Overlapping correlation clustering

Francesco Bonchi Aristides Gionis Antti Ukkonen
Yahoo! Research Barcelona, Spain
Email: {bonchi,gionis, aukkonen}@yahoo-inc.com

Abstract—**We introduce a new approach to the problem of overlapping clustering. The main idea is to formulate overlapping clustering as an optimization problem in which each data point is mapped to a small set of labels, representing membership to different clusters. The objective is to find a mapping so that the distances between data points agree as much as possible with distances taken over their label sets. To define distances between label sets, we consider two measures: a set-intersection indicator function and the Jaccard coefficient.**

To solve the main optimization problem we propose a local-search algorithm. The iterative step of our algorithm requires solving non-trivial optimization subproblems, which, for the measures of set-intersection and Jaccard, we solve using a greedy method and non-negative least squares, respectively.

Since our frameworks uses pairwise similarities of objects as the input, it lends itself naturally to the task of clustering structured objects for which feature vectors can be difficult to obtain. As a proof of concept we show how easily our framework can be applied in two different complex application domains. Firstly, we develop overlapping clustering of animal trajectories, obtaining zoologically meaningful results. Secondly, we apply our framework for overlapping clustering of proteins based on pairwise similarities of aminoacid sequences, outperforming the of state-of-the-art method in matching a ground truth taxonomy.

I. Introduction

In many real-world applications it is desirable to allow overlapping clusters as data points may intrinsically belong to more than one cluster. For example, in social networks users belong to numerous communities. In biology, a large fraction of proteins belong to several protein complexes simultaneously, and genes have multiple coding functions and participate in different metabolic pathways. In information retrieval and text mining, documents, news articles, and web pages can belong to different categories.

In this paper we formulate overlapping clustering as the problem of mapping each data point to a small set of labels that represent cluster membership. The number of labels does not have to be the same for all data points. The objective is to find a mapping so that the similarity between any pair of points in the dataset agrees as much as possible with the similarity of their corresponding sets of labels.

While this idea is general and could be instantiated in different clustering frameworks, in this paper we apply it to the setting of *correlation clustering* [1], a clustering paradigm defined as follows: given a complete graph with positive and negative edges, the objective is to partition the graph so as to minimize the number of positive edges cut by the partition plus the number of negative edges not cut.

In our formulation, we still require a complete graph as input, but every edge is associated with a weight, which is a number in $[0, 1]$. Weights represent similarity between data points and the extent to which data points should be assigned to the same cluster. For defining distances between sets of labels, we consider two measures: a set-intersection indicator function and the Jaccard coefficient. We also constrain the maximum number of cluster labels allowed, either globally or per data point. These alternatives, together with the possibility of having fractional or binary edge weights, produces a whole family of problems.

In details, we make the following contributions:

- We define OVERLAPPING-CORRELATION-CLUSTERING, an optimization problem that extends the framework of correlation clustering to allow overlaps (Section II). We show that the problem we define is $\mathbf{NP}$-hard. We also discuss interesting connections of our problem with graph coloring and dimensionality reduction (Section III).
- We propose to solve the OVERLAPPING-CORRELATION-CLUSTERING problem using a simple local-search algorithm. The iterative step of the local-search algorithm optimizes the labels of one object, given the labels of all other objects. Applying this local optimization we iteratively improve the cost of the solution, until no further improvement can be made. We apply this general framework both variants of the problem, Jaccard-coefficient and set-intersection (Section IV).
- In the case of Jaccard coefficient, the iterative step of the local-search algorithm corresponds to a new problem, which we call JACCARD-TRIANGULATION. We prove that JACCARD-TRIANGULATION is $\mathbf{NP}$-hard, and we devise a method based on non-negative least squares, followed by post-processing of the fractional solution (Section IV-B). In the case of set-intersection, the sub-problem is named HIT-N-MISS. This is a set-cover type of problem, which we solve using a greedy algorithm (Section IV-C).
- We evaluate our algorithms on synthetic and real datasets. The real datasets are taken from the domains of spatio-temporal trajectory analysis and bioinformatics. Our evaluation shows that our algorithms produce overlapping clusters that resemble the ground truth, and outperform state-of-the-art methods. We also experiment with the idea of speeding up the algorithms by randomly eliminating pairs of data points from consideration. Our results show that significant amount of pruning can be achieved without degrading the quality of the solution (Section V).

The presentation of the paper is completed by surveying the related literature in Section VII and discussing future work in Section VIII.

II. Problem definition

We consider a set of n objects $V = \{v_1, \ldots, v_n\}$, over which we define a pairwise *similarity* function $s(u, v)$. For example, if V represents a set of documents, then $s(u, v)$ may be defined as the cosine between the vector representation of documents u and v; if V represents the tuples of a database table, then $s(u, v)$ may be defined as the fraction of attributes that the tuples u and v agree; etc. Hereinafter, we simply consider the values $s(u, v)$ as input to our problem and we do not make any assumption on how to obtain those values. We consider that the similarity function s takes values in the interval $[0, 1]$. We also study special cases of our problems in which the similarity function takes only values in the set $\{0, 1\}$.

Non-overlapping clustering. In non-overlapping clustering, we have in our disposal k cluster labels, denoted by $L = \{1, \ldots, k\}$, and the task is to assign cluster labels for each object in V. In order words, the clustering is defined by a labeling function $\ell : V \to L$. The objective is to assign labels to objects so that, to the largest possible extent, similar objects get assigned the same label. The *correlation-clustering problem* provides a precise formulation for such an objective.

*Problem 1 (*CORRELATION-CLUSTERING*):* Given a set of n objects $V = \{v_1, \ldots, v_n\}$ and a similarity function s over $V \times V$, find a labeling function $\ell : V \to L$ that minimizes the cost

$$C_{\mathrm{cc}}(V, l) = \sum_{\substack{(u,v) \in V \times V \\ \ell(u) = \ell(v)}} (1 - s(u, v)) + \sum_{\substack{(u,v) \in V \times V \\ \ell(u) \neq \ell(v)}} s(u, v). \quad (1)$$

The intuition underlying the above problem definition is that if two objects u and v are assigned to the same cluster we should pay the amount of their dissimilarity $1 - s(u, v)$, while if they are assigned to different clusters we should pay the amount of their similarity $s(u, v)$. In the binary case, which is the most widely studied setting for CORRELATION-CLUSTERING, Equation (1) expresses the cost function as the number of object pairs that have similarity 0 and are clustered together plus the number of object pairs that have similarity 1 and are clustered in different clusters.

In the traditional setting, no constraint on the maximum number of clusters is given, i.e., $|L| = \Theta(n)$. This is indeed one of the advantages of CORRELATION-CLUSTERING: the number of clusters is not required in the input, but "discovered" by the method.

Allowing overlaps. We now discuss how we extend the definition of CORRELATION-CLUSTERING in order to take into account overlapping of clusters. The main idea is to redefine the mapping function ℓ. Instead of mapping each object in a single cluster label $c \in L$, we relax the function ℓ so that it can map objects to any subset of cluster labels. So if $\mathcal{L}_+ = 2^L \backslash \{\emptyset\}$ denotes the collection of all subsets of L except the empty set, we now define the multi-labeling function ℓ to be $\ell : V \to \mathcal{L}_+$. If an object v is mapped under ℓ to a set of cluster labels $\ell(v) = \{c_1, \ldots, c_s\} \in \mathcal{L}_+$, then we say that v participates in all the clusters $c_1, \ldots, c_s$.

