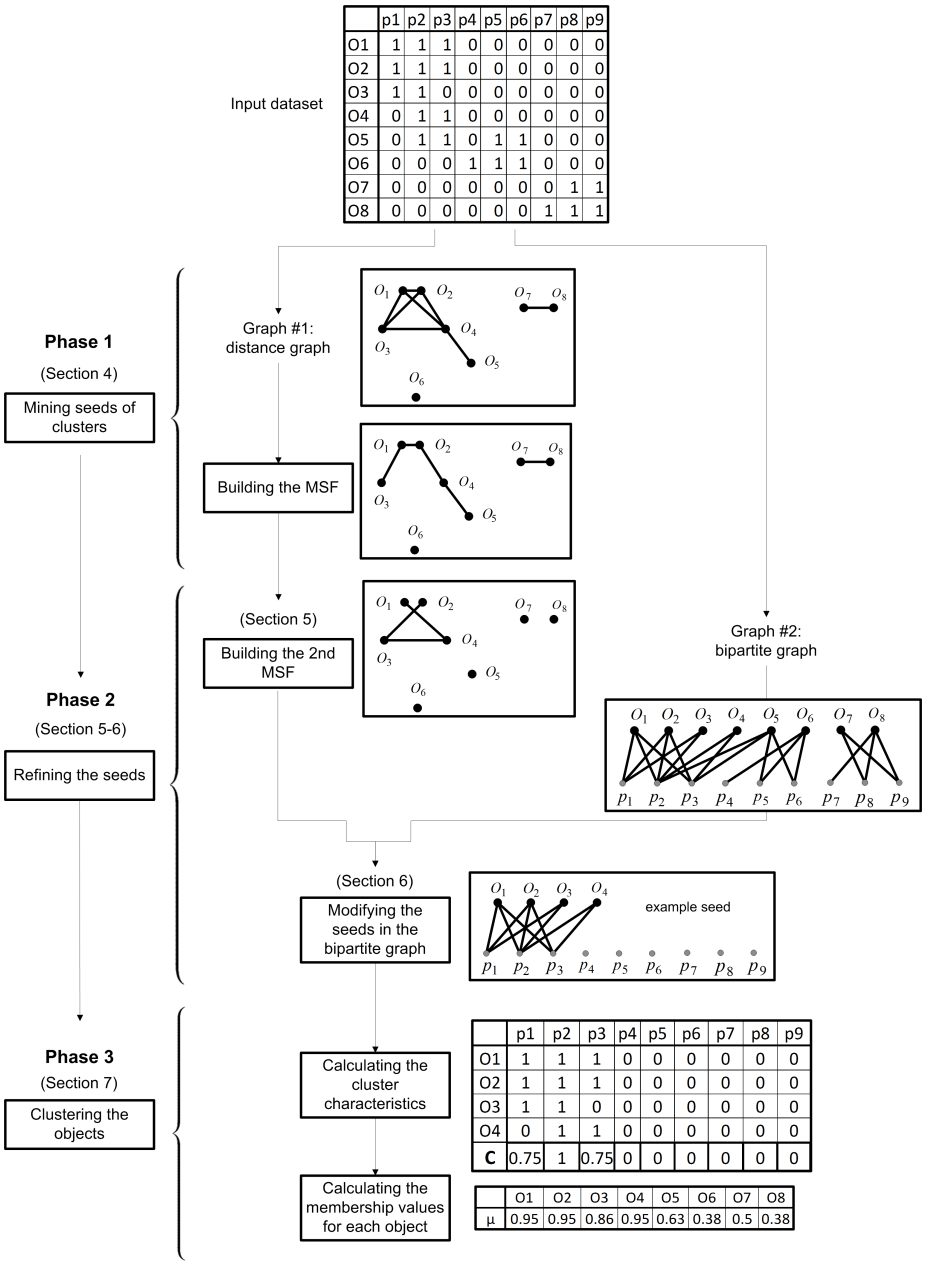
In this section we will give a short overview of the steps of the proposed algorithm (Figure 1).

Phase 1 is a cluster-seed mining process. The input is the data matrix, which is used to build a distance graph. Each object is represented by a row in the matrix, and each column corresponds to a property. The vertex set of the distance graph consists of the objects, the edgeweights show the similarities of the property vectors of the objects. The seeds are found by a MSF-based method.

Phase 2 is the refining of the seeds. The seeds are splitted if necessary, by a second MSF-based method. Then seeds are modeled in the bipartite graph with the corresponding properties. Properties that are not representative enough will be cut off. The output of this phase are the refined, bipartite seeds.

Phase 3 consists of computing the characteristic vectors of the seeds, and clustering the objects based on these characteristics. The output of the algorithm will be an object-cluster matrix, (in which each element shows how strongly a given object belongs to a given cluster) and the cluster labels of the vertices.

Our previous work (Keszler and Szirányi (2012)) was also based on using both standard and bipartite graphs on the same dataset. However, there are



	p1	p2	p3	p4	p5	p6	p7	p8	p9
O1	1	1	1	0	0	0	0	0	0
O2	1	1	1	0	0	0	0	0	0
O3	1	1	0	0	0	0	0	0	0
O4	0	1	1	0	0	0	0	0	0
O5	0	1	1	0	1	1	0	0	0
O6	0	0	0	1	1	1	0	0	0
O7	0	0	0	0	0	0	0	1	1
O8	0	0	0	0	0	0	0	1	1

	p1	p2	p3	p4	p5	p6	p7	p8	p9
O1	1	1	1	0	0	0	0	0	0
O2	1	1	1	0	0	0	0	0	0
O3	1	1	0	0	0	0	0	0	0
O4	0	1	1	0	0	0	0	0	0
C	0.75	1	0.75	0	0	0	0	0	0

	O1	O2	O3	O4	O5	O6	O7	O8
μ	0.95	0.95	0.86	0.95	0.63	0.38	0.5	0.38

Figure 1: Flowchart of the proposed algorithm.

several important improvements presented in this paper. There was only one round of MSF applied. The second round is an important change, since with this and the stopping condition we can avoid clustering problems illustrated on Figure 2, such as detecting paths as cluster seeds. The selection of the stopping condition is also an improvement. The algorithm applied for refining the seeds is proved to be convergent with a polynomial running time on the number of vertices (Section 6.2.2). One of the most important improvements compared to the former paper are the theoretical results on the density bounds (Section 9).

The advantage of this algorithm structure is that each phase or substep can be replaced by a different one without effecting the others.

4. Mining seeds of clusters

The first step of the seed mining phase is to build the distance graph of objects. The distance values are calculated from the similarities of the property vectors. In case of binary properties, the edgeweight is equal to the number of properties the two vectors do not agree on.

The seed mining method is a modified MST-based (see Definition 7) clustering, using Kruskal's algorithm.

