Motif and Hypergraph Correlation Clustering

Pan Li, Gregory J. Puleo, and Olgica Milenkovic, *Fellow, IEEE*

***Abstract*— Motivated by applications in social and biological network analysis we introduce a new form of agnostic clustering termed *motif correlation clustering*, which aims to minimize the cost of clustering errors associated with both edges and higher-order network structures. The problem may be succinctly described as follows: Given a complete graph G, partition the vertices of the graph so that certain predetermined "important" subgraphs mostly lie within the same cluster, while "less relevant" subgraphs are allowed to lie across clusters. Our contributions are as follows: We first introduce several variants of motif correlation clustering and then show that these clustering problems are NP-hard. We then proceed to describe polynomial-time clustering algorithms that provide constant approximation guarantees for the problems at hand. Despite following the frequently used LP relaxation and rounding procedure, the algorithms involve a sophisticated and carefully designed neighborhood growing step that combines information about both edges and motifs. We conclude with several examples illustrating the performance of the developed algorithms on synthetic and real networks.**

***Index Terms*— Correlation clustering, network motif, hypergraph, graph clustering.**

I. Introduction

CORRELATION clustering is an agnostic clustering method introduced by Chawla *et al.* [2]. The clustering objective has a simple description: For a collection of objects and, for some pairs of objects in this collection, one is given a quantitative assessment of whether the objects are *similar* or *dissimilar*. This information is represented using a labeled graph with edges marked by $+$ or $-$ symbols according to whether the endpoints are similar or dissimilar. The task is to partition the vertices of the graphs so that edges labeled by $+$ aggregate within clusters and edges labeled by $-$ go across clusters. Unlike many other well-known clustering methods, correlation clustering does not require the number of clusters to be specified in advance.

The correlation clustering optimization problem comes in two basic forms: *MinDisagree* and *MaxAgree*. The MinDisagree version, as its name suggests, aims to minimize the number of erroneously placed edges, while the MaxAgree version aims to maximize the number of correctly placed edges. Finding an optimal solution to either version gives the solution to the other version. Even when the graphs are complete, both problems are known to be NP-hard. Hence, the focus of all prior work in the field has been on approximating the optimal solutions [2]. The approximation hardness is fundamentally different for the two versions of the problem. For the MinDisagree problem over complete graphs, several randomized [3] and deterministic [4] algorithms with constant approximation guarantees are known and the problem is APX-hard [4]. In contrast, for the MaxAgree problem over complete graphs, a trivial method that outputs either singletons or a giant cluster yields a 2-approximate solution and the problem has been shown to have a PTAS [2].

Beyond binary labeled edged, several variants of correlation clustering also allow for edges to be endowed with pairs of continuously valued similarity and dissimilarity weights. Specifically, if an edge is placed between two clusters, the similarity weight is charged, and if an edge is placed within one cluster, the dissimilarity weight is charged instead. The goal of the MinDisagree formulation in this setting is to minimize the charge over all possible vertex set partitions. Clearly, an arbitrary choice of edge weights may lead to poor approximability results. To avoid this issue, one usually resorts to so-called *probability weights* [2].

We depart from classical correlation clustering problems by considering a new setting in which one is allowed to assign probability weights to both edges and arbitrary small induced subgraphs in the graph (e.g., triangles) and then perform the clustering so as to minimize the overall cost of both edge and motif placements or motif placements alone. This enables one to extend traditional correlation clustering by considering higher-order structures in the network such as paths, triangles and cycles. Given that subgraphs/motifs may be modeled as hyperedges in a hypergraph, our line of work complements recent works on spectral hypergraph clustering [5]–[9] and heuristic tensor spectral clustering methods [10], as well as other generalizations of correlation clustering [11]–[13]. Furthermore, the proposed method allows for handling motifs in directed graphs by converting the directed graphs into undirected graphs while retaining information about the "relevance" of directed subgraphs within the graph. This relevance information may be incorporated into similarity and dissimilarity weights (For example, if only feedforward triangle motifs are relevant, only those directed motifs will be assigned large weight in the undirected graph and hence encouraged to fall within one cluster).

Motif clustering may be useful for a large number of practical applications. As an example, the authors of [14] used

Manuscript received December 7, 2018; revised June 14, 2019; accepted August 22, 2019. Date of publication September 10, 2019; date of current version April 21, 2020. This work was supported in part by NSF under Grant CIF 1218764, Grant CIF 1117980, and Grant 1339388, and in part by STC Class 2010 under Grant CCF 0939370. G. J. Puleo was supported by the IC Postdoctoral Research Fellowship. This article was presented at the INFOCOM 2017 Conference [1].

P. Li and O. Milenkovic are with the Coordinated Science Laboratory, Department of Electrical and Computer Engineering, University of Illinois at Urbana–Champaign, Champaign, IL 61820 USA (e-mail: panli2@illinois.edu; milenkov@illinois.edu).

G. J. Puleo is with the Department of Mathematics, Auburn University, Auburn, AL 36849 USA (e-mail: gjp0007@auburn.edu).

Communicated by L. Ying, Associate Editor for Communication Networks.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIT.2019.2940246

the fact that triangular motifs carry relevant information in biological networks and clustering (directed) triangles allowed them to identify several communities in the Florida Bay food-web network. Following this line of work, the authors of [6] proposed to leverage the information in big-fan motifs, involving two predatory and two prey species, and to cluster the motifs in a non-uniform manner so as to obtain a hierarchical community structure including levels from producers to high-level consumers. Motif clustering also may be used for discovery of layered flows in a information networks, anomaly detection in communication networks and many other research areas [10]. More broadly, motif correlation clustering enables community recovery for graphs and networks in which higher order structures carry significantly more relevant information about the functionality, direction and strength of connections in the network than edges alone.

Our contributions are as follows. We rigorously formulate the first known MinDisagree motif correlation clustering problem, and show that it is NP-hard even when only some special motifs such as triangles are considered. Next, we introduce an extended correlation clustering framework which allows to both fine tune the cost of clustering edges and higher order network structures. We then describe a new two-stage clustering algorithm comprising an LP and rounding step, and show that the algorithm offers constant approximation guarantees that depend on the size of the motifs under consideration. We also provide examples illustrating that standard randomized pivoting algorithms fail on this instance of correlation clustering. Our exposition concludes with several examples pertaining to synthetic and real network analysis, such as social, flow, and anomaly detection networks, illustrating the advantages of motif correlation clustering over edge-based methods, which also allows us to understand richer structures in graphs.

We remarks that the topic of motif correlation clustering was introduced by the authors in [1]. There, the focus was exclusively on edges and a single type of motif, triangles. Furthermore, the work also introduced "overlapping" correlation clustering for triangle motifs. This work generalizes and extends the results in [1] by addressing clustering with motifs and mixed motif structures of arbitrary size (e.g., Theorem III.2). In the context of mixed motif clustering, this work provides new results that allow for controlling and balancing out the influence of higher-order and low-order motifs. Since the publication of [1], follow up work on motif correlation clustering was also reported in [15]. There, several versions of our single motif clustering methods have been adapted to yield improved approximation constants. Nevertheless, results reported in this work introduce both new sophisticated proof techniques for mixed motif models and improve the approximation guarantees reported in [1] and [15]. As an illustration, for the case when motifs are of size k, we improve the approximation constant in [15] from $4(k-1)$ to $2k$ (Theorem III.1).

II. Notation and Problem Formulation

Let $G(V, E)$ be a complete, undirected graph with vertex set V of cardinality n and edge set E of cardinality $\binom{n}{2}$. For simplicity, we assume that the vertices are endowed with distinct integer labels in $[n]$ and this labeling introduces a natural ordering of the vertices. Also, we let $C = (C_1, \ldots, C_s)$, $1 \leq s \leq n$, stand for a partition of the vertex set $[n]$ and $\mathcal{C}_n$ for the set of all partitions of $[n]$.

Let S be a subset of vertices and let $\mathcal{K}(S)$ denote the set of all k-subsets of vertices in S, where $2 \leq k < n$ is a constant independent of n. Clearly, $|\mathcal{K}(S)| = \binom{|S|}{k}$ and $|\mathcal{K}(V)| = \binom{n}{k}$. Denote a subgraph of G induced by a k-subset of vertices by $K^{(k)} \in \mathcal{K}(V)$ (whenever clear from the context, we omit the superscript k). Each K is associated with a pair of non-negative values (w_K^+, w_K^-). The weights w_K^+ and w_K^- indicate the respective costs of placing the k-tuple K across and within the same cluster, respectively, and they satisfy the probabilistic constraint $w_K^+ + w_K^- = 1$. Note that in practical settings, the most relevant motifs in a graph are edges and triangles. Typically, in practice, the k-tuple K corresponds to a graph motif and its corresponding weights are determined by the functionality of that motif.

For simplicity, we denote variables x associated with K-subsets by x_K; for edges uv, $u, v \in V$ we use $x_{uv} = x_{vu}$.