In a high-quality clustering, similar objects should be mapped to similar cluster labels. Thus, to evaluate a solution to overlapping clustering, we also need to select a similarity function H between sets of cluster labels, i.e., $H : \mathcal{L}_+ \times \mathcal{L}_+ \to [0, 1]$. We now have the necessary ingredients to define the problem of overlapping clustering.

*Problem 2 (*OVERLAPPING-CORRELATION-CLUSTERING*):* Given a set of n objects $V = \{v_1, \ldots, v_n\}$, a similarity function s over $V \times V$, and a similarity function H between sets, find a multi-labeling function $\ell : V \to \mathcal{L}_+$ that minimizes the cost

$$C_{\mathrm{occ}}(V, \ell) = \sum_{(u,v) \in V \times V} |H(\ell(u), \ell(v)) - s(u, v)|. \quad (2)$$

Our definition of clustering aims at finding a multi-labeling that "preserves" the similarities between objects. Note that considering the error term $|H - s|$ is meaningful since both functions H and s are similarity functions that take values in the range $[0, 1]$.

To make our problem concrete, we need to define the similarity function H between sets of cluster labels. In this paper, we consider two such functions: the Jaccard coefficient $J(E, F) = \frac{|E \cap F|}{|E \cup F|}$, and the set-intersection indicator function I:

$$I(E, F) = \begin{cases} 1 & \text{if } E \cap F \neq \emptyset \\ 0 & \text{otherwise.} \end{cases}$$

The Jaccard coefficient is a natural set-similarity function and it has been used in a wide range of applications. On the other hand, in certain applications, two objects sharing a single cluster label makes it sufficient to assert membership in the same cluster. In those latter cases, we use the set-intersection indicator function.

Constraints. So far we have assumed a finite alphabet of labels and hence a maximum number of clusters $|L| = k$. This can be seen as the typical constraint in which one needs to specify an upper bound in the total number of clusters. However, for many applications, while we may have in our disposal a large number of clusters we may not want to assign an object to all those clusters. For example, when clustering the users of a social network, we may want to use hundreds or even thousands of clusters, however, we may want to assign each user to a handful of clusters.

Thus, we consider a second type of constraint in which we require that each object v should be mapped to at most p clusters, that is, $|\ell(v)| \leq p$ for all $v \in V$.

III. Problem characterization

In this section we discuss the different variants of our problem, we establish its computational complexity, and we investigate connections with other problems.

First note that, based on our discussion from the previous section, the definition of Problem 2 is pertained by the following choices:

- the similarity function s may take values in the range $[0, 1]$ or it may take binary values in $\{0, 1\}$;
- the similarity function H may be the Jaccard coefficient J or the intersection indicator I; and
- we may impose the local constraint of having at most p cluster labels for each object or we may not.

Any combination of the above choices gives rise to a valid problem formulation. We systematically refer to any of these problems using the notation (r, H, p), where: $r \in \{\texttt{f}, \texttt{b}\}$ refers to the range of the function s: $\texttt{f}$ for fractional values and $\texttt{b}$ binary; $H \in \{J, I\}$ refers to the similarity function H: J for Jaccard coefficient and I for set-intersection; and p refers to the value of p of the local constraint, so $p = k$ means that there is no local constraint.

As an example of our notation, by $(\texttt{b}, H, k)$ we refer to two different problems, where s takes binary values, H can be either J or I, and there is no local constraint.

Hardness results. Our first observation is that all instances specified by $(r, H, 1)$ correspond to the CORRELATION-CLUSTERING problem defined in Problem 1. The reason is that when $|\ell(v)| = 1$ for all v in V, then both the Jaccard coefficient and the intersection indicator take just 0 or 1 values. In particular, $|H(\ell(u), \ell(v)) - s(u, v)|$ becomes $1 - s(u, v)$ when $\ell(u) = \ell(v)$ and $s(u, v)$ when $\ell(u) \neq \ell(v)$. Thus we easily establish that our problem is a generalization of the standard CORRELATION-CLUSTERING problem. Since CORRELATION-CLUSTERING is $\mathbf{NP}$-hard, and since $p = 1$ is a special case of any (r, H, p) problem, the previous observation implies that the general OVERLAPPING-CORRELATION-CLUSTERING problem is also $\mathbf{NP}$-hard. However, in order to show that the complexity of our problems does not derive exclusively from the hardness of the special case $p = 1$, we provide $\mathbf{NP}$-hardness results that do not rely on such special case.

Theorem 1: The problem instances (r, I, p), with $p > 1$, are $\mathbf{NP}$-hard.

Proof: We show that the $(\texttt{b}, I, p)$ problem is $\mathbf{NP}$-hard, which also gives the $\mathbf{NP}$-hardness for the $(\texttt{f}, I, p)$ problem. We obtain the reduction from the problem of COVERING-BY-CLIQUES [2, GT17], which is the following: given an undirected graph $G = (V, E)$ and an integer $C \leq |E|$, decide whether G can be represented as the union of $c \leq C$ cliques. We can show that a zero-cost solution to the $(\texttt{b}, I, C)$ problem identifies graphs having a covering by at most C cliques, and solutions with a cost larger than zero identify graphs that do not admit a covering by at most C cliques. Given an undirected graph $G = (V, E)$ we construct an instance of the $(\texttt{b}, I, C)$ problem by simply setting the set of objects to be the set of nodes V. For each edge $(u, v) \in E$ we set $s(u, v) = 1$, while if $(u, v) \notin E$ we set $s(u, v) = 0$. We also set the total number of cluster labels $k = C$. It is easy to verify our claim: a zero-cost solution of $(\texttt{b}, I, C)$ on input (V, s) corresponds to a covering of G by at most C cliques. ■

In addition, due to the inapproximability of the COVERING-BY-CLIQUES problem, we can deduce that the problem instances (r, I, p) do not admit polynomial-time constant factor approximation algorithms unless $\mathbf{P} = \mathbf{NP}$.

We now turn our attention in the problems (r, I, k), that is, using set-intersection and no local constraint. Moreover we consider the case in which we are allowed to use a very large number of cluster labels, in particular $k = \Theta(n^2)$.

Proposition 1: For the problem instances $(r, I, \Theta(n^2))$, the optimal solution can be found in polynomial time.

Proof: We start by giving each object a unique cluster label. Then we process each pair of objects for which $s(u, v) \geq \frac{1}{2}$. For any such pair of objects we make a new label, which we assign to both objects, and never use again. Thus, for pairs with $s(u, v) \geq \frac{1}{2}$, the intersection of $\ell(u)$ and $\ell(v)$ is not empty, and thus we pay $|1 - s(u, v)| \leq \frac{1}{2}$. On the other hand, for the pairs with $s(u, v) \leq \frac{1}{2}$, the intersection of $\ell(u)$ and $\ell(v)$ is empty, and thus we pay $|s(u, v)| \leq \frac{1}{2}$. Since I takes only 0/1 values, no other solution can cost less, and thus the previous process gives an optimal solution. ■

When we have binary similarities, the above process straightforwardly provides a zero-cost solution.

Corollary 1: The problem $(\texttt{b}, I, \Theta(n^2))$ admits a zero-cost solution that can be found in polynomial time.