The basic idea behind clustering with MST is that the vertices connected by edges of small weight in the tree are likely to be in one cluster. Previous methods usually work by finding the MST, then cutting edges until a certain criteria is satisfied. This criteria can be a weight threshold (e.g. Chowdhury and Murthy (1997), Vathy-Fogarassy et al. (2006), Yujian (2007), Wang et al.

(2009), Zhou et al. (2011)), the number of clusters(e.g. Xu et al. (2001), Jia et al. (2008), Peter (2012), Müller et al. (2012), one of the methods in Grygorash et al. (2006)), the size of clusters (Laszlo and Mukherjee (2005)), or some intra-cluster properties (e.g. Karthikeyan and Peter (2011), Goura et al. (2011)).

The above introduced papers were similar in the idea of first building the MST and then cutting edges by a clustering criteria. However, there exist a few bottom-up techniques as well.

An example of the bottom-up method is described in Felzenszwalb and Huttenlocher (2004) and is applied for image segmentation. The output of this algorithm is a partition of the vertex set.

Phase 1 of our algorithm also belongs to the bottom-up techniques. The main difference between our method and the one in Felzenszwalb and Huttenlocher (2004) is that our method is designed to handle outliers as well.

Our suggestion is to stop adding the edges when we reach the desired weight threshold, instead of building the complete MST and then cutting off edges. First we select a subgraph of the original graph by keeping the edges under the weight threshold, then run the MST finding algorithm on each component of the resulting graph. The advantage of this solution is that in the weight thresholded graph each component can be processed in parallel.

The construction of the weighted graph from the input matrix is done in $O(n^2 \cdot d)$, where n,d are the number of objects and properties respectively. The running time of Phase 1 is $O(|E| \cdot log|E|)$, since the edges are need to be sorted. This is common in case of MSF-based methods.

The pseudo-codes to produce an MSF of a graph (Algorithm 1), and to find seeds in the weight-thresholded graph are presented below (Algorithm 2). If no threshold value is given, $w_{th} = avg_{e \in E}(w(e)) + std_{e \in E}(w(e))$ will be used, where avg is the average value, std is the standard deviation of the edgeweights.

Algorithm 1 MSF($G = (V, E)$) — Minimum weight spanning forest

Require: Distance graph $G = (V, E)$

Ensure: $F = (V_F, E_F)$, a MSF of G.

1: $F = \emptyset$ {initialization}

2: $E = SortEdgeWeights(E)$ {sorts edgeweights in increasing order}

3: **for** $i = 1; i++; i \leq |E|)$ **do**

4: **if** $e_i \in E : F \bigcup e_i$ is cycle-free **then**

5: $F = F \bigcup e_i$

6: **print** F

The next two sections will present in details the second phase, where the seeds will be modified. First, a second MSF building step is carried out (Section 5), then the new set of seeds are processed in the bipartite graph (Section 6).

5. Refining the seeds - Building the 2nd MSF

Here, we apply a second MSF-building step, see Algorithm 3. The second MSF round is carried out by running Algorithm 2 on each seed found by the first round (Figure 1, Phase 2, step 1). The input of Algorithm 3 is a seed, and the corresponding MST. The edges of this MST will be removed, and the algorithm will be run on the remaining edge set. The new stopping condition will be

Algorithm 2 FINDSEED(G, w_{th}) — For finding cluster seeds in the distance graph

Require: Distance graph $G = (V, E)$, w_{th} edge weight threshold (optional)

Ensure: $G' = (V', E')$, such that $V' = V$, and $\forall e \in E' : w(e) \leq w_{th}$; and $F = (V_F, E_F)$, a MSF of G'.

1: if w_{th} is not given, $w_{th} = avg_{e \in E}(w(e)) + std_{e \in E}(w(e))$; $V' = V$;$E' = \emptyset$. {initialization}

2: **while** $\exists e \in E : w(e) \leq w_{th}$ **do**

3: $E' = E' \bigcup e$

4: F=MSF($G' = (V', E')$) {calling Algorithm 1}

5: **print** G', F

calculated from the edge set of the first MST. The output of the algorithm run on a seed will be a set of new seeds, since the original one might be splitted.

The threshold modification and the edge deletions are done in $O(|E|)$ for a seed, and it can be carried out in parallel for each seed, so the running time of this step is $O(|E|)$.

In Zhong et al. (2010) the authors also present a method of applying MST building twice. The input of the second MSF algorithm is the original graph without the edge set of the first MST. A second graph is built from the two MST edge set, and vertices are separated by graph-cut.

Test results of the seed mining process, and the seed modification process in the weighted standard graph are presented on Figure 2. The input dataset is a weighted graph, the output are the seeds after the second round of MSF mining.

Algorithm 3 MODIFYSEED(G', F) — For refining seeds in the distance graph

Require: $\{C_1, C_2, ..., C_{N_C}\}$:= components of G' (the output of Algorithm 2)

Ensure: $S = \{S_1, S_2, ...\}$ set of cluster seeds

1: **for** $i = 1; i++; i \leq N_C)$ **do**

2: $F_{2i} = \emptyset$ {initialization of the MSF for each component in G'}

3: $w_{th2} = avg_{e \in E_F}(w(e)) + std_{e \in E_F}(w(e))$

4: **for** $i = 1; i++; i \leq N_C)$ **do**

5: F_{2i} =FINDSEED($(V_{C_i}$,$E_{C_i} \setminus E_F), w_{th2}$) {calling Algorithm 2}

6: **print** S

The artificial input test datasets are illustrated on Figure 2(a). These test sets were constructed based on the typical distance based clustering problems mentioned in Zhong et al. (2010) and in Zahn (1971).

On Figure 2(b-e) the results of the first (left figures with red edges) and second (right figures with black edges) MSF rounds are shown for each input graph. After the second round, only the dense regions remain connected. The method can handle outliers (in contrast to graph partitioning methods), and applicable in case of cluster seeds of different sizes.

The drawback of several MST-based methods is that paths with small distances between the neighboring vertices are detected as clusters. With our approach, these types of subgraphs will not be detected as dense regions, see Figure 2(e). This is the result of the modified threshold value in case of the second MSF round.

The second frequently appearing drawback of this type of algorithms, is that overlapping clusters cannot be handled. This problem will be dealt with in Phase 3 (see Section 7). At this phase, the cluster seeds are disjoint subsets of the vertex (object) set.

Note that an object connected strongly to its neighboring objects might be removed after the second MSF iteration. However, if this object belongs to that dense region, it will be re-clustered in Phase 3. Examples will be presented in Section 8.