Our goal is to solve two MinDisagree versions of the problem: In the first version of the problem, termed *motif correlation clustering (MCC)*, we fix one motif graph on k vertices and then seek a vertex partition $C \in \mathcal{C}_n$ that minimizes the following objective function:

$$\textbf{MCC:} \quad \min_{C \in \mathcal{C}_n} \sum_{K^{(k)} \subseteq C_i, \text{ for some } i} w^-_{K^{(k)}} + \sum_{K^{(k)} \not\subseteq C_i, \text{ for all } i} w^+_{K^{(k)}}. \tag{1}$$

In the second version of the problem, termed *mixed motif correlation clustering (MMCC)*, we are allowed to fix multiple motif graphs of possibly different sizes $2 \leq k_1 < k_2 < \ldots < k_p$, and we seek a vertex partition $C = (C_1, \ldots, C_s)$, $s \geq 1$, that minimizes the following objective function:

$$\textbf{MMCC:} \quad \min_{C \in \mathcal{C}_n} \sum_{t=1}^{p} \lambda_t \left(\sum_{K^{(k_t)} \subseteq C_i, \text{ for some } i} w^-_{K^{(k_t)}} + \sum_{K^{(k_t)} \not\subseteq C_i, \text{ for all } i} w^+_{K^{(k_t)}} \right). \tag{2}$$

Here, $\lambda_t \geq 0$ are relevance factors of the motifs of size k_t. Note that by choosing $\lambda = 1$ for edges and setting all other relevance factors to zero, we arrive at the classical correlation clustering formulation. Furthermore, in both problems, we impose the probability constraint on the weights, requiring that $w_K^+ + w_K^- = 1$ for all K.

Clearly, both the MCC and MMCC problems are NP-complete, as the correlation clustering problem is NP-complete. Furthermore, even for restricted choices of motifs (e.g., for $k = 3$), the problems remain hard as it reduces to Partition into Triangles [18]. We formalize this statement in the following theorem and provide a detailed proof in Appendix.

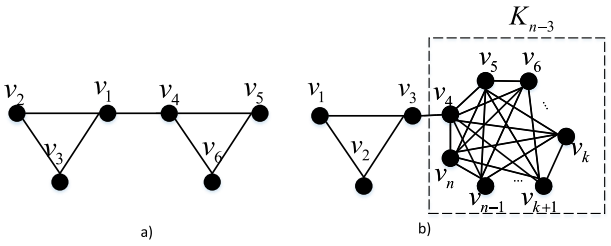

Fig. 1. Pivoting on edges and vertices of graphs. Note that K_{n-3} stands for a complete graph on $n-3$ vertices.

Subsequently, we focus on developing (constant) approximation algorithms for this class of problems.

Theorem II.1. For $k = 3$, the MCC problem is NP-complete.

We point out that one may also consider the MaxAgree version of the motif clustering problem where the objective functions in (MCC) and (MMCC) that summarize disagreement are replaced by objective functions that summarize agreement, and where correspondingly the min function is replaced by the max function. As for the case of correlation clustering, it is straightforward to show that taking the better of two clusterings, the all-singleton clustering and the single-component clustering provide a 2-approximation for the problem.

The MinDisagree version of correlation clustering is usually approximately solved using two approaches: Pivoting methods [3] and relaxed Integer Programming (IP) methods that reduce to solving a Linear Program (LP) followed by rounding [4]. The pivoting algorithm is a straightforward randomized approach that provides constant approximation guarantees for the expected value of the objective, and has straightforward, yet efficient, parallel implementations [16]. For the unweighted clustering problem, pivoting may be succinctly described as follows: One selects a pivot vertex uniformly at random, incorporates all its "similar" neighbors (i.e., those with edge label '+') into one cluster, removes all vertices in the newly formed cluster from the graph and then proceeds to iteratively repeat the same steps. Unfortunately, pivoting applied to motif clustering cannot lead to constant approximation results, as illustrated by the example to follow.

Consider the MCC problem for complete graphs and triple-motifs, i.e., motifs with $k = 3$. Suppose that each edge is labeled, with labels in the set $\{+, -\}$, and that each triple K is associated with a pair of weights $(w_K^+, w_K^-) \in \{(1,0), (0,1)\}$. Triples that correspond to triangles with positively labeled edges have weights $(w_K^+, w_K^-) = (1, 0)$, and are termed "positive" triples. All other triples have weights $(w_K^+, w_K^-) = (0, 1)$, and are termed "negative" triples. For this setting, neither pivoting on a pair of vertices (e.g., an edge) nor pivoting on a single vertex may provide constant approximation guarantees, as demonstrated by the examples in Figure 1. Both graphs are complete graphs but for ease of interpretation, only positively labeled edges are depicted. In the first case, one chooses a (positive) edge uniformly at random and includes in the cluster all positive edges connected to the pivoting edge. For Figure 1 a), the optimal clustering comprises two clusters, $C_1 = \{v_1, v_2, v_3\}$ and $C_2 = \{v_4, v_5, v_6\}$ and it has an MCC objective function value equal to zero. If one pivots on the edge (v_1, v_4), the resulting clustering contains one cluster only, $C1 = \{v_1, v_2, \ldots, v_6\}$, and this leads to a positive value of the objective function, and hence an unbounded ratio of the optimal and approximate objective. Pivoting on vertices may fail as well, which may be seen from Example b): The graph in b) has a unique optimal clustering with two clusters $C_1 = \{v_1, v_2, v_3\}$ and $C_2 = \{v_4, v_5, ..., v_n\}$. Choosing the vertex v_3 as pivot and including all vertices connected to v_3 through positive edges leads to v_1, v_2, v_4 being clustered together with v_3, thereby resulting in $\Omega(n^2)$ more errors than those incurred by the optimal clustering. As there are n vertices in the graph, the expected value of the objective may have a $\Omega(n)$ error.

III. Main Results

We describe next polynomial-time, constant approximation algorithms for the MCC and MMCC problems. For the former case, we propose two methods that offer different trade-offs between optimization performance and complexity, as measured in terms of the number of constraints used in the underlying LP program. The approach followed is to relax the IPs of (1) and (2) to LPs and then perform rounding of the fractional solutions. The main analytical difficulties encountered are that the LPs involve both edge and higher-order motif variables, and that trying to round all these variables simultaneously may cause inconsistencies and large rounding errors. More precisely, in the LP formulation one has to incorporate variables associated with k-tuples, while rounding only works with variables associated with pairs of vertices. To overcome this issue for the MCC problem, our first solution introduces motif variables in the LP and then performs rounding on edges by assigning to them a cost that reflects the value of the best-scoring motif that includes the edge. The second solution is based on an LP which involves both motif and edge variables and allows downstream rounding to be performed directly on the edge variables. The second method has fewer constraints in the underlying LP than the first method, and is hence more computationally efficient. The drawback is that it provides worse approximation guarantees than the first method. For the MMCC problem, one may use the second method developed for the MCC problem with the inclusion of additional constraints for k-tuple and edge variables. The approximation factor is determined by the size of largest motifs. An in-depth study of the trade-off between the achievable approximation ratio and the number of constraints is out of the scope of this work.

As in the formulation of the MCC problem, let K correspond to a k-tuple and let x_K denote the indicator variable for the event that the vertices in K are split among clusters (i.e., $x_K = 0$ if the vertices of K lie in the same cluster, and $x_K = 1$ otherwise). Relaxing the above integral constraint to $x_K \in [0, 1]$ and rewriting the probability weight constraints leads to the following relaxed MCC optimization problem:

$$\textbf{LP1:} \quad \min_{\{x_K\}} \sum_{K \in \mathcal{K}(V)} w_K^+ x_K + w_K^-(1 - x_K)$$

$$\text{s.t.} \quad x_K \in [0,1] \qquad (\text{for all } K \in \mathcal{K}(V))$$
$$x_{K_3} \le x_{K_1} + x_{K_2} \ (\text{for all } (K_1, K_2, K_3) \in \Upsilon)$$

where

$$\Upsilon = \{(K_1, K_2, K_3) \in [\mathcal{K}(V)]^3 : K_1, K_2, K_3 \text{ are distinct unordered k-tuples, } K_1 \cap K_2 \neq \emptyset,\ K_3 \subseteq K_1 \cup K_2\}.$$

Note that the constraints imposed on triples in Υ ensure that if two motifs share vertices and belong to the same cluster, the additional motifs formed by the vertices also belong to the same cluster. We refer to these as "triangle constraints", since they generalize the triangle inequality, which is obtained as a special case when $k = 2$ (by thinking of x_{uv} as a "distance" between u and v).

The LP solutions are rounded according to Algorithm 1, described below. The intuition behind the rounding algorithm is to use the fractional solutions of the LP k-tuple variables to perform rounding on pairs of variables. The reason for using different variables in the LP and in the rounding procedure is that the LP constraints are harder to state and analyze via pairwise variables, while rounding is harder to perform via k-tuple variables as they incur complex codependencies. The key is to transition from k-tuples to pairs of variables by recording the "best motif" to which an edge belongs to, and then using the corresponding fractional value of the motif variable to perform neighborhood growing via edge incorporation.

Algorithm 1 Rounding Procedure With $\alpha \le \frac{1}{k}$

```
Set S = V
while |S| ≥ k do
  Choose an arbitrary pivot vertex v in S
  For all u ∈ S \ {v}, compute y_vu = min_{K⊆S:v,u∈K} x_K
  Let N_α(v) = {u ∈ S \ {v} : y_vu ≤ α}
  if Σ_{j∈N_α(v)} y_vu > (α/2)|N_α(v)| then
    Output the singleton cluster {v}
    Let S = S \ {v}
  else
    Output the cluster C = N_α(v) ∪ {v}
    Let S = S \ C
  end if
end while
Output all clusters C
```

Theorem III.1. Let k be a constant motif size. For any $\alpha \le \frac{1}{k}$ and the probability constraint $w_K^+ + w_K^- = 1$ satisfied by every motif K of size k, the LP coupled with the rounding procedure of Algorithm 1 provides a $\frac{2}{\alpha}$-approximate solution to the MCC problem.