Connection with graph coloring. Given that the problem $(\texttt{b}, I, k)$ admits a solution of zero cost if we allow enough cluster labels, we next ask which is the minimum number of cluster labels k needed for a zero-cost solution. We characterize such number by pointing out a connection with GRAPH-COLORING problem, whose formualtion we recall next. A proper coloring of a graph $G = (V, E)$ is a function $c : V \to \{1, \ldots, k\}$ so that for all $(u, v) \in E$ we have $c(u) \neq c(v)$. The GRAPH-COLORING problem asks to find the smallest number k, known as the chromatic number $\chi(G)$ of G, for which a proper coloring of G exists.

Going back to the binary $(\texttt{b}, I, k)$ OVERLAPPING-CORRELATION-CLUSTERING problem, given the set of objects V and similarity function s, we consider *similar pairs* $P^+ = \{(u, v) \in V \times V \mid s(u, v) = 1\}$ and *dissimilar pairs* $P^- = \{(u, v) \in V \times V \mid s(u, v) = 0\}$. Using these we define the graph $\widehat{G} = (P^+, \widehat{E})$, with similar pairs as nodes, and the set of edges $\widehat{E}$ given by the dissimilar pairs as follows:

$$\{((u, v), (x, y)) \in P^+ \times P^+ | \\ \{(u, x), (u, y), (v, x), (v, y)\} \cap P^- \neq \emptyset\}.$$

We have:

Proposition 2: The chromatic number $\chi(\widehat{G})$ of $\widehat{G}$ is equal to the minimum number of cluster labels required by a zero-cost solution to the $(\texttt{b}, I, k)$ problem with input (V, s).

Proof: We observe that a color in $\widehat{G}$ corresponds to a cluster in our problem. The colors are assigned to pairs of objects in V, which ensures that the positive pairs P^+ are satisfied. On the other hand, the constraint of having a proper coloring, ensures that the negative pairs P^- are also satisfied. Thus, a proper coloring on $\widehat{G}$ corresponds to a zero-cost solution on our problem. ■

Although the previous result is theoretically interesting, it has limited practical relevance, as we are interested in minimizing the error given a specific number of clusters.

To make the connection practically useful, we would need to relax the GRAPH-COLORING problem, so that it allows for a less strict definition of coloring. Namely, we would like to allow for colorings that for certain cost may allow the following relaxations: (i) $(u, v) \in E$ not necessarily implies $c(u) \neq c(v)$ – corresponding to violations on P^-; and (ii) nodes may be left uncolored – corresponding to violations on P^+. We believe that this is an interesting path that may lead to novel algorithms for OVERLAPPING-CORRELATION-CLUSTERING. We plan to investigate this research direction in our future work.

Connection with dimensionality reduction. We finally note the similarity of our problem formulation with the *dimensionality-reduction* problem, in particular multidimensional scaling. Dimensionality reduction is a problem that has been studied, among other areas, in theory, data mining, and machine learning, and has many applications, for example, in proximity search, feature selection, component analysis, visualization, and more. At a high level, one is given a set of points in a high-dimensional space. The goal is to map each point x in a point $h(x)$ in a much lower-dimensional space in such a way that for any pair of points x and y, their distance $d(x, y)$ in the high-dimensional space is preserved as well as possible in the lower-dimensional space by the distance $d(h(x), h(y))$. The connection of the above statement with Equation (2), which defined our OVERLAPPING-CORRELATION-CLUSTERING problem is apparent. However, the difference is that dimensionality-reduction methods typically are defined for geometric spaces. Alternatively, they operate by hashing high-dimensional or complex objects in a way that similar objects have high collision probability [3], [4]. However, to the best of our knowledge, the case that the projected space is a set-system and similarities are measured by a set distance function has not been considered before.

IV. ALGORITHMS

We propose a local-search algorithm that optimizes the labels of one object in the dataset, when the labels of all other objects are fixed. We apply this framework both for the Jaccard coefficient and the intersection-function variants of the problem, proposing novel algorithms for these two local optimization problems in Sections IV-B and IV-C, respectively.

A. The local-search framework

A typical approach for multivariate optimization problems is to iteratively find the optimal value for one variable *given* values for the remaining variables. The global solution is found by repeatedly optimizing each of the variables in turn until the objective function value no longer improves. In most cases such a method will converge to a local optimum. The algorithm we propose falls into this framework. At the core of our algorithm is an efficient method for finding a good labeling of a single object given a fixed labeling of the other objects. We can guarantee that the value of Equation 2 is non-increasing with respect to such optimization steps. First, we observe that the cost of Equation (2) can be rewritten as

$$\begin{aligned} C_{\mathrm{occ}}(V, \ell) &= \frac{1}{2} \sum_{v \in V} \sum_{u \in V \setminus \{v\}} |H(\ell(v), \ell(u)) - s(v, u)| \\ &= \frac{1}{2} \sum_{v \in V} C_{v,p}(\ell(v) \mid \ell), \end{aligned}$$

where

$$C_{v,p}(\ell(v) \mid \ell) = \sum_{u \in V \setminus \{v\}} |H(\ell(v), \ell(u)) - s(v, u)| \qquad (3)$$

expresses the error incurred by vertex v when it has the labels $\ell(v)$, and the remaining nodes are labeled according to ℓ. The subscript p in $C_{v,p}$ serves to remind us that the set $\ell(v)$ should have at most p labels. Our general local-search strategy is summarized in Algorithm 1.

Algorithm 1 `LocalSearch`

```
1: initialize $\ell$ to a valid labeling;
2: while $C_{\mathrm{occ}}(V, \ell)$ decreases do
3:     for each $v \in V$ do
4:         find the label set $L$ that minimizes $C_{v,p}(L \mid \ell)$;
5:         update $\ell$ so that $\ell(v) = L$;
6: return $\ell$
```

Line 4 is the step in which `LocalSearch` seeks to find an optimal set of labels for an object v by solving Equation (3). This is also the place that our framework differentiates between the measures of Jaccard coefficient and set-intersection.

B. Local step for Jaccard coefficient

*Problem 3 (*JACCARD-TRIANGULATION*):* Consider the set $\{\langle S_j, z_j \rangle\}_{j=1\ldots n}$, where S_j are subsets of a ground set $U = \{1, \ldots, k\}$, and z_j are fractional numbers in the interval $[0, 1]$. The task is to find a set $X \subseteq U$ that minimizes the distance

$$d(X, \{\langle S_j, z_j \rangle\}_{j=1\ldots n}) = \sum_{j=1}^{n} |J(X, S_j) - z_j|. \qquad (4)$$

The intuition behind Equation (4) is that we are given sets S_j and "target similarities" z_j and we want to find a set whose Jaccard coefficient with each set S_j is as close as possible to the target similarity z_j. A moment's thought can convince us that Equation (4) corresponds exactly to the error term $C_{v,p}(\ell(v) \mid \ell)$ defined by Equation (3), and thus, in the local-improvement step of the `LocalSearch` algorithm.

To our knowledge, JACCARD-TRIANGULATION is a new and interesting problem, which has not been studied before, in particular in the context of overlapping clustering. The most-related problem that we are aware of is the problem of finding the Jaccard median, which was recently studied by Chierichetti et al. [5]. The Jaccard-median problem is a special case of the JACCARD-TRIANGULATION problem, where all similarities z_j are equal to 1. Chierichetti et al. provide a PTAS for the Jaccard-median problem. However, their techniques seem mostly of theoretical interest, and do not extend beyond the special case where all $z_j = 1$.