6. Refining the seeds - Modifying the seeds in the bipartite graph

The seed mining phase, and the first step of the seed refining process is finished. The next step is to model each seed as a bipartite graph, for further analysis. One vertex class will be formed by the objects of the seed, and the other one by the corresponding properties. The analysis consists of finding objects and properties that do not belong strongly enough to the seed. This is done by dense bipartite subgraph mining within each seed (Figure 1, Phase 2, step 2).

6.1. Previous methods

Since finding bipartite cliques (bicliques) or counting them is an NP-complete problem (Kutzkov (2012)), some relaxations are need to be made in order to achieve lower computational complexity. Otherwise only exponential running time algorithms exist, for example Zhang et al. (2008).

In Du et al. (2008) the authors present a method with a two-level clustering: first a seed mining step is carried out, then the remaining vertices are clustered. A bipartite graph is used for both steps, and the seeds are defined as the maximal bicliques. The running time of their method is $O(|E|^2)$ on sparse graphs, however it is exponential in general. Other solutions, such as Tanay et al. (2002) or Dourisboure et al. (2009) reach polynomial running time by assuming limited (constant) vertex degrees. In Geva and Sharan (2011) the biclique mining process is completed with a greedy expansion step. But within the seed identification step, only small subsets of vertices are taken into consideration. If it is not necessary to gain overlapping clusters, further simplifications can be made (Suzuki and Wakita (2009)).

The size of the cluster might also be interesting, as in case of biclustering gene expression data (Mitra and Banka (2006)). If the expected size of the cluster is large enough compared to the whole dataset, random sample based methods are also applicable, e.g. Mishra et al. (2003).

6.2. Our dense bipartite subgraph mining method

We present known density bounds of subgraphs in bipartite graphs, then we introduce our dense bipartite subgraph mining method with a corresponding new theoretical result. The Dense Bipartite Subgraph lemma presents a lower bound on the reachable density value of a subgraph in a bipartite graph, with size conditions, however in applications this limit can be significantly higher.

Our approach for finding seeds is also a two-level method, such as Du et al. (2008), however for the first phase a standard graph is used, and the cluster seeds

in their method form complete bipartite subgraphs (bicliques). Our method is applicable regardless of the size or number of clusters. The running time of our method is quadratic in the number of vertices, see Section 9.

The final seeds will still be disjoint considering the object side of the bipartite graph, however overlaps between the property sets of the seeds might occur. On Figure 3 (b) the first seed shares properties with the second and the third one.

6.2.1. Density bounds of subgraphs in bipartite graphs

It is well known in graph theory that every graph of average degree d contains a subgraph of minimum degree at least $d/2$, and this bound is tight. Bipartite graphs with analogous properties can also be constructed.

Below we investigate the situation where, instead of prescribed minimum degree, we need to find a subgraph in which every vertex is required to be adjacent to at least a prescribed proportion of the other vertex class of the subgraph (Definition 5), and at least a positive given fraction is selected from each vertex class of the initial graph. Without the condition on the cardinalities of vertex classes, the problem would be rather simple because selecting any vertex together with its neighbors we obtain a subgraph (star) in which all vertices are completely joined with the other vertex class.

Dense Bipartite Subgraph Lemma. *Let c, r, and c' be reals such that $0 < r < c < 1$ and $c' \leq \frac{c-r}{1-r}$. Then every bipartite graph $G = (V, E)$ with density at least c contains a bipartite subgraph $G' = (V', E')$ with local density at least c', such that $|P \cap V'| \geq r|P|$ and $|Q \cap V'| \geq r|Q|$, where P and Q denote*

the vertex classes of G. Moreover, a subgraph G' satisfying these conditions can be found in polynomial (more precisely, quadratic) time. (The proof is presented in Section 9.)

6.2.2. Modifying the seeds in the bipartite graph

To obtain the final seeds, density restrictions are made for each vertex individually in both vertex classes of the seeds (local density condition, see Definition 5).

We apply Algorithm 4 on each seed, based on the principle that vertices (both objects and properties) not satisfying the degree constraint are successively removed. Note that removal changes the order of the corresponding vertex class, hence the situation may become better or worse for a vertex in the other class, depending on whether it was non-adjacent or adjacent to the vertex just removed. A check is performed, and deletions are only made if the density has grown.

The dense bipartite subgraph mining will be run on each seed, in parallel. After this step of the seed refining phase, each object will have a given proportion of the properties within each seed, and the same holds for the subset of properties belonging to that seed.

Once the algorithm stops, the degree constraints are automatically satisfied (otherwise the latest round of the **while** loops decreased n' and a further round will be performed). Hence, we need to show in the proof that this happens before any of the situations $|P'| < r|P|$ and $|Q'| < r|Q|$ occurs.

Algorithm 4 DENSEBIP(c, r, c') — Large locally dense bipartite subgraph (assuming $0 < r < c < 1$ and $0 < c' \leq \frac{c-r}{1-r}$)

Require: Bipartite graph $G = (V, E)$ with vertex classes P, Q and density at least c

Ensure: Bipartite subgraph $G' = (V', E')$ with vertex classes $P' \subseteq P$, $Q' \subseteq Q$, $|P'| \geq r|P|$, $|Q'| \geq r|Q|$, and local density at least c'

1: $P' := P$, $Q' := Q$ {initialization}
2: $n' := |P'| + |Q'|$
3: **while** $\exists x \in P' : |N_{Q'}(x)| < c'|Q'|$ **do**
4: $\quad P' := P' \setminus \{x\}$
5: **while** $\exists x \in Q' : |N_{P'}(x)| < c'|P'|$ **do**
6: $\quad Q' := Q' \setminus \{x\}$
7: **if** $|P'| + |Q'| < n'$ **then**
8: $\quad$ **return to** 2
9: **print** P', Q'

The running time of this step of Phase 2 is quadratic in the number of vertices of the bipartite graph modeling each seed (see Section 9). In case of an input matrix with size $n \times d$, the running time of this step is $O((n+d)^2)$. The process can be run in parallel on each seed as well.

The overall running time of Phase 2 (including Section 5) is $O(|E|)+O((n+d)^2) = O((n+d)^2)$, since $|E| = O(n^2)$.

This section completes the steps of the seed finding and refining phases of the algorithm. The last phase will be the clustering, where objects outside the seeds can also be clustered.

7. Clustering the objects

The output of Algorithm 4 is the final set of bipartite seeds. In this section we will present the idea of calculating the characteristics of clusters based on the seeds, and the method of calculating membership values for each object. As the final output, the algorithm provides an object-cluster matrix, in which each element represents the strength of connection between each object-cluster pair.

For each cluster, the characteristics is derived from the corresponding seed. In case of an $S = \{O_S, P_S, E_S\}$ seed, where O_S,P_S and E_S represents the set of objects, set of properties and set of edges respectively, the characteristics is calculated in the following way:

$$C_S(i) = \begin{cases} \sum_{o_j \in O_S} M_{ij}/|O_S|, & \text{if } p_i \in P_S \\ NULL, & \text{otherwise,} \end{cases} \tag{1}$$

where M is the input object-property matrix.