Proof. The proof is given in Appendix. □

Simple algebraic manipulations (see Appendix) lead to

$$|\Upsilon| = \sum_{i=k+1}^{2k-1} \binom{|V|}{i} \left[\binom{i}{k} \binom{k}{2k-i} /2 \right] \left[\binom{i}{k} - 2 \right]. \quad (3)$$

For constants $k \ll n$, $|\Upsilon| = \Theta(n^{2k-1})$ since the dominating term in the sum is indexed by $i = k$. This indicates that the number of constraints in the LP grows exponentially with the size of the motif, which may lead to computational issues when the motifs are large. The next LP has a significantly smaller number of triangle constraints, reduced from $\Omega(n^{2k-1})$ to $\Omega(n^3)$. In particular, this LP excludes a number of triangle inequalities as constrains. One cannot reduce the number of constraints below $\Theta(n^k)$, as $\Theta(n^k)$ variables are needed to represent all possible k-tuples.

To describe the LP, we introduce some auxiliary variables. Let z_{vu}, $v, u \in V$, denote the indicator of the event that a pair of vertices v, u belong to different clusters (i.e., $z_{vu} = 0$ if v and u belong to the same cluster, and $z_{vu} = 1$ otherwise). By replacing the indicator variables by $z_{vu} \in [0,1]$ and letting $x_K \in [0,1]$ as before, we arrive at the following LP problem formulation.

$$\textbf{LP2:} \quad \min_{\{x_K\},\{z_{vu}\}} \ \sum_{K \in \mathcal{K}(V)} w_K^+ x_K + w_K^-(1 - x_K) \quad (4)$$

$$\text{s.t.} \quad x_K \ge z_{vu} \quad (\text{for all } K \in \mathcal{K}(V) \text{ and } v, u \in K), \quad (5)$$

$$x_K \le \frac{1}{k-1} \sum_{v,u \in K, v<u} z_{vu}, \quad x_K \le 1, \quad (\text{for all } K \in \mathcal{K}(V)), \quad (6)$$

$$z_{vu} \ge 0 \qquad (\text{for all } u, v \in V),$$

$$z_{v_2 v_3} \le z_{v_1 v_2} + z_{v_1 v_3} \quad (\text{for all distinct vertices } v_1, v_2, v_3 \in V).$$

A simple counting argument reveals that the number of constraints in the LP equals $\Theta(\binom{n}{k}\binom{k}{2} + \binom{n}{3})$. Note that the inequalities (5) and (6) handle constraints on the k-tuples: Placing any pair of vertices in K across clusters places K across clusters, and placing K across clusters causes placing at least $k-1$ many pairs of vertices across clusters. For the practically most relevant case $k = 3$, the number of constraints in the above described optimization problem is roughly twice that of classical LP-based correlation clustering solvers [4].

Algorithm 2 describes the rounding procedure for the solution of **LP2**. In this case, the procedure reduces to the classical region growing method of [2], [4].

Algorithm 2 Rounding Procedure With Parameters $\alpha, \beta \le \frac{1}{k}$

```
Let S = V(G)
while |S| ≥ k do
  Choose an arbitrary pivot vertex v in S
  Let N_α(v) = {u ∈ S \ {v} : z_vu ≤ α}
  if Σ_{u∈N_α(v)} z_vu > βα|N_α(v)| then
    Output the singleton cluster {v}
    Let S = S \ {v}
  else
    Output the cluster C = N_α(v) ∪ {v}
    Let S = S \ C
  end if
end while
Output S
```

Theorem III.2. Let k be a constant size of a motif. For any $\alpha, \beta \leq \frac{1}{k}$ and the probability constraint $w_K^+ + w_K^- = 1$ satisfied by every motif K of size k, the LP coupled with the rounding procedure of Algorithm 2 provides a $\frac{1}{\alpha\beta}$-approximate solution to the MCC problem.

Proof. The proof of the theorem is presented in Appendix. □

Observe that the approximation guarantees of Theorem 2 are worse than those of Theorem 1, which is the price paid for reducing the number of constraints.[1] Furthermore, since the rounding procedure operates on pairs of vertices only and does not involve variables for k-tuples, it may be used for solving the MMCC problem as well. We outline the corresponding result in what follows.

Let $S = \{k_1, k_2, ..., k_p\}$ be the set of motif sizes of interest, and let $\mathcal{K}_t(V)$ be the set of all k_t-tuples of V. Using the same notation as in the MCC version of the problem, we state the following LP relaxation for the MMCC problem:

$$\textbf{LP3:} \quad \min_{\{x_K\},\{z_{uv}\}} \quad \sum_{t=1}^{p} \lambda_t \left[\sum_{K \in \mathcal{K}_t(V)} w_K^+ x_K + w_K^-(1 - x_K) \right]$$

$$\text{s.t. } x_K \geq z_{vu} \quad \text{(for all } K \in \mathcal{K}_t(V),\ t \in [p] \text{ and } u, v \in E(K)),$$

$$x_K \leq \frac{1}{k-1} \sum_{v,u \in K, v < u} z_{vu}, \quad x_K \leq 1,$$

$$\text{(for all } K \in \mathcal{K}_t(V),\ t \in [p]),$$

$$z_{vu} \geq 0 \qquad \text{(for all } u, v \in V),$$

$$z_{v_2 v_3} \leq z_{v_1 v_2} + z_{v_1 v_3}$$

$$\text{(for all distinct vertices } v_1, v_2, v_3 \in V).$$

The rounding method accompanying this LP is also described in Algorithm 2, with the parameters α, β bounded from above by $\frac{1}{k^*}$, where $k^* = \max S = \max\{k_1, k_2, ..., k_p\}$.

Corollary III.3. For $\alpha, \beta \leq \frac{1}{k^*}$, and all motif weights satisfying the probability constraint $w_K^+ + w_K^- = 1$, the rounded LP algorithm provides an $\frac{1}{\alpha\beta}$-approximate solution to the MMCC problem.

Proof. Note that the simplest way to prove this result is to focus on the largest motif only, and use the previously described MCC result. In particular, the stated result does not depend on the particular choices of the parameter λ. □

Still, one can derive more precise and stronger approximation guarantees by focusing on all motifs simultaneously, in which case the analysis becomes rather tedious and involved. For the special case of two motifs ($p = 2$) with sizes $k_1 = 2$ and $k_2 = k$ respectively, we provide tighter approximation results in Theorem III.4. Here, both the parameters α, β depend on λ. The underlying derivations are relegated to Appendix.

Theorem III.4. Consider the MMCC problem with two types of motifs of sizes $k_1 = 2$ and $k_2 = k$. The objective function **LP3** may be rewritten as

$$\sum_{u,v \in V} \left[w_{uv}^+ z_{uv} + w_{uv}^-(1 - z_{uv}) \right] + \lambda \sum_{K \in \mathcal{K}} \left[w_K^+ x_K + w_K^-(1 - x_K) \right],$$

where z_{uv} and x_K are variables associated with pairs of vertices and k-tuples of vertices, and λ is a parameter that can be tuned to balance the penalties induced by edges and motifs of size k.

Let r_0 be a constant equal to

$$r_0 = \frac{k-2}{1 + \lambda n^{k-1}}.$$

Then, for any $\alpha \leq 1/k, \beta \leq 1/(k - r_0)$, and provided that the weights satisfy the probability constraint $w_K^+ + w_K^- = 1$ for both k-tuples and edges (i.e., $w_{uv}^+ + w_{uv}^- = 1$), the LP and rounding procedure of Algorithm 2 produce a $\frac{1}{\alpha\beta}$-approximate solution to the edge-motif MMCC problem.

Note that Theorem III.4 is particularly useful when $\lambda \sim O(n^{-(k-1)})$. As the number of k-tuples over the number of edges is $\Theta(n^{k-1})$, the condition $\lambda \sim O(n^{-(k-1)})$ essentially captures the case when the averaged contribution of k-tuples is at best comparable to the averaged contribution of edges.

IV. Numerical Results for Small Social Networks

We evaluated our (M)MCC methods on the well known Zachary karate club network [17], and two benchmark networks from [10], which were originally tested using the method described in [10] (henceforth termed TSC). In all the experiments, we considered motifs of size $k = 2$ and $k = 3$ only. Hence, one of the motifs are edges and for the case $k = 3$, the motif may be selected based on the particular application, as subsequently described. When solving the MCC problem, we use the **LP2** formulation as it contains fewer constraints than **LP1** and thus can be solved more efficiently. We then leverage Algorithm 2 for downstream rounding. When solving the MMCC problem, we use a combination of **LP3** and Algorithm 2. The (M)MCC problems rely on solving LPs with a large number of constraints. As a consequence, current (M)MCC solvers do not have desired scalability properties but may be improved in terms of using two approaches, one of which involves exploiting the fact that sparsity of motifs leads to sparse effective constraints; and alternatively, subsampling the set of constraints and then sub-optimally solving the LP problem. In the latter context, deriving performance guarantees may be challenging and dependent on the selected constraints. Note that the TMC methods proposed in [10] do random walks according to heuristically reduced higher-order Markov chains, which may be more efficient, but they do not hold provable performance guarantees like the methods in this work. Thus, an important open problem is to bridge this gap between theory and practice.

A. A Benchmark Social Network: Zachary's Karate Club [17]

We first test the performance of the CC, MCC and MMCC methods on the Zachary's karate club network. In the CC

[1] Note that Theorem 2 is a natural generalization of the results first described in the preliminary version of this work [1]. The analysis of Algorithm 2 described in [1] was tightened in [15], establishing an approximation constant of $4(k-1)$, which is worse than the result of Theorem 1.