However, since JACCARD-TRIANGULATION is a generalization of the Jaccard-median problem that has been proven $\mathbf{NP}$-hard [5], the following is immediate.

Theorem 2: JACCARD-TRIANGULATION is $\mathbf{NP}$-hard.
We next proceed to discuss our proposed algorithm for the JACCARD-TRIANGULATION problem. The idea is to introduce a variable x_i for every element $i \in U$. The variable x_i indicates if element i belongs in the solution set X. In particular, $x_i = 1$ if $i \in X$ and $x_i = 0$ otherwise. We then assume that the size of set X is t, that is,

$$\sum_{i \in U} x_i - t = 0. \tag{5}$$

Now, given a set S_j with target similarity z_j we want to obtain $J(X, S_j) = z_j$, for all $j = 1, \ldots n$, or

$$J(X, S_j) = \frac{\sum_{i \in S_j} x_i}{|S_j| + t - \sum_{i \in S_j} x_i} = z_j,$$

which is equivalent to

$$z_j t - (1 + z_j) \sum_{i \in S_j} x_i = z_j |S_j|, \tag{6}$$

and we have one Equation of type (6) for each pair $\langle S_j, z_j \rangle$. We observe that Equations (5) and (6) are linear with respect to the unknowns x_i and t. On the other hands, the variables x_i and t take integral values, which implies that the system of Equations (5) and (6) cannot be solved efficiently. Instead we propose to relax the integrality constraints to non-negativity constraints $x_i, t \geq 0$ and solve the above system in the least-squares sense. Thus, we apply a non-negative least-squares optimization method (NNLS) and we obtain estimates for the variables x_i and t.

The solution we obtain from the NNLS solver has two drawbacks: (*i*) it does not incorporate the constraint of having at most p labels, and (*ii*) most importantly, it does not have a clear interpretation as a set X, since the variables x_i may take any non-negative value, not only 0–1. We address both of these problems with a greedy post-processing of the fractional solution: We first sort the variables x_i in decreasing order, breaking ties arbitrarily. We then obtain a set X_q by setting the first q variables x_i to 1 and the rest to 0. We vary q from 1 to p. Out of the p different sets X_q that we obtain this way, we select the one that minimizes the cost $d(X_q, \{\langle S_j, z_j \rangle\})$, and return this as the solution to the JACCARD-TRIANGULATION problem.

An alternative approach could be to optimize the sum of squares of differences of Equation (6), for all j, subject to the constraint of Equation (5), the constraint $\sum_{i \in U} x_i \leq p$, and the constraints $0 \leq x_i \leq 1$. This formulation leads to a quadratic program, which can be also solved by standard solvers, albeit in way less efficient than non-negative least squares. Since this computation is performed for each object in the inner loop of Algorithm 1, in this paper, for efficiency reason we adopt the non-negative least squares formulation.

C. Local step for set-intersection function

Following the approach of the previous section, we formulate the problem that we need to solve for the local-improvement step of the `LocalSearch` algorithm (line 4 of Algorithm 1) in the case of the set-intersection function I.

*Problem 4 (*HIT-N-MISS*):* Let $\mathcal{C}$ be a collection of n tuples of the from $\langle S_j, h_j, m_j \rangle$, with $j = 1 \ldots n$, where S_j are subsets of a ground set $U = \{1, \ldots, k\}$, while h_j and m_j are non-negative numbers. A set $X \subseteq U$ partition $\mathcal{C}$ in $\mathcal{C}_X = \{S_j \mid I(X, S_j) = 1\}$ and $\mathcal{C}_{\bar{X}} = \{S_j \mid I(X, S_j) = 0\}$. The task is a find a set X in order to minimize the distance

$$d(X, \{\langle S_j, h_j, m_j \rangle\}) = \sum_{j \mid S_j \in \mathcal{C}_X} h_j + \sum_{j \mid S_j \in \mathcal{C}_{\bar{X}}} m_j. \tag{7}$$

Once again, we should be able to verify that the Equation (7) corresponds to the cost $C_{v,p}(\ell(v) \mid \ell)$ defined by Equation (3) in the case that the cluster-label similarity function H is the set-intersection function I. In fact, for the problems defined by Equation (3) we always have $h_j + m_j = 1$. However, since we do not know how to take advantage of the additional structure $h_j + m_j = 1$ we just formulate Problem 4 in its generality.

The HIT-N-MISS problem is related to set-cover type of problems. As in set cover, we are given a collection $\mathcal{C}$ of sets S_j. Each set is accompanied with two penalty scores, a *hit* penalty p_j and a *miss* penalty n_j. Our task is to find a new set X in order to either hit or miss the sets S_j, as dictated by their penalty scores h_j and m_j. In particular, for each set S_j that X hits we have to pay its hit penalty h_j, while for each set S_j that X misses we have to pay its miss penalty m_j. The HIT-N-MISS problem is isomorphic to the *positive-negative partial set-cover* problem, studied by Miettinen [6], who showed that the problem is not approximable within a constant factor, but it admits an $O(\sqrt{n} \log n)$ approximation.

In our setting we solve the HIT-N-MISS problem with a simple greedy strategy: Starting from $X_0 = \emptyset$, let X_t the current solution and let $A = U \setminus X_t$ be the set of currently available items (cluster labels). Then for the next step of the greedy we pick the item i from the set of available items A that yields the lowest distance cost, evaluated as $d(X_t \cup \{i\}, \{\langle S_j, h_j, m_j \rangle\})$. We terminate when there is no further decrease in the cost or when we reach the maximum number of cluster labels allowed, i.e., $t = p$.

D. Initialization

The local search algorithm described above requires an initial labeling of the nodes. This can be done in several ways. In the experiments that follow we always use a random initialization. However, in our future work we plan to investigate an initialization based on solving a standard graph coloring problem on the graph $\widehat{G}$ of Proposition 2.

V. Experimental evaluation

We next report our experimentation aimed at assessing under what conditions our algorithms, dubbed OCC-JACC (Jaccard) and OCC-ISECT (set-intersection indicator), can reconstruct a ground truth clustering.

We consider two publicly available datasets (EMOTION and YEAST), originally used in the context of multi-label classifiers. In these data each example is associated with multiple labels (as opposed to only one class)[1]. Such a labeling can be interpreted as a *ground truth* overlapping clustering g, where each label induces a cluster. In this experiment the input to our algorithms are the Jaccard coefficients between the labels of every object pair. For OCC-ISECT we convert the weights to positive and negative edges by labeling the edge as positive unless the Jaccard coefficient is equal to zero.

Performance is evaluated by comparing the labeling ℓ produced by the algorithms, with the ground truth clustering g, using precision and recall, defined as in [7]:

$$\text{prec}_g(\ell) = \frac{|P(\ell) \cap P(g)|}{|P(\ell)|} \qquad \text{rec}_g(\ell) = \frac{|P(\ell) \cap P(g)|}{|P(g)|},$$

where $P(x) = \{(u, v) : x(u) \cap x(v) \neq \emptyset\}$ is the set of pairs of objects with at least one common label in labeling x. We also report the average cost per edge, i.e., the cost of the solution $C_{\text{occ}}(V, \ell)$ as in Eq. (2), divided by the number of edges in the input.