The membership values for the objects are derived from the similarities between the cluster characteristic vectors and their property vectors. The similarities are evaluated only for the properties belonging to the seeds. The membership value of object i with respect to the cluster with seed S_j is calculated as follows:

$$\mu_{ij} = \sum_{p_k \in P_{S_j}} |M_{ik} - C_{S_j}(k)| \tag{2}$$

If an object reaches a membership value as high as the minimum membership values of the objects of the corresponding seed, it will be clustered. The rest of the objects will not be clustered automatically. The minimum of the membership values necessary for clustering depends on the application. Since an object might reach the threshold of clustering in case of more than one cluster, overlaps might occur.

Since each object belongs to at most one seed, the time complexity of calculating the characteristics is $O(n \cdot d)$. (As Phase 2, this can be run in parallel on each seed.) Clustering the objects is done in $O(n^2 \cdot d)$, which is the algorithmic complexity of this phase.

With parallelization the overall running time of the three-phased method is: $O(|E| \cdot log|E|) + O(n+d)^2 + O(n^2 \cdot d) = O(n^3) + O(n+d)^2 + O(n^2 \cdot d)$.

8. Test results

In this section test results on synthetic and real-world datasets are also presented.

8.1. Synthetic example

An artificial test dataset is introduced on Figure 3a. The dataset was constructed in order to demonstrate the effectiveness of our method in finding similar objects, and in selecting relevant subset of properties (dense bipartite subgraphs).

The bipartite graph (26 objects, 24 properties) contains 2 bicliques ($O_{11}-O_{15}$ and $O_{16}-O_{20}$), and one with additional properties (O_1-O_{10}). The fourth subgraph is a counter example ($O_{21}-O_{26}$). These subgraphs are marked with black, the remaining edges (gray) were selected randomly.

On Figure 3 the results of the seed mining and refining steps are presented. The three dense regions were detected by our method, with the automatic threshold used in Algorithm 2 and 3. On Figure 3b the output of the second MSF round is shown: the seeds are highlighted in bold. Note that some objects of the dense regions were not selected (second seed), and the seeds contain additional properties.

The latter problem is solved in Phase 2 by applying Algorithm 4. The parameter r was set to 0.75, that is, at least 75% of the properties and objects in each seed are needed to be kept. (This setting depends on how dense and large subgraphs do we want to gain as clusters.) The output of this seed refining

step is presented on Figure 3c. The additional loosely connected properties were ruled out in case of the second and third seeds, however some remained in case of the first.

However, we still have lost objects, that should have been selected by the seed-finding step. This problem was mentioned at the end of Section 5, and is solved by Phase 3 of the algorithm. In case of each seed the characteristics and the membership values of each object-cluster pair were calculated. The results are presented on Figure 3d. Besides the original seed vertices, other objects are also clustered.

8.2. Application related datasets

8.2.1. Test results on DIMACS datasets

The method was also tested on real-world datasets (free-access DIMACS datasets (Dolphins (Lusseau et al. (2003)), Jazz (Gleiser and Danon (2003)), Football (Girvan and Newman (2002))), see Tables 1 and 2.

The Dolphins dataset describes the interaction between 62 dolphins. The object-property matrix is constructed as follows: the i^{th} row shows the dolphins which the i^{th} dolphin is interacting with (1 - interaction, 0 - no interaction). Our goal is to find subgroups of dolphins with dense connection systems. The Jazz dataset contains the co-operating network of 198 jazz musicians (2742 edges). The Football dataset describes the network of football games between 115 teams. The goal in both cases is finding dense regions within the dataset. In the head of each subtable the average density of the corresponding dataset is also noted.

Table 1 presents the results on the Dolphins dataset. The gained cluster seeds after the 2nd MSF round (Phase 2, step 1 of our method) are significantly denser compared to the average density of the dataset (Table 1a). The density of the final seeds (output of Phase 2) have been further increased. The results corresponding to the stopping condition for Algorithm 2 are highlighted in bold. The final clusters (Phase 3) are presented in Table 1b with the identifier of the dolphins. The dolphins appearing in both clusters are highlighted in bold.

Note that the seed refining steps of Phase 2 resulted in an increased density. Furthermore, the cluster density values of the suggested stopping condition are higher or at least as high as other settings below and above this threshold. The capability of handling outliers and overlaps between clusters are also illustrated in Table 1b.

Test results of the other two datasets are presented in Table 2.

8.2.2. Comparison with other methods

Our algorithm was compared to other clustering methods by using the commonly tested Southern Women dataset (Freeman (2003)), in what the social activities (14 events) of 18 women was documented, see Figure 5. The advantage of our method compared to Barber et al. (2008) and Suzuki and Wakita (2009) is the capability of handling overlaps between clusters, see Figure 5d. Du et al. (2008) also detects overlapping clusters, but the resulting densities are significantly lower than our results. However, their method clusters all objects, while ours detect outliers that did not correspond strongly enough to the clusters. The advantage of our seed mining method is that the seeds do not need

Table 1: Test results on real-world datasets.

Results with the stopping condition for the MSF building phase, see Algorithm 2) are highlighted in bold. Results of lower and higher threshold values are shown before and after this, respectively. The size parameter r was set to 0.75 (see Dense Bipartite Subgraph Lemma). The density of the final seeds are significantly higher than the average density of the dataset. Columns: number of seeds (N), number of objects within each seed (size), density after first refining step, final seed size and density.

(a) Dolphins dataset - Results of the two seed refining steps (see Phase 2).

Dolphins dataset - Average density 0.0827				
Seeds - 2^{nd} MSF round			Final seeds	
N	objects	density	objects	density
5	3	0.11	3	0.15
	4	0.15	4	0.20
	2	0.129	2	0.17
	2	0.129	2	0.17
	6	0.131	6	0.173
2	**9**	**0.11**	**9**	**0.15**
	18	**0.129**	**18**	**0.17**
1	47	0.10	47	0.125

(b) Dolphins dataset - Results of Phase 3 (final clusters). Dolphins appearing in both clusters are highlighted in bold.