TABLE I

WEIGHT ASSIGNMENTS FOR THE KARATE CLUB NETWORK; K_3 STANDS FOR A TRIANGLE (COMPLETE GRAPH ON 3 VERTICES), WHILE P_3 DENOTES A PATH WITH THREE VERTICES

	Edges		Motifs			λ
Subgraphs	Non-edges	Edges	Non-motifs	K_3	P_3	
CC	0.47	1	—	—	—	0
MCC	—	—	0.49	1	2/3	—
MMCC	0.45	1	0.5	1	2/3	0.2

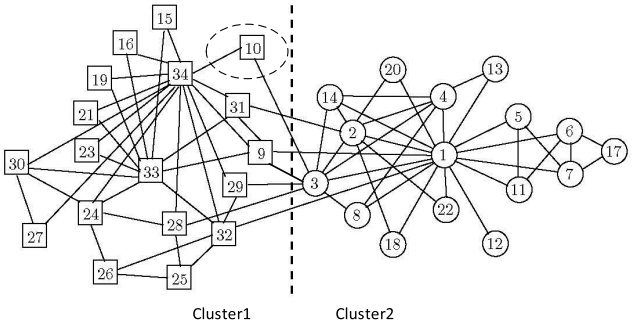

Fig. 2. Clustering results for the CC, MCC and MMCC method performed on the Zachary's karate club network. Vertex 10 is erroneously clustered by the CC method, but correctly clustered by both the MCC and MMCC methods.

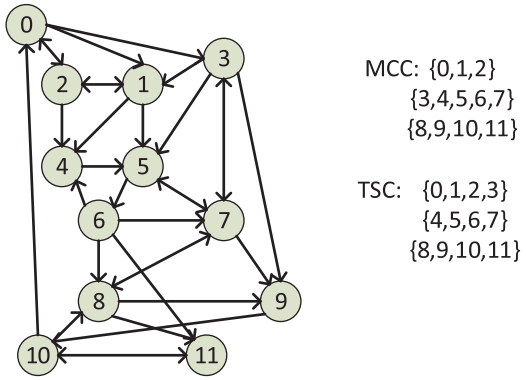

Fig. 3. Example of a flow network, with layers detection performance of MCC and TSC. Left: The layered flow network; Right: The clustering results.

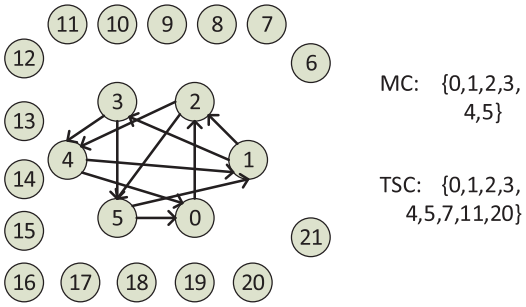

Fig. 4. Anomaly detection in networks and clustering.

model, we assign weights to each pair of vertices depending on whether they are connected by an edge or not. For the MCC method, we focus on 3-tuples and assign weights to the 3-tuple weights according to whether their corresponding vertices form a triangle or a path. We use both triangles (K_3) and 3-paths (P_3) as motifs to ensure that nodes with very small degree can be clustered more accurately by examining their inclusion into important motifs involving vertices of large degree. The MMCC method uses both 2-tuples and 3-tuples. The weight assignments used in all the described methods are listed Table I. The result is shown in Figure 2. Although we tested CC for a number of choices for the weights, we inevitably ended up with one clustering error, vertex 10. This vertex is connected to vertex 34 in Cluster 1 and vertex 3 in Cluster 2. On the other hand, the MCC and MMCC methods recovered the ground truth clustering by taking into account the K_3 and P_3 motifs. The reason for this finding is that in social networks, vertices within a cluster typically connect to some central vertices in the same cluster (like vertex 34 and vertex 1). Hence, they form many triangles and 3-paths containing the central vertices.

B. Partitioning Layered Flow Networks

The next example is what we refer to as a *layered flow network* (see Figure 3). The information flow between two layers typically follows the same direction while feedback loops are primarily contained within a layer. The task is to detect the layers in the network. To perform the layer clustering, we assign the value 1 to each weight w_K corresponding to a directed 3-cycle (i.e., a triple $\{j_1, j_2, j_3\}$ with edges directed according to $j_1 \to j_2$, $j_2 \to j_3$, $j_3 \to j_1$, or the reverse order), encouraging the corresponding triples to lie within a layer, while we assign a arbitrary weight in $[0.41, 0.48]$ to all other type of triples. The choice of the weights of the negative edges is governed by the fact that there are significantly more triples other than directed 3-cycles, constraining the weights to be close to, but slightly smaller than 0.5. The clustering results are shown in Figure 3. Both MCC and the method of [10] produce similar clustering results, which identify the layers of the network. The only difference is observed for the node with label 3. The MCC method emphasizes the feedback loops inside a layer, and hence node 3 is placed in the same cluster as nodes $4, 5, 6, 7$. The other method emphasizes the importance of the direction of information flow and thus the flow from node 3 to node 1 does not permit clustering nodes $3, 4, 5, 6, 7$ together.

C. Anomaly Detection

Practical networks usually contain bidirectional edges, i.e., edges that allow both directions of traversal. A large number of these edges lie within directed 3-cycles [10]. Hence, if a part of a network contains many directed 3-cycles but very few bidirectional edges, it may be viewed as an anomaly.

An illustrative example is shown in Figure 4, in which the nodes labeled 0-5 form an anomalous component which we wish to detect as it contains 8 directed 3-cycles without any bidirectional edges. The edges between nodes 6-21 are generated according to a standard Erdős-Rényi model with probability 0.25 and to keep the figure simple, those edges

were not plotted. Note that each of the nodes labeled $0-5$ has 4 outgoing and 2 incoming edges within the group of vertices containing $6-21$. There are 20 directed triangles without bidirectional edges.

To use our MCC method, we set the weights for the triangles without bidirectional edges to 1, and those for other types of triangles to a value smaller than 0.42. As the results shown in the Figure 4 demonstrate, our method outperforms the TSC method in terms of detecting the anomaly.

APPENDIX

To prove that the problem is in NP, we focus our attention on the case $(w_K^+, w_K^-) \in \{(1,0),(0,1)\}$. Since $w_K^+ \in \{0,1\}$, as before, we refer to a triple K with $w_K^+ = 1$ (respectively, $w_K^+ = 0$) as "positive" (respectively, "negative"). We also use the term "positive error" to indicate that a positive triple is placed across clusters and "negative error" to indicate that a negative triple is placed within one cluster.

Following the approach used to prove NP-hardness of CC [2], we use a reduction from the NP-complete Partition into Triangles [18] problem. Given a (not necessarily complete) graph $G_\triangle = (V, E)$, containing n vertices where n is a multiple of 3, the goal is to decide whether it can be partitioned into triangles.

As the first step in our proof, we construct a graph G^w that has the same vertex set as $G_\triangle$ and view triangles of G^w as motifs. We set the weights of triples G^w that correspond to triangles in $G_\triangle$ to $(1,0)$, and the weights of all other triples in G^w to $(0,1)$. We solve the MCC problem over G^w under the additional constraint that the size of each cluster is at most 3. The existence of an efficient algorithm for solving this MCC would imply the existence of an efficient algorithm for partitioning $G_\triangle$ into triangles, a contradiction. As the original MCC algorithm does not necessarily generate clusters with bounded size 3, in what follows we describe how to construct another graph, H^w, such that the triples-MCC algorithm applied on H^w results in a bounded cluster-size run of MCC on G^w.

The basic idea behind our approach is to impose the constraint on the size of clusters in G^w by adding vertices in H^w for each triple in G^w, and then making the triples formed by the the newly added vertices positive and other triples negative. In this way, a cluster in the new graph H^w with more than 3 vertices in G^w causes a large number of negative errors and hence cannot be part of an optimal clustering.

We now describe how to construct the graph H^w based on G^w. In addition to the vertices of G^w, for every triple $\{u_1,u_2,u_3\}$ in G^w, H^w contains additional n^5 vertices, denoted by $C_{u_1u_2u_3}$. For simplicity of notation, we write $C_{u_1u_2u_3} \cup \{u_1,u_2,u_3\} = C'_{u_1,u_2,u_3}$. Clearly, H^w contains $n + n^5\binom{n}{3}$ vertices. We classify the triples in H^w into three types:

1) T-I triples: $\{u_1,u_2,u_3\}$, for all $u_1,u_2,u_3 \in V(G^w)$.
2) T-II triples: triples in C'_{u_1,u_2,u_3} that are not T-I triples.
3) T-III triples: triples that are neither T-I triples nor T-II triples.

The number of T-I triples is $\binom{n}{3}$. As they are inherited from G^w, we keep their weights equal to those in G^w. The number of T-II triples equals $\binom{n}{3}[\binom{n^5+3}{3}-1]$, and we assign to them the weights $(1,0)$. The number of T-III triples equals $\binom{n+n^5\binom{n}{3}}{3} - \binom{n}{3}\binom{n^5+3}{3}$, and we assign to them the weights $(0,1)$.

Consider now a clustering $\mathcal{C}^*$ of H^w of the following form:

1) There are $\binom{n}{3}$ nonoverlapping clusters;
2) Each cluster corresponds to one of the sets $C_{u_1u_2u_3}$ or one of the sets $C'_{u_1u_2u_3}$;
3) Each vertex u inherited from $V(G^w)$ lies in exactly one cluster.