We considered both the case where the algorithm is given only the total number of clusters k, as well as a variant where also the vertex-specific bound on the number of labels, denoted p, is given. In the former case p was not specified, while in the latter case k was fixed to the true number of distinct labels.

Medians of the metrics over 30 trials together with 90% confidence intervals are shown in Figure 1 and Figure 2 for the OCC-JACC and OCC-ISECT algorithms, respectively. With the EMOTION data OCC-JACC performs better than OCC-ISECT, with YEAST the situation is reversed. Observe that in the case where p is varied, precision is high already for low values, as p increases also recall increases, which makes sense as the number of pairs that belong to the same cluster tends to increase when overlaps are allowed.

In the basic form the input to our algorithms contains all $|V| \times |V|$ pairwise similarities. However, it turns out that there can be a lot of redundancy in this input. Often we can prune most of the pairwise comparisons with negligible loss in quality. This is an important characteristic, as it allows us to apply the algorithm also for larger data sets. Selecting the best set of edges to prune is an interesting problem in its own right. In this experiment we took the simple approach, and prune edges at random: an edge is taken in consideration with probability q (denoted the pruning threshold) independently of the other edges. In Figure 3 we show edge-specific cost as well as precision and recall as a function of q for the OCC-JACC algorithm (the curves are again medians over 30 trials). Clearly with these example data sets the pruning threshold can be set very low. Also, there is a noticeable "threshold effect" in the cost/edge that may serve as an indicator to find the pruning threshold in a setting where a ground truth is not available. This suggests that in practice it is not necessary to use all pairwise comparisons, a sample of the graph may be enough. In fact, the results for YEAST shown in figures 1 and 2 were computed with $q = 0.05$. In terms of computational speedup pruning has a very positive effect. Using the full YEAST data (without pruning) our Python implementation takes 400 seconds to finish on a 1.8GHz CPU, with pruning ($q = 0.05$) this can be reduced to 70.

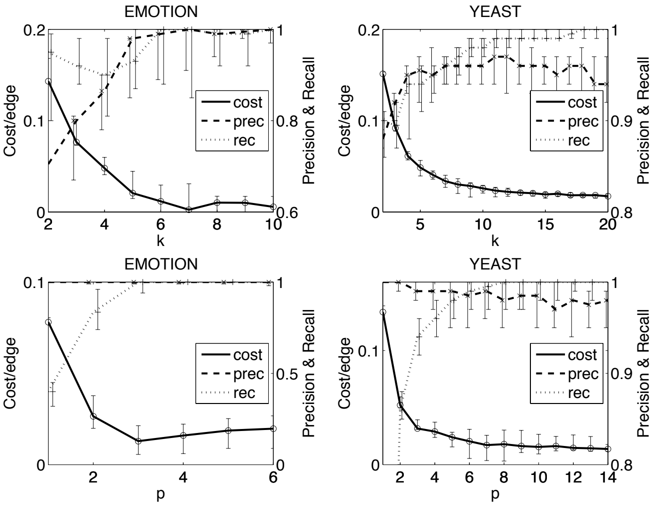

Fig. 1. Cost per edge, precision and recall of OCC-JACC as a function of k, the total number of distinct labels (top row), and as a function of p, the maximum number of labels per vertex (bottom row).

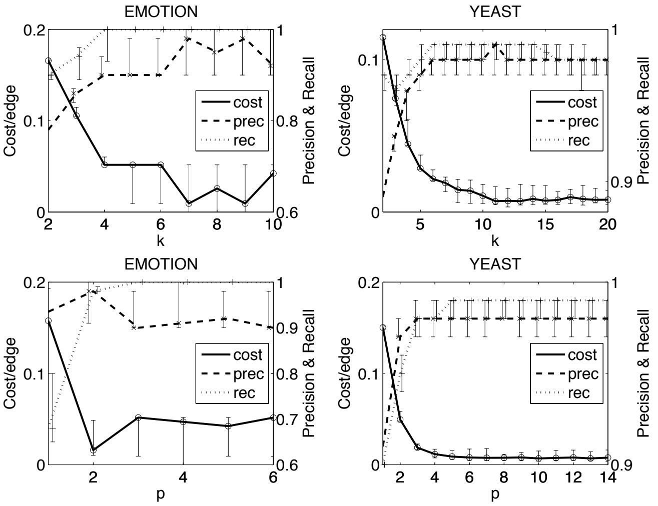

Fig. 2. Same as in Figure 1, but using OCC-ISECT.

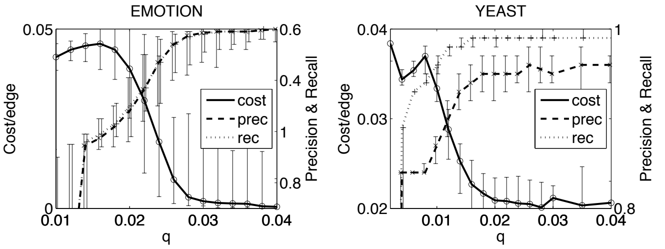

Fig. 3. Pruning experiment using OCC-JACC. Cost/edge and precision and recall as a function of the pruning threshold q.

[1] EMOTION has 593 objects and 6 labels. YEAST has 2417 objects and 14 labels. More information at: http://mulan.sourceforge.net/datasets.html

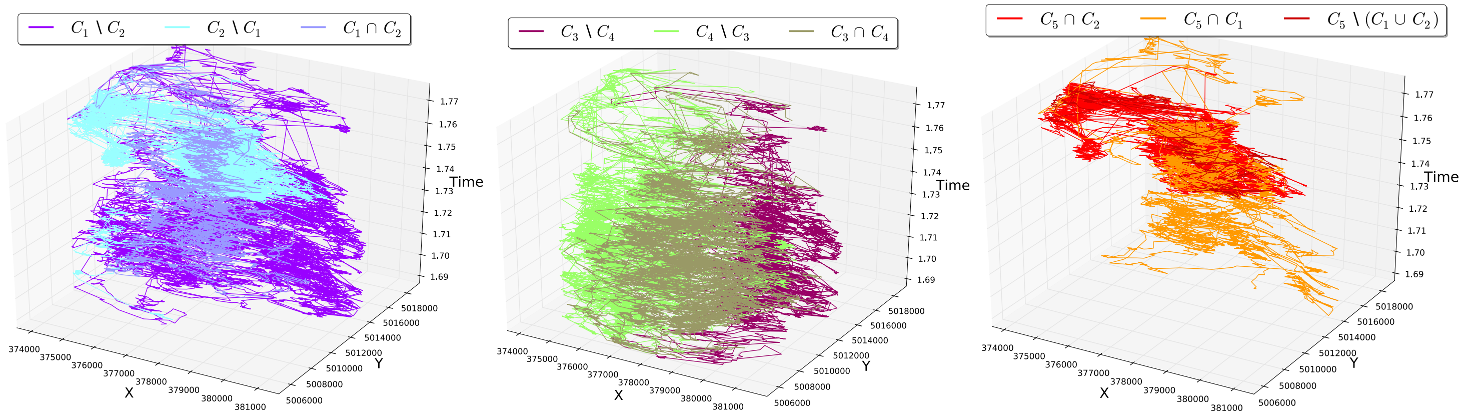

Fig. 4. Plots in space and time of all the trajectories in the five clusters obtained in the STARKEY'93 dataset. Time axis is $\times 10^8$ and it counts elapsed seconds since 12/31/87, which represents a starting point of the Starkey Ungulate Research project.