Seed	Dolphins in two clusters
1st	**19**,22,24,25,30,**46**,**51**,52
2nd	14-19, 34-41, 44-46, 51

Table 2: Further test results on real-world datasets. Notation is the same as in Table 1

(a) Football dataset

Football dataset - Average density 0.0927				
Seeds - 2^{nd} MSF round			Final seeds	
N	objects	density	objects	density
11	8	0.096	8	0.126
	10	0.096	10	0.126
	11	0.093	11	0.123
	4	0.089	4	0.118
	8	0.094	8	0.124
	9	0.094	9	0.124
	11	0.1	11	0.13
	2	0.096	2	0.126
	9	0.1	9	0.13
	10	0.092	10	0.12
	2	0.091	2	0.12
9	**18**	**0.096**	**18**	**0.126**
	12	**0.094**	**12**	**0.124**
	13	**0.09**	**13**	**0.12**
	12	**0.093**	**12**	**0.122**
	8	**0.093**	**8**	**0.124**
	9	**0.094**	**9**	**0.124**
	11	**0.98**	**11**	**0.13**
	9	**0.093**	**9**	**0.122**
	9	**0.99**	**9**	**0.13**

(b) Jazz dataset

Jazz dataset - Average density 0.1399				
Seeds - 2^{nd} MSF round			Final seeds	
N	objects	density	objects	density
1	62	0.24	62	0.30
1	**128**	**0.186**	**128**	**0.235**
1	162	0.16	122	0.16
1	162	0.16	162	0.20

to be complete subgraphs, therefore it is applicable in the presence of noise as well.

The method of Du et al. (2008) was compared to ours on the example described in Section 8.1. Figure 4c presents the result of their method. The seeds in their version are maximal bicliques, and the figure shows the 14 largest ones. In this case the final clusters were the seeds themselves. The results clearly show, that although their clusters are denser than ours, they split the vertices into too many parts. In contrast with their method, ours is capable of contracting seeds (in Phase 3).

Another comparison was carried out on the Dolphins dataset presented in Section 8.2.1. The adjacency matrix of the bipartite graph and some examples of the seeds found by (Du et al. (2008) are presented on Figures 4d and 4e. Since the graph is sparse, and the overlap between the neighborhood of the dolphins is small, the biclique-enumeration based method finds a large number of small seeds. Due to the number of these seeds, only some of the largest are shown. Our method found two clusters, and the results were detailed in Table 1.

As a conclusion, the advantage of our method compared to modularity-based techniques is that it is able to find overlapping clusters or outliers as well. On the other hand, compared to the two-level biclique-mining method it is more suitable to work in case of noise or in sparse graphs, since our method can detect a dense subgraph (compared to the average density of the graph) even if it does not contain maximal bicliques. Also note that in case of dense graphs, enumerating all bicliques would be quite inefficient, in contrast to ours that has

polynomial running time regardless of the density.

9. Proof of the Dense Bipartite Subgraph Lemma

Here we present the proof of the Dense Bipartite Subgraph Lemma.

Suppose that the **while** loops are performed exactly k times during the algorithm. For $i = 1, 2, \ldots, k$ let p_i and q_i denote the number of vertices removed from P' and Q', respectively, in the ith round of the **while** loops. (Some of them, namely p_1, p_k, and/or q_k may be zero.) Let us further denote $p := |P|$, $q := |Q|$, $p' := |P'|$, $q' := |Q'|$. By assumption, $|E| \geq cpq$. We observe that

- removing the p_i vertices from P', fewer than $c'p_i(q - \sum_{1 \leq j < i} q_j)$ edges are deleted;
- removing the q_i vertices from Q', fewer than $c'q_i(p - \sum_{1 \leq j \leq i} p_j)$ edges are deleted.

These are direct consequences of the conditions given in lines 3 and 5 of the algorithm. When the algorithm stops, $|E_{del}|$, the number of edges deleted altogether is less than

$$|E_{del}| < \sum_{i \geq 1} c'p_i(q - \sum_{1 \leq j < i} q_j) + \sum_{i \geq 1} c'q_i(p - \sum_{1 \leq j \leq i} p_j) \tag{3}$$

The right hand side can be rewritten as

$$\begin{aligned} &c'(p_1 q + q_1(p - p_1) + p_2(q - q_1) + q_2(p - p_1 - p_2) + \ldots \\ &+ p_k(q - q_1 - \cdots - q_{k-1}) + q_k(p - p_1 - \cdots - p_k)) \end{aligned} \tag{4}$$

With further rearrangements, using that $p = \sum_{i\geq 1} p_i + p'$, and $q = \sum_{i\geq 1} q_i + q'$ we get:

$$\begin{aligned} |E_{del}| <& c'((p-p')q + (q-q')p - \\ & - (p_1 + \cdots + p_k)(q_1 + \cdots + q_k)) \end{aligned}$$

$$|E_{del}| < c'((p-p')q + (q-q')p - (p-p')(q-q'))$$

$$|E_{del}| < c'(pq - p'q') \tag{5}$$

Thus, the number of edges remaining in G' is

$$|E'| > cpq - c'(pq - p'q') = (c-c')pq + c'p'q'. \tag{6}$$

This $|E'|$ cannot exceed $p'q'$, hence after rearrangement we obtain

$$(c-c')pq < (1-c')p'q',$$

$$\frac{c-c'}{1-c'} < \frac{p'q'}{pq}. \tag{7}$$

On the other hand, if at least one of the inequalities $p' < rp$ and $q' < rq$ is valid, then we necessarily have $p'q' < rpq$ (because $p' \leq p$ and $q' \leq q$ always hold). Consequently, in that case we would have

$$\frac{c-c'}{1-c'} < r,$$

$$c - c' < r - rc',$$

$$c - r < c'(1-r),$$

$$c' > \frac{c-r}{1-r}, \tag{8}$$

contradicting the assumption of the lemma. Thus, both $p' \geq rp$ and $q' \geq rq$ are valid.

The conditions for executing the steps purely depend on vertex degrees, which can be evaluated in linear time; moreover, at most $(1-r)|V|$ vertices can be removed (i.e., $k \leq (1-r)|V|$ holds for the number of rounds for the **while** loops). Thus, the overall running time of the algorithm is polynomial (quadratic).

10. Conclusions

We have introduced a dense subgraph mining method in bipartite graphs using the advantages of both the standard and the bipartite graph models. The algorithm consists of three main phases: a seed mining in a standard graph, a seed refining phase both in the standard and bipartite model and a clustering phase. Our method is applicable for clusters of any size, and the number of clusters is not need to be fixed either. It is able to detect overlapping clusters and outliers in bipartite graphs such as dense bipartite mining methods (in contrast with graph partitioning techniques), but with polynomial running time. Test were run on synthetic and real-world datasets as well, presented in Section 8. Besides the clustering method, new theoretical results on density bounds of subgraphs in bipartite graphs with size and local density constraints are discussed as well. In the future, further analysis and tests on the optimal size of clusters will be carried out for more application areas.

References

Barber, M. J., Faria, M., Streit, L., and Strogan, O. (2008). Searching for communities in bipartite networks. In Bernido, C. C. and Bernido, V. C., editors, *Searching for Communities in Bipartite Networks: Proceedings of the 5th Jagna International Workshop*, volume 1021, pages 171–182. AIP Conf. Proc.