In the above clustering, there are no errors arising due to T-III triples, because all T-III triples are negative and $\mathcal{C}^*$ has property 2). The only errors arise from T-I triples and T-II triples. The number of errors induced by T-I triples is at most $\binom{n}{3}$, while T-II triples errors in $\mathcal{C}^*$ may be grouped into two categories. First, a triple may have two vertices in $C_{u_1u_2u_3}$ and one vertex in $\{u_1,u_2,u_3\}$ that lies in another cluster. The number of this type of clustering errors is bounded from above by $n(\binom{n-1}{2}-1)\binom{n^5}{2}$. Second, a triple may have one vertex in $C_{u_1u_2u_3}$ and two vertices in $\{u_1,u_2,u_3\}$ that lie in another cluster. The number of this type of errors is upper bounded by $\binom{n}{2}(n-3)\binom{n^5}{1}$. Therefore, the total number of errors in $\mathcal{C}^*$ is bounded from above by

$$n\left(\binom{n-1}{2}-1\right)\binom{n^5}{2} + \binom{n}{2}(n-3)\binom{n^5}{1} + \binom{n}{3} \sim O(n^{13}).$$

We may convert the clustering $\mathcal{C}^*$ into a partition G^w based on the clustering of T-I triples. The clustering $\mathcal{C}^*$ essentially *partitions the vertices of* G^w into clusters containing exactly three vertices. Our subsequent arguments aim to establish that the number of errors in a clustering that contains at least one cluster with at least four vertices from $V(G)$ must be larger than the number of errors induced by $\mathcal{C}^*$.

For that purpose, consider another clustering of H^w, denoted by $\mathcal{C}'$. First, we show that in order for $\mathcal{C}'$ to have fewer errors than $\mathcal{C}^*$, the size of any cluster in $\mathcal{C}'$ must lie in the interval $[n^5-n^4, n^5+n^4]$. Suppose that on the contrary there exists a cluster containing more that n^5+n^4 vertices. Then, there are at least $\binom{n^5}{2}n^4 \sim \Omega(n^{14})$ negative errors caused by placing T-III triples into this cluster. Furthermore, each cluster must contain at least n^5-n^4 vertices of a clique, otherwise there are at least $\binom{n^5}{2}n^4 \sim \Omega(n^{14})$ positive errors generated by splitting the T-II triples. Second, note the each vertex in $V(G^w)$ belongs to $\binom{n-1}{2}$ different triples of G^w. Since the size of each cluster of $\mathcal{C}'$ is smaller than n^5+n^4, for each vertex in $V(G)$, the number of negative errors caused by splitting the T-II triples that contains this vertex and two vertices from some $C_{u_1u_2u_3}$ is lower bounded by $\binom{n^5}{2}\binom{n-1}{2} - \binom{n^5}{2} - \binom{n^4}{2}$.

Assume now that there exists a cluster of $\mathcal{C}'$ that contains four vertices inherited from $V(G^w)$, say $\{u_1,u_2,u_3,u_4\}$. Then, as the size of the cluster is lower bounded by n^5-n^4, from the pigeonhole principle it follows that there exists at least one vertex in $\{u_1,u_2,u_3,u_4\}$, say j_1, and at least $\frac{1}{4}(n^5-n^4)$ other vertices that do not lie in one of the sets $C_{u_1\,u'u''}$ for some $u', u'' \in v(G^w)$. Hence, the number of negative errors caused by T-III triples within this cluster is at least $\binom{\frac{1}{4}(n^5-n^4)}{2}$.

TABLE II

OVERVIEW OF THE DIFFERENT CASES STUDIED IN THE PROOF OF THEOREM III.1. OUTPUT REFERS TO THE OUTPUT OF THE ALGORITHM; THE "COST OF SPLITTING" AND "JOINT CLUSTERING" REFER TO THE COST OF SPLITTING THE k-TUPLE IN K OR PLACING ALL OF K INTO INTO THE OUTPUT CLUSTER, RESPECTIVELY. ADDITIONAL CONDITIONS ARE SPECIFIC FOR THE CASE UNDER INVESTIGATION

Output	Cost of	$K \ni v$	Additional conditions	Approx. constant	Case#
$\{v\}$	splitting	yes	$K \cap [S \setminus \mathcal{N}'_\alpha(v)] \neq \emptyset$	$1/\alpha$	1
	splitting	yes	$K \cap [S \setminus \mathcal{N}'_\alpha(v)] = \emptyset$	$2/\alpha$	1
$\mathcal{N}'_\alpha(v)$	joint clustering	yes	—	$1/[1-(k-1)\alpha]$	2.1
	joint clustering	no	Given $u \in \mathcal{N}_\alpha(v)$, $\exists K' \subseteq \mathcal{N}'_\alpha(v)$, $K' \ni v, u$, $x_{K'} \leq \alpha/2$	$2/[2-(2k-1)\alpha]$	2.1.1
	joint clustering	no	Given $u \in \mathcal{N}_\alpha(v)$, $\forall K' \subseteq \mathcal{N}'_\alpha(v)$, $K' \ni v, u$, $x_{K'} > \alpha/2$	$2/[2-(2k-1)\alpha]$	2.1.2
	splitting	yes	—	$1/\alpha$	2.2
	splitting	no	$\exists K'$, $K' \ni v, x_{K'} \geq 1-\alpha/2$ $K' \setminus \mathcal{N}'_\alpha(v) = K \setminus \mathcal{N}'_\alpha(v)$	$2/[2-(2k-1)\alpha]$	2.2.1
	splitting	no	$\forall K'$, $K' \ni v, x_{K'} < 1-\alpha/2$ $K' \setminus \mathcal{N}'_\alpha(v) = K \setminus \mathcal{N}'_\alpha(v)$	$2/\alpha$	2.2.2

The total number of errors induced by such a clustering is therefore at least

$$n\binom{n^5}{2}\left(\binom{n-1}{2}-1\right) - n\binom{n^4}{2} + \binom{\frac{1}{4}(n^5-n^4)}{2},$$

which is larger than the number of errors in the clustering $\mathcal{C}^*$, for n sufficiently large. Therefore, the optimal triangle-clustering has to be of the form of $\mathcal{C}^*$, imposing a constraint on the size of clusters in G^w.

Since we assume that the weights satisfy the probability constraint $w_K^+ + w_K^- = 1$, we will use w_K to refer to w_K^+ and $1 - w_K$ to refer to w_K^-.

Let $\mathcal{N}_\alpha(v)$ be the set defined in the rounding procedure. If $\mathcal{N}_\alpha(v) \neq \emptyset$, $\mathcal{N}_\alpha(v)$ contains at least $k-1$ elements, because if $x_K \leq \alpha$ for some k-tuple K, then all its elements (except possibly v) lie in $\mathcal{N}_\alpha(v)$. Let $\mathcal{N}'_\alpha(v) = \mathcal{N}_\alpha(v) \cup \{v\}$. For convenience, we also define, given a pivot vertex v and a k-tuple K that contains v, $y_K = \sum_{u \in K \setminus \{v\}} y_{vu}$. Furthermore, we let

$$K_{min}^{(uv)} = \arg \min_{K:\, u,v \in K} x_K.$$

Thus, by using the LP constraint and the definition of y_{uv}, we have

$$\frac{1}{k-1} y_K \leq x_K \leq \sum_{u \in K \setminus \{v\}} x_{K_{min}^{(uv)}} = y_K. \tag{7}$$

Let $\mathcal{K}_v$ be the set of all the k-tuples K such that $K \subseteq \mathcal{N}'_\alpha(v)$, $K \ni v$. When v is a pivot vertex and $K \in \mathcal{K}_v$, we know that

$$y_K \leq (k-1)\alpha \leq \frac{k-1}{k}. \tag{8}$$

The following proof often uses another form of the constraint in the underlying LP, i.e., $x_{K_1} \geq x_{K_3} - x_{K_2}$ for any $(K_1, K_2, K_3) \in \Upsilon$.

Next, we compare the rounding cost and the LP cost for different types of outputs of the algorithm. As the LP cost naturally gives a lower bound of the optimal cost, the ratio between the rounding cost and the LP cost characterizes the approximation ratio. Since the rounding procedure produces clusters, we also refer to the rounding cost as cluster-cost or clustering cost. All possible cases and their corresponding approximation constants are listed in Table II.

Case 1: *The output is the singleton cluster* $\{v\}$. The clustering cost when outputting a singleton $\{v\}$ is $\sum_{K \subseteq \mathcal{K}(S): v \in K} w_K$ while the LP cost is $\sum_{K \subseteq \mathcal{K}(S): v \in K} (1-w_K)(1-x_K) + w_K x_K$.

If $K \cap \left[S \setminus \mathcal{N}'_\alpha(v)\right] \neq \emptyset$, we have $x_K > \alpha$, so charging each such k-tuple $\frac{1}{\alpha} w_K x_K$ times its LP-cost compensates for the cluster-cost. Therefore, it suffices to consider the k-tuples $K \in \mathcal{K}_v$. For $K \in \mathcal{K}_v$, the LP cost is bounded by

$$\begin{aligned}
&\sum_{K \in \mathcal{K}_v} (1-w_K)(1-x_K) + w_K x_K \\
&\geq \sum_{K \in \mathcal{K}_v} (1-w_K)(1-y_K) + w_K \frac{1}{k-1} y_K \\
&= \sum_{K \in \mathcal{K}_i} w_K \left[\frac{k}{k-1} y_K - 1\right] + (1-y_K) \\
&\geq \sum_{K \in \mathcal{K}_v} \frac{1}{k-1} y_K \geq \frac{\alpha}{2} \binom{|\mathcal{N}_\alpha(v)|}{k-1},
\end{aligned}$$

where the first inequality is due to (7), the second inequality is due to (8) and $w_K \leq 1$, while the third inequality is due to the condition that the algorithm outputs a singleton cluster $\{v\}$. Therefore, charging $\frac{2}{\alpha}$ for the k-tuple is enough to compensate for the cluster-cost.