VI. Applications

Since our method uses pairwise similarities of objects as the input, it lends itself naturally to the task of clustering structured objects for which feature vectors can be difficult to obtain. In this section we discuss clusterings for two of such data: movement trajectories, and protein sequences.

A. Overlapping Clustering of Trajectories

Spatio-temporal data mining is an active research area with a huge variety of application domains, e.g., mobility management, video surveillance, mobile social networks, atmosphere analysis, biology and zoology just to mention a few. The basic entities in analysis are usually trajectories of moving objects, i.e., sequences $\langle (x_0, y_0, t_0), \ldots, (x_n, y_n, t_n) \rangle$ of observations in space and time. Trajectories are structured complex objects, and mining them is often an involved task. They might have different lengths (i.e., different number of observations), and therefore viewing them simply as vectors and using standard distance functions (e.g., Euclidean) is not feasible. Moreover, the different nature of space and time implies different granularity and resolution issues. While there has been quite some research on clustering trajectories [8], [9], to the best of our knowledge the problem of overlapping clustering of trajectories has not been studied.

To motivate the application of overlapping clustering for trajectories, consider, for instance, a well-shaped cluster C_1 of trajectories of GPS-equipped cars going from a south-west suburb to the city center between 8 and 9am, and another cluster C_2 moving inside the city center along a specific path, from 3 to 4pm. Now consider a trajectory that moves from the south-west suburb to the city center in the morning, and within the city center in the afternoon: it is quite natural to assign this trajectory to both clusters C_1 and C_2.

To apply our framework to trajectories we need to compute a distance among trajectory pairs. We choose the EDR distance [10], which is *time-tolerant*, that is, it is defined even among trajectories of different length.[2]

TABLE I

Result of clustering on STARKEY'93 *dataset. On the diagonal, we report the population of each cluster, using C,D, and E to distinguish between number of cattle, deer and elks respectively. In the non-diagonal cells of the matrix, we report the population of the overlap among two clusters.*

C_1	C_2	C_3	C_4	C_5	clusters
16E 6D	3E 2D	3E	5E	3E 2D	C_1
	4E 2D 38C	∅	1E	30C	C_2
		13E 6D	9E	∅	C_3
			21E 2D	∅	C_4
				3E 2D 33C	C_5

We use animal movement data generated by the Starkey project.[3] The dataset contains the radio-telemetry locations of elks, deer, and cattle from 1993 to 1996. We select to work on all the three species together and only for 1993. The dataset contains only 88 trajectories (corresponding to 33 elks, 14 deer, and 41 cattle), but each trajectory is very long. The whole dataset has 79,987 (x, y, t) observations, an average of 909 observations per trajectory.[4]

Studying overlapping trajectory clusters in this context might be of zoological interest, as discussed by Coe et al. [11].

> *These three species have important social, ecological, and economic values. Understanding their interspecific interactions may clarify two recurring issues in their management: competition for food and competition for space, both of which may result in decreased animal fitness. [...] Accurate predictions of ungulate distributions over time and space may help managers regulate densities and understand effects of specific ungulates on ecosystem processes. [...] Overlapping distributions could be evidence for competition or dependence. Non-overlap could be an expression of active avoidance or ecological separation, which occurs when two species evolved together.*

[2] We normalize the EDR distance to stay in the range $[0, 1]$ making it suitable for our method after conversion to similarity: $\mathrm{sim}(u, v) = 1 - \mathrm{edr}(u, v)$.

[3] http://www.fs.fed.us/pnw/starkey/

[4] We computed EDR using the following space and time tolerance parameters: $\Delta.x = \Delta.y = 2.5k, \Delta.t = 500k$.

We report the results of a clustering obtained with our framework with $k = 5$ and $p = 2$. Interestingly in this context both definitions of similarity between sets of elements—Jaccard and set-intersection—yield very consistent clusterings, with just few elements assigned to a different cluster. In Table I we report the results obtained with the Jaccard similarity measure. We can observe that two clusters C_2 and C_5 contain mostly cattle, and few individuals of the other species, while the other 3 clusters do not contain any cattle. In particular cluster C_4 contains mainly elks. This is in line with the zoological domain knowledge. Cattle are introduced in late spring or early summer for the grazing season. During summer elks and deer avoid cattle, but in late summer and fall the three species overlap in some areas, due to exploitive competition for forage resources that have become depleted [11].

It is interesting to observe that cluster C_5, containing 33 cattle, 3 elks and 2 deer, is almost perfectly covered by C_1 and C_2, with the 3 elks and 2 deer falling in the former, and the cattle in the latter. In Figure 4 the trajectories in each cluster are plotted in space and time. We can see that clusters C_3 and C_4, containing only elks and deer, cover most of the area available with C_3 moving in higher X than C_4, while clusters C_1, C_2, and C_5 contain individuals that almost never enter in the area $Y < 5014k$. Cluster C_2 contains almost all cattle that enter the area only in late spring ($\text{Time} > 1.724$), plus few elks and deer (belonging to cluster C_1, too) that move closer to the cattle than what the others in clusters C_3 and C_4.

B. Overlapping Clustering of Protein Sequences

An important problem in genomics is the study of evolutionary relatedness of proteins. We use our algorithms to cluster proteins to homologous groups given pairwise similarities of their amino-acid sequences. Such similarities are computed by the sequence alignment tool BLAST [12]. We follow the approach of Paccanaro et al. [13] and Nepusz et al. [14], and compare the computed clustering against a ground truth given by SCOP, a manually crafted taxonomy of proteins [15]. The SCOP taxonomy is a tree with proteins at the leaf nodes. The ground truth clusters used in the experiments are subsets of the leafs, that is, proteins, rooted at different SCOP *superfamilies*. These are nodes on the 3rd level below the root.

We compare our algorithm with the SCPS algorithm [13], [14], a spectral method for clustering biological sequence data. The experiment is run using datasets 1–4 from Nepusz et al.,[5] which contain pre-computed sequence similarities together with the ground-truth clusterings. Dataset $D1$ contains 669 sequences and 5 ground-truth clusters, dataset $D2$ contains 587 sequences and 6 clusters, dataset $D3$ contains 567 sequences and 5 clusters, and dataset $D4$ contains 654 sequences and 8 ground-truth clusters.

To compare with the SCPS algorithm we first computed non-overlapping clusterings of all four datasets. All algorithms were given the correct number of clusters as a parameter.

TABLE II

Precision, recall, and their harmonic mean F-score, for non-overlapping clusterings of protein sequence datasets computed using SCPS [14] and the OCC algorithms. BL is the precision of a baseline that assigns all sequences to the same cluster.