Benchettara, N., Kanawati, R., and Rouveirol, C. (2010). Supervised machine learning applied to link prediction in bipartite social networks. In *Proceedings of 2010 International Conference on Advances in Social Networks Analysis and Mining*, pages 326–330. IEEE.

Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(9):1124–1137.

Chowdhury, N. and Murthy, C. (1997). Minimum spanning treebased clustering technique: Relationship with bayes classifier. *Pattern Recognition*, 30(11):1919–1929.

Cousty, J., Bertrand, G., Najman, L., and Couprie, M. (2009). Watershed cuts: Minimum spanning forests and the drop of water principle. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:1362–1374.

Cousty, J., Bertrand, G., Najman, L., and Couprie, M. (2010). Watershed cuts: Thinnings, shortest path forests, and topological watersheds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:925–939.

Danek, O., Matula, P., Maska, M., and Kozubek, M. (2012). Smooth chan-vese segmentation via graph cuts. *Pattern Recognition Letters*, 33(10):1405–1410.

Dourisboure, Y., Geraci, F., and Pellegrini, M. (2009). Extraction and classification of dense implicit communities in the web graph. *ACM Trans. Web*, 3(2):7:1–7:36.

Du, N., Wang, B., Wu, B., and Wang, Y. (2008). Overlapping community detection in bipartite networks. In *Web Intelligence*, pages 176–179. IEEE.

Feige, U. (2004). Approximating maximum clique by removing subgraphs. *SIAM J. Discrete Math.*, 18(2):219–225.

Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *Int. J. Comput. Vision*, 59(2):167–181.

Freeman, L. C. (2003). Finding social groups: A meta-analysis of the southern women data. In *Dynamic Social Network Modeling and Analysis. The National Academies*, pages 39–97.

Geva, G. and Sharan, R. (2011). Identification of protein complexes from co-immunoprecipitation data. *Bioinformatics (Oxford, England)*, 27(1):111–117.

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

Gleiser, P. and Danon, L. (2003). Advanced complex systems. (6):565.

Goura, V. M. K. P., Rao, N. M., and Reddy, M. R. R. (2011). A dynamic clustering technique using minimum-spanning tree. *IPCBEE*, 7:66–70.

Grygorash, R., Zhou, Y., and Jorgensen, Z. (2006). Minimum spanning tree based clustering algorithms. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI06)*, pages 73–81.

Jain, A. K. (2010). Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666.

Jancura, P. and Marchiori, E. (2010). Dividing protein interaction networks for modular network comparative analysis. *Pattern Recogn. Lett.*, 31(14):2083–2096.

Jia, Y., Wang, J., Zhang, C., and Hua, X.-S. (2008). Augmented tree partitioning for interactive image segmentation. In *ICIP*, pages 2292–2295. IEEE.

Karthikeyan, T. and Peter, S. J. (2011). Edge connectivity based clustering through minimum spanning tree. 1(2):57–61.

Keszler, A. and Szirányi, T. (2012). A mixed graph model for community detection. *International Journal of Intelligent Information and Database Systems*, page In Press.

Kutzkov, K. (2012). An exact exponential time algorithm for counting bipartite cliques. *Inf. Process. Lett.*, 112(13):535–539.

Laszlo, M. and Mukherjee, S. (2005). Minimum spanning tree partitioning

algorithm for microaggregation. *IEEE Transactions on Knowledge and Data Engineering*, 17(7):902–911.

Lusseau, D., Schneider, K., Boisseau, O. J., Haase, P., Slooten, E., and Dawson, S. M. (2003). The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 54(4):396–405.

Mishra, N., Ron, D., and Swaminathan, R. (2003). On finding large conjunctive clusters. In *COLT*, pages 448–462.

Mitra, S. and Banka, H. (2006). Multi-objective evolutionary biclustering of gene expression data. *Pattern Recognition*, 39(12):2464 – 2477. Bioinformatics.

Müller, A. C., Nowozin, S., and Lampert, C. H. (2012). Information theoretic clustering using minimum spanning trees. In *DAGM/OAGM Symposium*, volume 7476 of *Lecture Notes in Computer Science*, pages 205–215. Springer.

Peter, S. J. (2012). Local density-based hierarchical clustering for overlapping distribution using minimum spanning tree. *International Journal of Computer Applications*, 43(12):7–11. Published by Foundation of Computer Science, New York, USA.

Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review*, 1(1):27 – 64.

Suzuki, A. and Tokuyama, T. (2005). Dense subgraph problems with output-

density conditions. In *ISAAC*, volume 3827 of *Lecture Notes in Computer Science*, pages 266–276. Springer.

Suzuki, K. and Wakita, K. (2009). Extracting multi-facet community structure from bipartite networks. In *Proceedings of the 2009 International Conference on Computational Science and Engineering - Volume 04*, CSE '09, pages 312–319, Washington, DC, USA. IEEE Computer Society.

Tan, J. (2008). Inapproximability of maximum weighted edge biclique and its applications. In *TAMC'08: Proceedings of the 5th international conference on Theory and applications of models of computation*, pages 282–293, Berlin, Heidelberg. Springer-Verlag.

Tanay, A., Sharan, R., and Shamir, R. (2002). Discovering statistically significant biclusters in gene expression data. In *In Proceedings of ISMB 2002*, pages 136–144.

Vathy-Fogarassy, A., Kiss, A., and Abonyi, J. (2006). Hybrid minimal spanning tree and mixture of gaussians based clustering algorithm. In *Foundations of Information and Knowledge Systems*, volume 3861 of *Lecture Notes in Computer Science*, pages 313–330. Springer Berlin / Heidelberg.

Wang, X., Wang, X., and Wilkes, D. M. (2009). A divide-and-conquer approach for minimum spanning tree-based clustering. *IEEE Transactions on Knowledge and Data Engineering*, 21:945–958.

Xu, Y., Olman, V., and Xu, D. (2001). Minimum spanning trees for gene expression data clustering. *Genome Informatics*, 12:24–33.

Yujian, L. (2007). A clustering algorithm based on maximal θ-distant subtrees. *Pattern Recognition*, 40(5):1425 – 1431.

Zahn, C. (1971). Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, pages 68–86.

Zhang, Y., Chesler, E. J., and Langston, M. A. (2008). On finding bicliques in bipartite graphs: a novel algorithm with application to the integration of diverse biological data types. In *Proceedings of the 41st Hawaii International Conference on System Sciences*, pages 473–481. IEEE Computer Society.

Zhong, C., Miao, D., and Wang, R. (2010). A graph-theoretical clustering method based on two rounds of minimum spanning trees. *Pattern Recognition*, 43(3):752 – 766.

Zhou, Y., Grygorash, O., and Hain, T. F. (2011). Clustering with minimum spanning trees. *International Journal on Artificial Intelligence Tools*, 20(1):139–177.