Case 2: *The output is the cluster* $\mathcal{N}'_\alpha(v)$.

Case 2.1: First, consider the cost of the k-tuples inside the cluster. If $v \in K$, then we have $K \in \mathcal{K}_v$ and thus $x_K \leq y_K \leq (k-1)\alpha$. As the LP cost is $\geq (1-w_K)(1-x_K)$ and the cluster-cost is $1 - w_K$, charging $\frac{1}{1-(k-1)\alpha}$ for this tuple suffices to compensate for the cluster-cost.

If $v \notin K$, order the vertices in $\mathcal{N}_\alpha(v)$ in such a way that for any $u_1, u_2 \in \mathcal{N}_\alpha(v)$, $u_1 \prec u_2$ iff $y_{vu_1} < y_{vu_2}$ and assign an arbitrary order ($u_1 \prec u_2$) when the equality ($y_{vu_1} = y_{vu_2}$) holds.

For each vertex $u \in \mathcal{N}_\alpha(v)$, let $R_u = \{u' \in \mathcal{N}_\alpha(v) \colon u' \preceq u\}$, and let $\mathcal{K}_v^{(u)}$ be the set of k-tuples $K \in \mathcal{N}_\alpha(v)$ such that u is the largest vertex of K according to $\prec$. Thus, if $K \in \mathcal{K}_v^{(u)}$, then $u \in K$ and $K \subseteq R_u$.

Note that because of the order, we have $\sum_{u' \in R_u} y_{vu'} \leq \frac{\alpha}{2}|R_u|$. Now for all $u \in \mathcal{N}_\alpha(v)$, let us consider the total cost of the k-tuples in R_u. The corresponding cluster-cost is $\sum_{K \in \mathcal{K}_v^{(u)}} 1 - w_K$ while the LP cost is $\sum_{K \in \mathcal{K}_v^{(u)}} (1-x_K)(1-w_K) + x_K w_K$.

Next, let $\mathcal{K}'_v$ be the set of k-tuples $K' \subseteq \mathcal{N}'_\alpha(v)$ with v, $u \in K'$.

Case 2.1.1: *There is a clique $K' \in \mathcal{K}'_v$ with $x_{K'} \leq \frac{\alpha}{2}$.*

Let $K^* = (K \setminus \{u\}) \cup \{v\}$. Observe that $v \in K'_v \cap K^*$ and that $K \subseteq K'_v \cup K^*$. Hence, the LP constraints imply that for all $K \in \mathcal{K}_v^{(u)}$, we have

$$x_K \leq x_{K'} + x_{K^*} \leq \frac{\alpha}{2} + (k-1)\alpha = \frac{(2k-1)\alpha}{2}.$$

So, charging $\frac{2}{2-(2k-1)\alpha}$ for each k-tuple in $\mathcal{K}_v^{(u)}$ is enough to compensate for the cluster-cost.

Case 2.1.2: *For all $K' \in \mathcal{K}'_v$, we have $x_{K'} > \frac{\alpha}{2}$.* Let $K \in \mathcal{K}_v^{(u)}$, and let $\{u_1, \ldots, u_{k-1}\}$ be the vertices in $K \setminus \{u\}$. Let $y_K = \sum_{u_j \in K \setminus \{g\}} y_{vu_j}$. For each $j \in \{1, \ldots, k-1\}$, let $K_j = (K \setminus \{u_j\}) \cup \{v\}$. As each $K_j \in \mathcal{K}'_v$, the LP constraints imply:

$$\begin{aligned} 1 - x_K &\geq 1 - \min_{j \in \{1,\ldots,k-1\}} \{x_{K_{min}^{(vu_j)}} + x_{K_j}\} \\ &= 1 - \min_{j \in \{1,\ldots,k-1\}} \{y_{vu_j} + x_{K_j}\} \\ &\geq 1 - \left[\frac{1}{k-1} y_K + \frac{1}{k-1} \sum_{j=1}^{k-1} x_{K_j}\right] \end{aligned}$$

and

$$\begin{aligned} x_K &\geq \max_{j \in \{1,\ldots,k-1\}} \{x_{K_j} - x_{K_{min}^{(vu_j)}}\} \\ &= \max_{j \in \{1,\ldots,k-1\}} \{x_{K_j} - y_{vu_j}\} \geq \frac{1}{k-1} \sum_{j=1}^{k-1} x_{K_j} - \frac{1}{k-1} y_K. \end{aligned}$$

Let $\sigma = \sum_{j=1}^{k-1} x_{K_j}$. Manipulating these inequalities yields, for each $K \in \mathcal{K}_v^{(u)}$,

$$\begin{aligned} &(1 - w_K)(1 - x_K) + w_K x_K \\ &\geq (1 - w_K)\left(1 - \frac{2}{k-1}\sigma\right) - \frac{1}{k-1} y_K + \frac{1}{k-1}\sigma. \end{aligned} \tag{9}$$

The LP constraints yield $x_{K_j} \leq (k-1)\alpha$ for each $j \in \{1, \ldots, k-1\}$, since $i \in K_j$ for each j, by the same argument used to establish inequality (7). Since each $K_j \in \mathcal{K}'_v$, we have $\alpha/2 \leq x_{K_j} \leq (k-1)\alpha$ for each j, so that $\sigma \in [(k-1)\frac{\alpha}{2}, (k-1)^2\alpha]$. The inequality (9) is linear in σ, so we study its behavior when σ is an endpoint of this interval. When $\sigma = (k-1)\frac{\alpha}{2}$, we obtain

$$\begin{aligned} &(1 - w_K)(1 - x_K) + w_K x_K \\ &\geq (1 - w_K)(1 - \alpha) + \frac{\alpha}{2} - \frac{1}{k-1} y_K, \end{aligned}$$

and when $\sigma = (k-1)^2\alpha$, we obtain

$$\begin{aligned} &(1 - w_K)(1 - x_K) + w_K x_K \\ &\geq (1 - w_K)(1 - 2(k-1)\alpha) + (k-1)\alpha - \frac{1}{k-1} y_K \\ &= (1 - w_K)(1 - 2(k-1)\alpha) + (k - \frac{3}{2})\alpha + \frac{\alpha}{2} - \frac{1}{k-1} y_K \\ &\geq (1 - w_K)(1 - 2(k-1)\alpha) \\ &\qquad + (1 - w_K)(k - \frac{3}{2})\alpha + \frac{\alpha}{2} - \frac{1}{k-1} y_K \\ &= (1 - w_K)(1 - (k - \frac{1}{2})\alpha) + \frac{\alpha}{2} - \frac{1}{k-1} y_K. \end{aligned}$$

Since $k \geq 2$ we clearly have $(1 - w_K)(1 - (k - \frac{1}{2})\alpha) + \frac{\alpha}{2} \leq (1 - w_K)(1 - \alpha) + \frac{\alpha}{2}$, so by linearity, we also have

$$\begin{aligned} &(1 - w_K)(1 - x_K) + w_k x_K \\ &\geq (1 - w_K)(1 - (k - \frac{1}{2})\alpha) + \frac{\alpha}{2} - \frac{1}{k-1} y_K \end{aligned} \tag{10}$$

for all $K \in \mathcal{K}_v^{(u)}$. Now, recall that $\sum_{u' \in R_u} y_{vu'} \leq \frac{\alpha}{2}|R_u|$; as every vertex in R_u appears in exactly $\binom{|R_u|-1}{k-2}$ k-tuples of $\mathcal{K}_v^{(u)}$, this implies that

$$\begin{aligned} &\frac{1}{k-1} \sum_{K \in \mathcal{K}_v^{(u)}} y_K = \frac{1}{k-1} \binom{|R_u| - 1}{k-2} \sum_{u' \in R_u} y_{vu'} \\ &\leq \frac{\alpha}{2} \binom{|R_u| - 1}{k-2} \frac{|R_u|}{k-1} = \frac{\alpha}{2} \binom{|R_u|}{k-1} = \frac{\alpha}{2} \left|\mathcal{K}_v^{(u)}\right|. \end{aligned}$$

Thus, summing inequality (10) over all tuples in $\mathcal{K}_v^{(u)}$ yields the following lower bound on the total LP-cost of these tuples:

$$\begin{aligned} &\sum_{K \in \mathcal{K}_v^{(u)}} [(1 - w_K)(1 - x_K) + w_k x_K] \\ &\geq \sum_{K \in \mathcal{K}_v^{(u)}} \left[(1 - w_K)(1 - (k - \frac{1}{2})\alpha) + \frac{\alpha}{2} - \frac{1}{k-1} y_K\right] \\ &\geq \sum_{K \in \mathcal{K}_v^{(u)}} \left[(1 - w_K)(1 - (k - \frac{1}{2})\alpha)\right] \\ &= (1 - (k - \frac{1}{2}))\alpha \sum_{K \in \mathcal{K}_v^{(u)}} (1 - w_K). \end{aligned}$$

Therefore, charging $\frac{2}{2-(2k-1)\alpha}$ for each k-tuple in $\mathcal{K}_v^{(u)}$ suffices to compensate for the cluster-cost.