	BL	SCPS	OCC-ISECT	OCC-JACC
dataset	prec	prec/recall/F-score	prec/recall/F-score	prec/recall/F-score
D1	0.21	0.56 / 0.82 / 0.664	0.70 / 0.67 / **0.683**	0.57 / 0.55 / 0.561
D2	0.17	0.59 / 0.89 / 0.708	0.86 / 0.83 / **0.844**	0.64 / 0.63 / 0.637
D3	0.38	0.93 / 0.88 / **0.904**	0.81 / 0.43 / 0.558	0.73 / 0.39 / 0.505
D4	0.14	0.30 / 0.64 / 0.408	0.64 / 0.56 / **0.598**	0.44 / 0.39 / 0.412

Results are shown in Table II. SCPS has a higher recall in every case, but with datasets 1, 2, and 4, the OCC-ISECT algorithm achieves a substantially higher precision. In practice this means that if OCC-ISECT assigns two sequences to the same cluster, they belong to the same cluster also in the ground truth with higher probability than when using SCPS.[6]

We also conduct a more fine-grained analysis of the results using the SCOP taxonomy. Intuitively the cost of a clustering errors should take distances induced by the taxonomy into account. If two proteins are placed in the same cluster, they should contribute more (less) to the clustering cost if their distance in the taxonomy is higher (lower). Consequently, we define the SCOP similarity between two proteins as follows:

$$\mathrm{sim}(u, v) = \frac{\mathrm{d}(\mathrm{lca}(u, v))}{\max(\mathrm{d}(u), \mathrm{d}(v)) - 1}, \tag{8}$$

where $\mathrm{d}(u)$ is the depth of a node in the tree (the root is at depth 0), and $\mathrm{lca}(u, v)$ denotes the lowest common ancestor of the nodes u and v. We then define the cost of a clustering to be $1 - \mathrm{sim}(u, v)$ for two proteins that are assigned to the same cluster, and $\mathrm{sim}(u, v)$ for two proteins assigned to different clusters.

TABLE III

Comparing clusterings cost based on distance on the SCOP taxonomy, for different values of p, the maximum number of labels per protein.

	SCPS	OCC-ISECT-p1	OCC-ISECT-p2	OCC-ISECT-p3
D1	0.231	0.196	0.194	0.193
D2	0.188	0.112	0.107	0.106
D3	0.215	0.214	0.214	0.231
D4	0.289	0.139	0.133	0.139
	SCPS	OCC-JACC-p1	OCC-JACC-p2	OCC-JACC-p3
D1	0.231	0.208	0.202	0.205
D2	0.188	0.137	0.130	0.127
D3	0.215	0.243	0.242	0.221
D4	0.289	0.158	0.141	0.152

The results of Table III suggest that the OVERLAPPING-CORRELATION-CLUSTERING algorithms, find a clustering that is better in agreement with the SCOP taxonomy than are the clusterings found by SCPS. However, while allowing overlaps is beneficial, we do not observe a significant improvement as the node-specific constraint p is increased. Moreover, we observe that only a small number of proteins are assigned to multiple clusters. We conjecture that this is due to the

[5] The dataset and the SCPS application are available at http://www.paccanarolab.org/software/scps/.

[6] Note that these numbers are not directly comparable with the ones in [14] as they define precision and recall in a slightly different way.

similarities produced by BLAST, which imply very well defined clusters in most of the cases. Nevertheless, it is worth noting that our methods, regardless of the parameters used, do not find unnecessarily large overlaps, when this is not dictated by the data.

VII. Related work

Correlation Clustering. The problem of CORRELATION-CLUSTERING was first defined by Bansal et al. [1]. In their definition, the input is a complete graph with positive and negative edges. The objective is to partition the nodes of the graph so as to minimize the number of positive edges that are cut and the number of negative edges that are not cut; corresponding to our problems $(\mathrm{b}, H, 1)$. This is an APX-hard optimization problem which has received a great deal of attention in the field of theoretical computer science [16], [17], [18], [19].

Ailon et al. [16] considered a variety of correlation clustering problems. They proposed an algorithm that achieves expected approximation ratio 5 if the weights obey the probability condition. If the weights X_{ij} obey also the triangle inequality, then the algorithm achieves expected approximation ratio 2. Swamy [19] has applied semi-definite programming to obtain a 0.76-approximation algorithm for the corresponding maximization problem: maximize agreements, rather than minimize disagreements. Giotis and Guruswami [18] consider correlation clustering when the number of clusters is given, while Ailon and Liberty [17] study a variant of correlation clustering where the goal is to minimize the number of disagreements between the produced clustering and a given ground truth clustering.

To the best of our knowledge, no previous work has consider the possibility of overlaps in correlation clustering, i.e., the problem (r, H, p), with $p > 1$.

Overlapping clustering. In 1979 Shepard and Arabie introduced the ADCLUS algorithm [20] for additive clustering, which perhaps can be considered the first overlapping-clustering method. The method, which has been later applied in the marketing domain [21], subsumes hierarchical clustering as a special case and can be regarded as a discrete analog of principal components analysis.

Regardless this ancient roots, in the last decades overlapping clustering has not attracted as much attention as non-overlapping clustering. One close sibling is fuzzy clustering [22], where each data point has a membership value in all the clusters. In this context cluster membership is "soft", as apposed to our paper that we are interested in "hard" cluster assignments. Obviously a hard (and overlapping) cluster assignment can be obtained by thresholding membership values. The prototypical fuzzy-clustering method is *fuzzy c-means*, which is essentially a soft version of k-means.

Recently mixture-models have been generalized to allow overlapping clusters. Banerjee et al. [7] generalize the work of Segal et al. [23] to work with any regular exponential family distribution, and corresponding Bregman divergence. The work of Banerjee et al. [7] has later been extended to co-clustering by Shafiei and Milios [24]. Multiplicative mixture models have been proposed as a framework for overlapping clustering by Fu and Banerjee [25].

Our work distinguishes from this body of research as it develops within the correlation clustering framework, and thus it has a different input and different objectives. One of the main differences is that the above discussed methods are not easily applicable when features vectors are not available, as in our application on trajectories and proteins.

Multiple clustering solutions. A large body of work studies the problem of discovering multiple clustering solutions [26], [27], [28], [29], [30]. The objective in these papers is to discover multiple clusterings for a given dataset. Each of the clusterings needs to be of high quality and the clusterings are required to be different with each other in order to cover different aspects of the dataset. Each of the discovered clusterings is non-overlapping, so this clustering paradigm is not directly comparable with our clustering result.

Constrained clustering. The binary version of correlation clustering $(\mathrm{b}, H, 1)$ with positive and negative links, can be seen as a "soft" instance of clustering with *Must-Link* (ML) and *Cannot-Link* (CL) constraints. For the latter problem there exists an extensive literature [31], [32], [33], [34], [35]. However, constraint clustering and correlation clustering are qualitatively different problems. In constrained clustering there distances and additional ML and CL constraints, while in correlation clustering distances and constraints coincide.

Although presented in the context of ML and CL constraints, the work by Scripps and Tan [36] essentially deals with a binary version of correlation clustering, and it adopts "cloning" to fix the problem of bridge nodes (or "bad triplets" in the jargon of Ailon et al. [16]). Cloning essentially means allowing overlaps. In the notation we introduce in this paper, their problem is exactly (b, I, k) with no predefined number of clusters. As we prove in Corollary 1, such a problem always admits a straightforward zero-cost solution. Scripps and Tan [36] are interested only in zero-cost solutions, while trying to minimize the number of clones (i.e., overlaps). Instead we consider the problem of finding minimum cost solutions with a prefixed number of clusters, or with constraint on the maximum number of clusters per object.

Applications. Developments in overlapping clustering has mainly been driven by the concrete needs of applications. For instance, driven by the need to cluster microarray gene expression data, various methods for overlapping clustering [37], [23] and overlapping bi-clustering [38], [39] have been proposed.