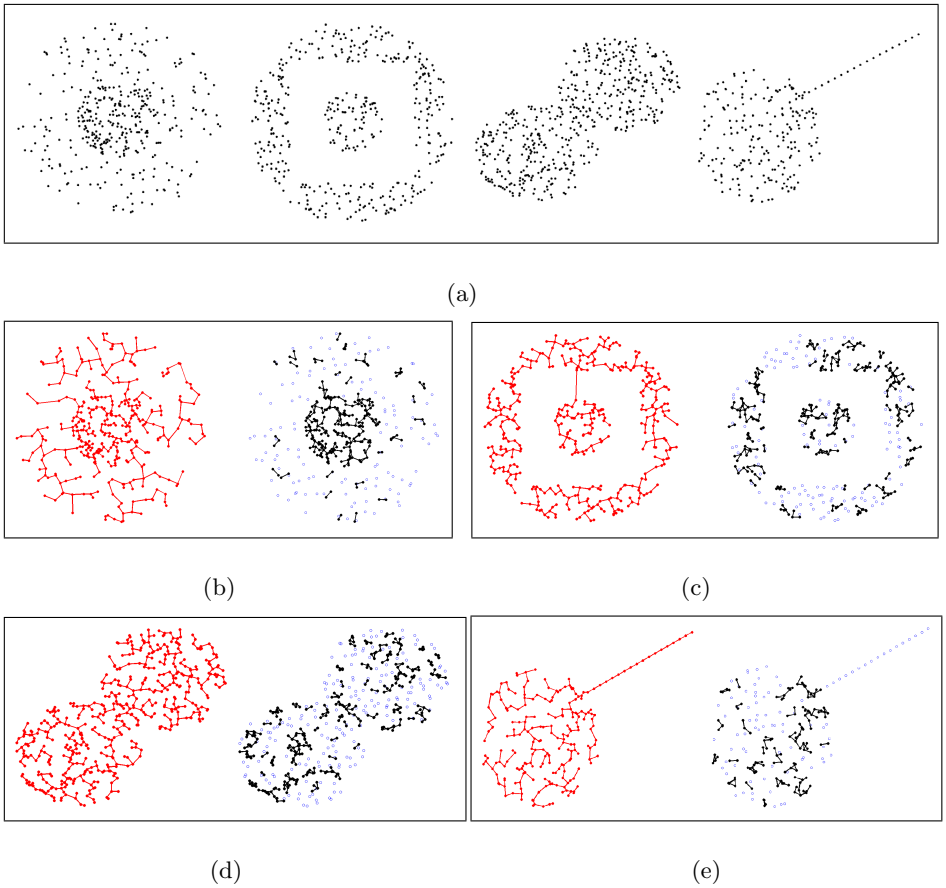
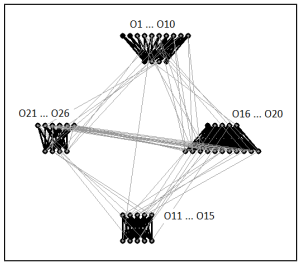
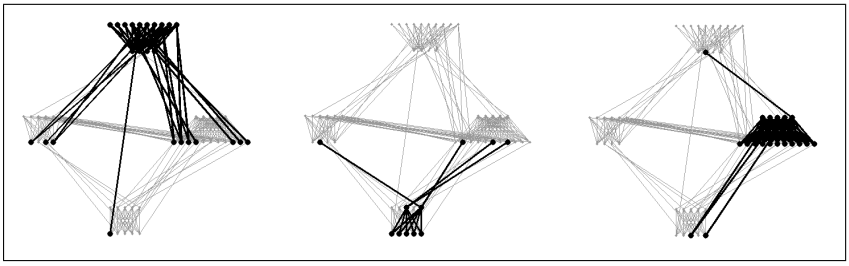


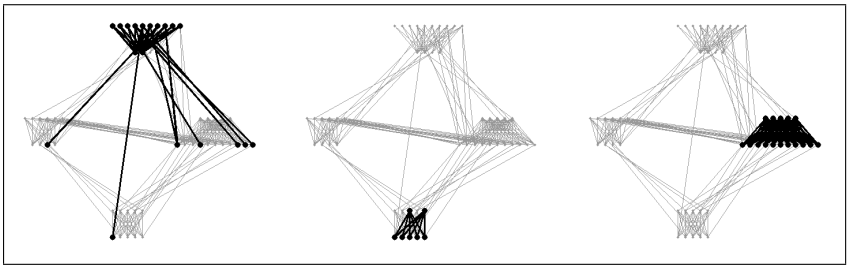
Figure 2: Test results of seed mining process. The four input graphs (a), and the result of the seed mining process (b-e).The output of the first MSF building phase are shown in red (left), the output of the second MSF building phase are shown in black (right). Only the densest regions remain connected.



(a) Test graph (26 objects, 24 properties).



(b) Seeds: output of the second MSF building step (Phase 2, step 1).

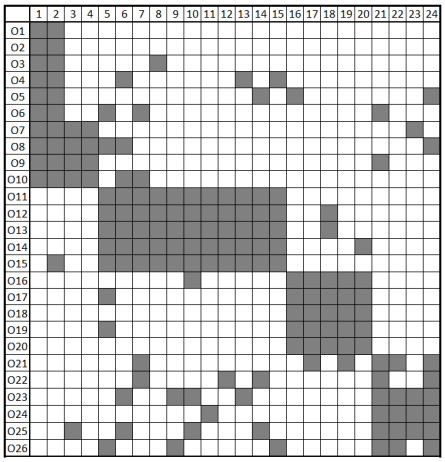


(c) Seeds after the seed refining process (Phase 2, step 2). Overlaps occur between the property set of the seeds.

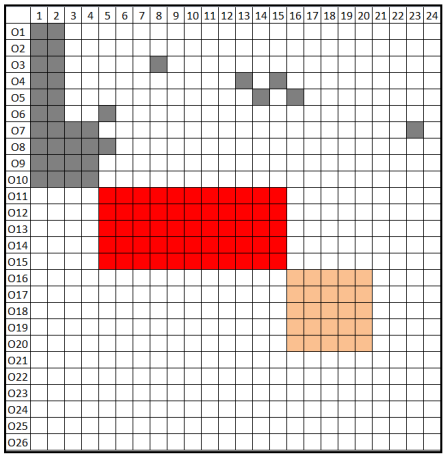
	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10	O11	O12	O13	O14	O15	O16	O17	O18	O19	O20	O21	O22	O23	O24	O25	O26
C1	0,96	0,96	0,91	0,87	0,87	0,92	0,91	0,92	0,96	0,96	0,41	0,36	0,36	0,36	0,47	0,52	0,46	0,47	0,4	0,52	0,68	0,74	0,59	0,69	0,64	0,66
C2	0,43	0,43	0,48	0,57	0,38	0,48	0,29	0,38	0,29	0,43	1	1	1	1	1	0,38	0,38	0,43	0,33	0,76	0,29	0,48	0,48	0,33	0,38	0,48
C3	0,61	0,61	0,56	0,5	0,61	0,44	0,5	0,44	0,5	0,44	0,28	0,33	0,33	0,33	0,22	1	1	1	1	1	0,72	0,5	0,56	0,61	0,56	0,5
Cluster ID	C1	C1	C1	C1	C1	C1	C1	C1	C1	C1	C2	C2	C2	C2	C2	C3	C3	C3	C3	C3						

(d) Final clusters($C1 - C3$). Cluster-membership values for objects $O_1 - O_{26}$. Seeds are marked in each cluster. O_{11}, O_{12} and O_{14} were also clustered, besides the seed of $C2$.