Case 2.2: *Compensating the cost of splitting a k-tuple.* Each tuple K split during clustering incurs a cluster-cost of w_K and an LP-cost of $x_K w_K + (1 - x_K)(1 - w_K)$. First, suppose that K is a split k-tuple. Since K was split, $x_K > \alpha$, and charging $\frac{1}{\alpha}$ times the LP cost pays for such a K.

Let $S' \subseteq S \setminus \mathcal{N}'_\alpha(v)$ be such that $|S'| \leq k-1$. Furthermore, let $\mathcal{K}_v^{(S')}$ be the set of split tuples K such that $v \notin K$ and $K \setminus \mathcal{N}'_\alpha(v) = S'$. According to the definition of S', for any split tuple K, there is a corresponding S'. We show that the total cluster-cost of the tuples in $\mathcal{K}_v^{(S')}$ is at most a constant times their total LP-cost. To establish the claim, let $\mathcal{S}_\mathcal{N}$ be the collection of all subsets $S_\mathcal{N} \subseteq \mathcal{N}_\alpha(v)$ with $|S_\mathcal{N}| = k-1-\left|S'\right|$.

Case 2.2.1: *There is some $S_\mathcal{N} \in \mathcal{S}_\mathcal{N}$ such that $x_{\{v\} \cup S_\mathcal{N} \cup S'} \geq 1 - \frac{\alpha}{2}$.* For each $K \in \mathcal{K}_v^{(S')}$, let $\tilde{S} = K \cap \mathcal{N}_\alpha(v)$, and take an arbitrary set $\bar{S} \subseteq \mathcal{N}_\alpha(v) \setminus \tilde{S}$ with $\left|\bar{S}\right| = k - 1 - |\tilde{s}|$. We have $x_{\{v\} \cup \tilde{S} \cup \bar{S}} \leq (k-1)\alpha$ and thus

$$\begin{aligned} x_K &\geq x_{\{v\} \cup S' \cup S_\mathcal{N}} - x_{\{v\} \cup \tilde{S} \cup \bar{S}} \\ &\geq 1 - \frac{\alpha}{2} - (k-1)\alpha = \frac{2 - (2k-1)\alpha}{2}, \end{aligned}$$

and in particular $x_K \geq \frac{2-(2k-1)\alpha}{2}$ for all $K \in \mathcal{K}_v^{(S')}$. Thus, charging $\frac{2}{2-(2k-1)\alpha}$ times the LP-cost to each $K \in \mathcal{K}_v^{(S')}$ pays for the cluster-cost of all such edges.

Case 2.2.2: *For all $S_\mathcal{N} \in \mathcal{S}_\mathcal{N}$, $x_{\{i\} \cup S_\mathcal{N} \cup S'} < 1 - \frac{\alpha}{2}$.* Take any $K \in \mathcal{K}_v^{(S')}$ and let $\tilde{S} = K \cap \mathcal{N}_\alpha(v)$. Suppose that

$\tilde{S} = \{u_1, u_2, ..., u_{|\tilde{S}|}\}$. For each $u_j \in \tilde{S}$, let $K_j = (K \setminus \{u_j\}) \cup \{v\}$. Note that each tuple K_j is a split tuple. We have:

$$\begin{aligned} 1 - x_K &\geq 1 - \min_{u_j \in \tilde{S}} [x_{K_{min}^{(vu_j)}} + x_{K_j}] \\ &= 1 - \min_{u_j \in \tilde{S}} [y_{vu_j} + x_{K_j}] \\ &\geq 1 - \frac{1}{\left|\tilde{S}\right|} \sum_{u_j \in \tilde{S}} (y_{vu_j} + x_{K_j}); \\ x_K &\geq \max_{u_j \in \tilde{S}} [x_{K_j} - x_{K_{min}^{(vu_j)}}] = \max_{u_j \in \tilde{S}} [x_{K_j} - y_{vu_j}] \\ &\geq \frac{1}{\left|\tilde{S}\right|} \sum_{u_j \in \tilde{S}} (x_{K_j} - y_{vu_j}). \end{aligned}$$

Let $\sigma_x = \sum_{u_j \in \tilde{S}} x_{K_j}$ and let $\sigma_y = \sum_{u_j \in \tilde{S}} y_{K_j}$. The inequalities above yield the following lower bound on the LP-cost of K:

$$\begin{aligned} &(1 - w_K)(1 - x_K) + w_K x_K \\ &\geq w_K \left[\frac{2}{\left|\tilde{S}\right|}\sigma_x - 1\right] + 1 - \frac{1}{\left|\tilde{S}\right|}(\sigma_x + \sigma_y). \end{aligned} \tag{11}$$

We have $x_{K_j} \geq \alpha$ by definition and $x_{K_j} \leq 1 - \frac{2}{\alpha}$ due to the assumptions made for this case. Thus, we have $\sigma_x \in [\alpha \left|\tilde{S}\right|, (1 - \frac{\alpha}{2}) \left|\tilde{S}\right|]$. As the lower bound in inequality (11) is linear in σ_x, we study the behavior of the bound at the endpoints. When $\sigma_x = \alpha \left|\tilde{S}\right|$, we have

$$\begin{aligned} &(1 - w_K)(1 - x_K) + w_K x_K \geq (2\alpha - 1)w_K + 1 - \alpha - \frac{\sigma_y}{\left|\tilde{S}\right|} \\ &\geq (2\alpha - 1)w_K + (1 - \frac{3\alpha}{2})w_K + \frac{\alpha}{2} - \frac{\sigma_y}{\left|\tilde{S}\right|} \\ &= \frac{\alpha}{2}w_K + \frac{\alpha}{2} - \frac{\sigma_y}{\left|\tilde{S}\right|}. \end{aligned}$$

Here, we used the fact that $\alpha < 2/3$. When $\sigma_x = (1 - \frac{\alpha}{2}) \left|\tilde{S}\right|$, we obtain

$$(1 - w_K)(1 - x_K) + w_K x_K \geq (1 - \alpha)w_K + \frac{\alpha}{2} - \frac{\sigma_y}{\left|\tilde{S}\right|}.$$

Since $\alpha \leq 2/3$, we have $1 - \alpha \geq \frac{\alpha}{2}$, so that the inequality

$$(1 - w_K)(1 - x_K) + w_K x_K \geq \frac{\alpha}{2}w_K + \frac{\alpha}{2} - \frac{\sigma_y}{\left|\tilde{S}\right|} \tag{12}$$

holds for σ_x at both endpoints of the interval, and thus holds for all $K \in \mathcal{K}_v^{(S')}$. With $\tilde{S} = K \cap \mathcal{N}_\alpha(v)$ as before, we have $\left|\tilde{S}\right| = k - \left|S'\right|$ for all $K \in \mathcal{K}_v^{(S')}$, and indeed the map $K \mapsto (K \cap \mathcal{N}_\alpha(v))$ is a bijection from $\mathcal{K}_v^{(S')}$ to $\binom{\mathcal{N}_\alpha(v)}{k-|S'|}$. Since each vertex of $\mathcal{N}_\alpha(v)$ lies in exactly $\binom{|\mathcal{N}_\alpha(v)|-1}{k-|S'|-1}$ of the sets in $\binom{\mathcal{N}_\alpha(v)}{k-|S'|}$, we have

$$\begin{aligned} \sum_{K \in \mathcal{K}_v^{(S')}} \frac{1}{\left|\tilde{S}\right|} \sum_{u_j \in \tilde{S}} y_{vu_j} &= \sum_{\tilde{S} \in \binom{|\mathcal{N}_\alpha(i)|}{k-|S'|}} \frac{1}{k - |S'|} \sum_{u_j \in \tilde{S}} y_{vu_j} \\ &= \frac{1}{k - |S'|} \binom{|\mathcal{N}_\alpha(i)| - 1}{k - |S'| - 1} \sum_{u \in \mathcal{N}_\alpha(v)} y_{vu} \\ &\leq \frac{1}{k - |S'|} \binom{|\mathcal{N}_\alpha(v)| - 1}{k - |S'| - 1} \frac{\alpha \left|\mathcal{N}_\alpha(v)\right|}{2} \\ &= \frac{\alpha}{2} \binom{|\mathcal{N}_\alpha(v)|}{k - |S'|} = \frac{\alpha}{2} \left|\mathcal{K}_v^{(S')}\right|. \end{aligned}$$

Thus, summing inequality (12) over all tuples in $\mathcal{K}_v^{(S')}$ yields the following lower bound on the total LP-cost of the underlying tuples:

$$\begin{aligned} &\sum_{K \in \mathcal{K}_v^{(S')}} [(1 - w_K)(1 - x_K) + w_K x_K] \\ &\geq \sum_{K \in \mathcal{K}_v^{(S')}} \left[\frac{\alpha}{2}w_K + \frac{\alpha}{2} - \frac{\sigma_y}{\left|\tilde{S}\right|}\right] \\ &\geq \frac{\alpha}{2} \sum_{K \in \mathcal{K}_v^{(S')}} w_K. \end{aligned}$$

Thus, charging each tuple in $\mathcal{K}_v^{(S')}$ a factor of $\frac{2}{\alpha}$ times its LP-cost is enough to pay for the cluster-cost.

In summary, if $\alpha = 1/k$ and we define $c = \max\{\frac{1}{1-\alpha}, \frac{1}{1-(k-1)\alpha}, \frac{2}{2-(2k-1)\alpha}, \frac{2}{\alpha}\} = \frac{2}{\alpha} = 2k$, then Algorithm 1 charges each k-tuple at most a factor of $2k$ times its LP.