Even though detecting communities in social networks is a problem that has been studied extensively, only few researchers have addressed the problem of detecting overlapping communities; for a survey, see Fortunato [40, Section 11]. The best known approach to detect overlapping communities is the CFinder algorithm based on *clique percolation* [41]. According to the CFinder method, communities are discovered by finding k-cliques and merging them when they share $k-1$

nodes. As a node can belong to multiple cliques, the method generates overlapping communities.

Tang and Liu [42] cluster edges instead of nodes, which results in overlapping communities of nodes. Following up the latter work, Wang et al. [43] study the problem of discovering overlapping groups in social media, and they propose a co-clustering framework based on users and tags.

VIII. FUTURE WORK

We present a novel formulation for overlapping clustering. In a nutshell, to each data point is assigned a set of labels representing membership to different (overlapping) clusters. Defining a similarity function $H(\ell(u), \ell(v))$ between cluster labels, allow us to define our objective function over the "residual error" $|H(\ell(u), \ell(v)) - s(u, v)|$, where $s(u, v)$ is the input similarity function between pairs of data points. In this paper we consider summing the error terms $|H(\ell(u), \ell(v)) - s(u, v)|$ over all pairs of data points. An interesting future direction is to apply our idea to other clustering paradigms, different than correlation clustering. For example, one can consider only the error terms among data points and k "prototypical" data points.

With respect to the concrete optimization problems defined in this paper, it would be interesting to investigate different approaches, for example, using non local-search algorithms such as the idea based on relaxed graph-coloring, mentioned in Section III. Other interesting directions are to apply graph coloring solutions for the initialization step, as discussed in Section IV-D, and to design an approximation algorithm for the JACCARD-TRIANGULATION problem. Finally, it will be very interesting to apply the approach to different application domains.

Acknowledgements. This work was partially supported by the Spanish Centre for the Development of Industrial Technology under the CENIT program, project CEN-20101037, "Social Media" (www.cenitsocialmedia.es).

REFERENCES

[1] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine Learning*, vol. 56, no. 1–3, 2004.

[2] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1979.

[3] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher, "Min-wise independent permutations," in *STOC*, 1998.

[4] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *STOC*, 1998.

[5] F. Chierichetti, R. Kumar, S. Pandey, and S. Vassilvitskii, "Finding the jaccard median," in *SODA*, 2010.

[6] P. Miettinen, "On the positive-negative partial set cover problem," *Information Processing Letters*, vol. 108, no. 4, 2008.

[7] A. Banerjee, C. Krumpelman, J. Ghosh, S. Basu, and R. J. Mooney, "Model-based overlapping clustering," in *KDD*, 2005.

[8] S. Gaffney and P. Smyth, "Trajectory clustering with mixtures of regression models," in *KDD*, 2009.

[9] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in *SIGMOD*, 2007.

[10] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories." in *SIGMOD*, 2005.

[11] P. K. Coe, B. K. Johnson, K. M. Stewart, and J. G. Kie, "Spatial and temporal interactions of elk, mule deer, and cattle," *Transactions of the 69th North American Wildlife and Natural Resources Conference*, 2004.

[12] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403–410, 1990.

[13] A. Paccanaro, J. A. Casbon, and M. A. S. Saqi, "Spectral clustering of protein sequences," *Nucleic Acids Research*, vol. 34, no. 5, 2006.

[14] T. Nepusz, R. Sasidharan, and A. Paccanaro, "Scps: a fast implementation of a spectral method for detecting protein families on a genome-wide scale," *BMC Bioinformatics*, vol. 11, no. 1, p. 120, 2010.

[15] A. Murzin, S. Brenner, T. Hubbard, and C. Chothia, "Scop - a structural classification of proteins database for the investigation of sequences and structures," *Journal of Molecular Biology*, vol. 247, no. 4, 1995.

[16] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: ranking and clustering," in *STOC*, 2005.

[17] N. Ailon and E. Liberty, "Correlation clustering revisited: The "true" cost of error minimization problems," in *ICALP*, 2009.

[18] I. Giotis and V. Guruswami, "Correlation clustering with a fixed number of clusters," in *SODA*, 2006.

[19] C. Swamy, "Correlation clustering: maximizing agreements via semidefinite programming," in *SODA*, 2004.

[20] R. N. Shepard and P. Arabie, "Additive clustering: Representation of similarities as combinations of discrete overlapping properties," *Psychological Review*, vol. 86, no. 2, 1979.

[21] P. Arabie, J. D. Carroll, W. DeSarbo, and J. Wind, "Overlapping clustering: A new method for product positioning," *Journal of Marketing Research*, vol. 18, no. 3, 1981.

[22] J. C. Bezdek and S. K. Pal, Eds., *Fuzzy Models for Pattern Recognition – Methods That Search for Structures in Data*. IEEE Press, 1992.

[23] E. Segal, A. Battle, and D. Koller, "Decomposing gene expression into cellular processes," in *PSB*, 2003.

[24] M. M. Shafiei and E. E. Milios, "Model-based overlapping co-clustering," in *Proc. of the Fourth Workshop on Text Mining*, 2006.

[25] Q. Fu and A. Banerjee, "Multiplicative mixture models for overlapping clustering," in *ICDM*, 2008.

[26] E. Bae, J. Bailey, and G. Dong, "A clustering comparison measure using density profiles and its application to the discovery of alternate clusterings," *Data Mining and Knowledge Discovery*, vol. 21, 2010.

[27] R. Caruana, M. F. Elhawary, N. Nguyen, and C. Smith, "Meta clustering," in *ICDM*, 2006.

[28] M. S. Hossain, S. Tadepalli, L. T. Watson, I. Davidson, R. F. Helm, and N. Ramakrishnan, "Unifying dependent clustering and disparate clustering for non-homogeneous data," in *KDD*, 2010.

[29] Z. Qi and I. Davidson, "A principled and flexible framework for finding alternative clusterings," in *KDD*, 2009.

[30] E. Müller, S. Günnemann, I. Färber, and T. Seidl, "Discovering multiple clustering solutions: Grouping objects in different views of the data," in *ICDM*, 2010.

[31] S. Basu, A. Banerjee, and R. J. Mooney, "Active semi-supervision for pairwise constrained clustering," in *SDM*, 2004.

[32] S. Basu, M. Bilenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *KDD*, 2004.

[33] D. Klein, S. D. Kamvar, and C. D. Manning, "From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering," in *ICML*, 2002.

[34] K. Wagstaff and C. Cardie, "Clustering with instance-level constraints," in *ICML*, 2000.

[35] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained k-means clustering with background knowledge," in *ICML*, 2001.

[36] J. Scripps and P.-N. Tan, "Clustering in the presence of bridge-nodes," in *SDM*, 2006.

[37] A. Battle, E. Segal, and D. Koller, "Probabilistic discovery of overlapping cellular processes and their regulation," in *RECOMB*, 2004.

[38] Y. Cheng and G. M. Church, "Biclustering of expression data," in *ISMB*, 2000.

[39] Q. Fu and A. Banerjee, "Bayesian overlapping subspace clustering," in *ICDM*, 2009.

[40] S. Fortunato, "Community detection in graphs," *Phys. Rep. 486*, 2010.

[41] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, 2005.

[42] L. Tang and H. Liu, "Scalable learning of collective behavior based on sparse social dimensions," in *CIKM*, 2009.

[43] X. Wang, L. Tang, H. Gao, and H. Liu, "Discovering overlapping groups in social media," in *ICDM*, 2010.