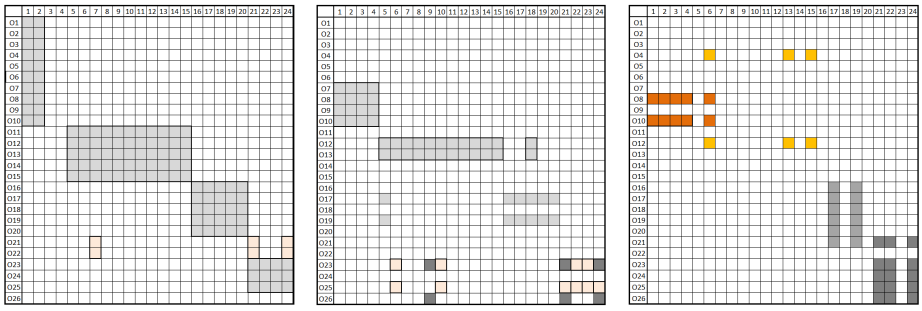
Figure 3: The output of our method phase by phase on a test graph.



(a) The adjacency matrix of the test graph on Figure 3a (edges are marked by dark gray cells).

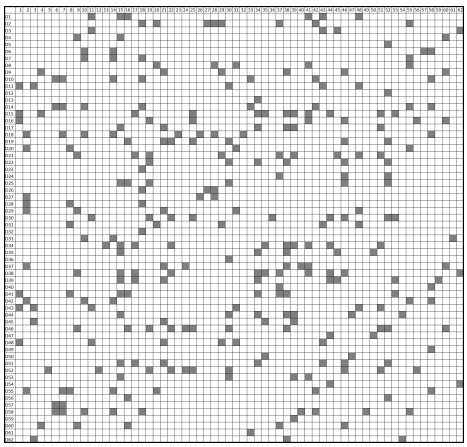


(b) The output of our method, the three clusters are marked in the adjacency matrix.

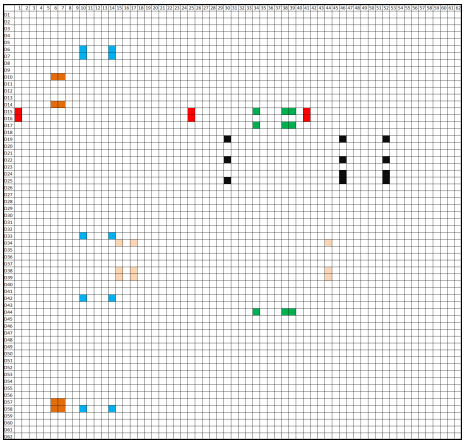


(c) Results of (Du et al. (2008))on input Fig. 4a. The 14 largest clusters are marked on three subfigures. Example: the light orange cluster on the right side is a biclique of objects O_4, O_{12} and properties p_6, p_{13}, p_{15}. The data set is highly over segmented.

Figure 4

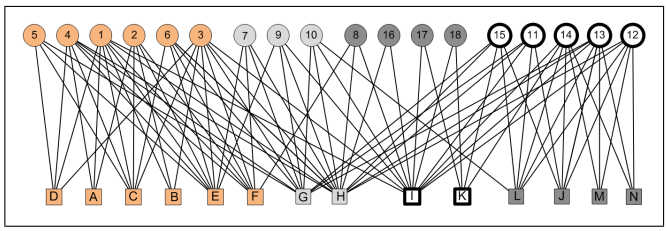


(d) Adjacency matrix of the Dolphins dataset. Dark cells denote the edges.

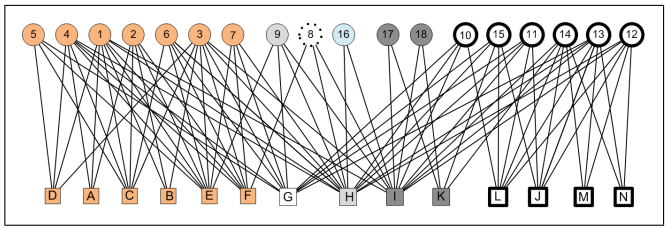


(e) Example cluster seeds of method Du et al. (2008) on the Dolphins dataset, marked by different colors.

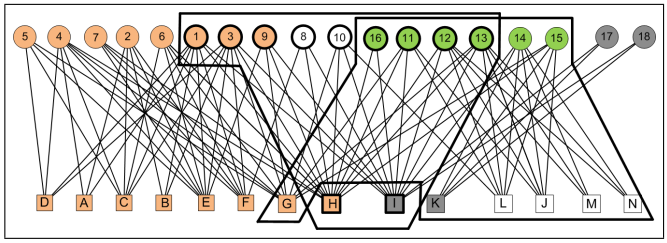
Figure 4: Test results on the artificial dataset presented on Figure 3a and on the DIMACS Dolphins dataset.



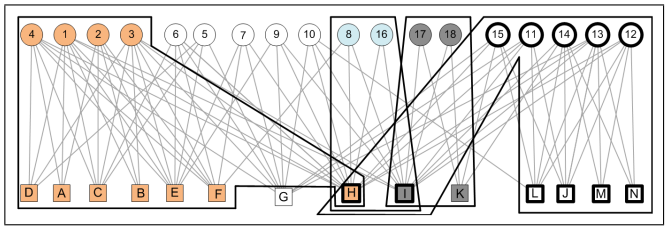
(a) Barber et al. (2008) - No overlaps or outliers.



(b) Suzuki and Wakita (2009) - No overlaps or outliers.



(c) Du et al. (2008) - Overlaps are handled. The density of clusters: 0.64, 0.66, 1 and 1.



(d) Result of our method. Overlaps are handled. The density of clusters: 0.8, 0.89, 1 and 1. Outliers were detected.

Figure 5: Test results on the Southern Women dataset. The object set contains 18 women, the property set models 14 social event they attended.