We continue to use the notation introduced in Appendix. In particular, we let $\mathcal{N}'_\alpha(v) = \mathcal{N}_\alpha(v) \cup \{v\}$ and let $\mathcal{K}_v$ be the set of all k-tuples K such that $K \subseteq \mathcal{N}'_\alpha(v)$, $K \ni v$. The following proof often uses some immediate consequences of the LP constraints; here we adopt the convention that $z_{uu} = 0$ for all $u \in V$:

1) $z_{u_1u_2} \geq z_{u_1u_3} - z_{u_2u_3}$ for any $u_1, u_2, u_3 \in V$;
2) $x_K \geq \max_{uu' \in K} z_{uu'} \geq \max_{u,u' \in K} [z_{vu} - z_{vu'}]$, for any $v \in V$;
3) $$\begin{aligned} x_K &\leq \frac{1}{k-1} \sum_{u,u' \in K, u < u'} z_{uu'} \\ &\leq \frac{1}{k-1} \sum_{u,u' \in K, u < u'} (z_{vu} + z_{vu'}) \leq \sum_{u \in K} z_{vu}, \end{aligned}$$ for any $v \in V$.

As before, we prove the approximation guarantees by comparing the rounding cost and the LP cost. An overview of the different cases encountered and the corresponding approximation constants is provided in Table III.

Case 1: *The output is the singleton cluster* $\{v\}$. The clustering cost when outputting a singleton $\{v\}$ is $\sum_{K \subseteq \mathcal{K}(S): v \in K} w_K$ while the LP cost is $\sum_{K \subseteq \mathcal{K}(S): i \in K} (1 - w_K)(1 - x_K) + w_K x_K$.

If $K \cap [S \setminus \mathcal{N}'_\alpha(v)] \neq \emptyset$, we have $x_K > \alpha$, so charging each such k-tuple $\frac{1}{\alpha}$ times its LP-cost compensates for the cluster-cost. Therefore, it suffices to consider the k-tuples $K \in \mathcal{K}_v$.

For any $K \in \mathcal{K}_v$, we have $\frac{1}{k-1} \sum_{u \in K \setminus \{v\}} z_{vu} \leq x_K \leq \sum_{u \in K \setminus \{v\}} z_{vu}$, where the inequalities are based on the LP constraints. By observing that $z_{vu} \leq \alpha$, we have the following

TABLE III

OVERVIEW OF THE DIFFERENT CASES STUDIED IN THE PROOF OF THEOREM III.2. OUTPUT REFERS TO THE OUTPUT OF THE ALGORITHM; THE "COST OF SPLITTING" AND "JOINT CLUSTERING" REFER TO THE COST OF SPLITTING THE k-TUPLE IN K OR PLACING ALL OF K INTO INTO THE OUTPUT CLUSTER, RESPECTIVELY. ADDITIONAL CONDITIONS ARE SPECIFIC TO THE CASE UNDER INVESTIGATION

Output	Cost of	$K \ni v$	Additional conditions	Approx. constant	Case#
$\{v\}$	splitting	yes	$K \cap [S \setminus \mathcal{N}'_\alpha(v)] \neq \emptyset$	$1/\alpha$	1
	splitting	yes	$K \cap [S \setminus \mathcal{N}'_\alpha(v)] = \emptyset$	$1/(\alpha\beta)$	1
$\mathcal{N}'_\alpha(v)$	joint clustering	yes	—	$1/[1-(k-1)\alpha]$	2.1
	joint clustering	no	Given $u \in \mathcal{N}_\alpha(v)$, $z_{vu} \leq \alpha\beta$,	$1/[1-k\alpha\beta]$	2.1.1
	joint clustering	no	Given $u \in \mathcal{N}_\alpha(v)$, $z_{vu} > \alpha\beta$,	$1/[1-(k-1)\alpha-\alpha\beta]$	2.1.2
	splitting	yes	—	$1/\alpha$	2.2
	splitting	no	$\exists u \in K \setminus \mathcal{N}_\alpha(v)$, $z_{vu} \geq (1+\beta)\alpha$	$1/(\alpha\beta)$	2.2.1
	splitting	no	$\forall u \in K \setminus \mathcal{N}_\alpha(v)$, $z_{vu} < (1+\beta)\alpha$	$1/[\alpha(1-(k-1)\beta)]$	2.2.2

bound on the LP cost of K:

$$\begin{aligned}&(1-w_K)(1-x_K)+w_K x_K\\ &\geq (1-w_K)(1-\sum_{u\in K\setminus\{v\}} z_{vu})+w_K\frac{1}{k-1}\sum_{u\in K\setminus\{v\}} z_{vu}\\ &= w_K\left[\left(\frac{k}{k-1}\sum_{u\in K\setminus\{v\}} z_{vu}\right)-1\right]+(1-\sum_{u\in K\setminus\{v\}} z_{vu}).\end{aligned}$$

Since each z_{vu} for $u \in K$ satisfies $z_{vu} \leq \alpha \leq 1/k$, the quantity in square brackets is negative, so that $w_K \leq 1$ implies

$$(1-w_K)(1-x_K)+w_K x_K \geq \frac{1}{k-1}\sum_{u\in K\setminus\{v\}} z_{vu}.$$

Summing over all $K \in \mathcal{K}_v$, we see that

$$\begin{aligned}&\sum_{K\in\mathcal{K}_v}[(1-w_K)(1-x_K)+w_K x_K]\\ &\geq \sum_{K\in\mathcal{K}_v}\sum_{u\in K\setminus\{v\}}\frac{1}{k-1}z_{vu} \geq \alpha\beta\binom{|\mathcal{N}_\alpha(v)|}{k-1},\end{aligned}$$

where the last inequality follows from the condition $\sum_{u\in\mathcal{N}_\alpha(v)} z_{vu} > \beta\alpha\,|\mathcal{N}_\alpha(v)|$ that causes the algorithm to output $\{v\}$ as a singleton cluster.

Therefore, charging $\frac{1}{\alpha\beta}$ times the LP-cost to each k-tuple in $\mathcal{K}_v$ is enough to compensate for the total clustering cost of the tuples in $\mathcal{K}_v$.

Case 2: *The output is the cluster* $\mathcal{N}'_\alpha(v)$.

Case 2.1: First, consider the cost of the k-tuples inside the cluster. If $v \in K$, then we have $x_K \leq \sum_{u\in K\setminus\{v\}} z_{vu} \leq (k-1)\alpha$. As the LP cost is $\geq (1-w_K)(1-x_K)$ and the cluster-cost is $1-w_K$, charging $\frac{1}{1-(k-1)\alpha}$ for this tuple suffices to compensate for the cluster-cost.

If $v \notin K$, order the vertices in $\mathcal{N}_\alpha(v)$ in such a way that for any $u, u' \in \mathcal{N}_\alpha(v)$, $u \prec u'$ iff $z_{vu} < z_{vu'}$ and assign an arbitrary order ($u \prec u'$) when the equality ($z_{vu} = z_{vu'}$) holds.

For each vertex $u \in \mathcal{N}_\alpha(v)$, let $R_u = \{u' \in \mathcal{N}_\alpha(v)\colon u' \preceq u\}$, and let $\mathcal{K}_v^{(u)}$ be the set of cliques $K \in \mathcal{K}_v^{(u)}$ such that l is the largest vertex of K according to $\prec$. Thus, if $K \in \mathcal{K}_v^{(u)}$, then $u \in \mathcal{K}_v^{(u)}$ and $K \subseteq R_u$.

Note that because of the order, we have $\sum_{u'\in R_u} z_{vu'} \leq \alpha\beta|R_u|$. Fix some $u \in \mathcal{N}_\alpha(v)$, and consider the total cost of the k-tuples in $\mathcal{K}_v^{(u)}$. The corresponding cluster-cost is $\sum_{K\in\mathcal{K}_v^{(u)}} 1-w_K$ while the LP cost is $\sum_{K\in\mathcal{K}_v^{(u)}}(1-x_K)(1-w_K)+x_K w_K$.

Let $\mathcal{K}'_v$ be the set of k-tuples $K \subseteq \mathcal{N}'_\alpha(v)$ with $v, u \in K$.

Case 2.1.1: $z_{vu} \leq \beta\alpha$. In this case, for each $K \in \mathcal{K}_v^{(u)}$, we have

$$x_K \leq \sum_{u'\in K} z_{vu'} \leq k z_{vu} \leq k\beta\alpha,$$

so that charging $\frac{1}{1-k\beta\alpha}$ times the LP-cost to each k-tuple in $\mathcal{K}_v^{(u)}$ suffices to pay for the cluster cost of all such tuples.

Case 2.1.2: $z_{vu} > \beta\alpha$. In this case, by using $x_K \leq \sum_{u\in K} z_{vu}$, we have $1-x_K \geq 1-\sum_{u\in K} z_{vu}$. Furthermore,

$$x_K \geq z_{vu} - \min_{u'\in K\setminus\{u\}} z_{vu'} \geq z_{vu} - \frac{1}{k-1}\sum_{u'\in K\setminus\{u\}} z_{vu'}.$$

Letting $\sigma = \sum_{u'\in K\setminus\{u\}} z_{vu'}$ so that $1-x_K \geq 1-z_{vu}-\sigma$, we have the following lower bound on the LP-cost of K:

$$\begin{aligned}&(1-w_K)(1-x_K)+w_K x_K\\ &\geq (1-w_K)(1-z_{uv}-\sigma)+w_K(z_{uv}-\frac{1}{k-1}\sigma)\\ &= (1-w_K)(1-2z_{uv}-\frac{k-2}{k-1}\sigma)+z_{uv}-\frac{1}{k-1}\sigma.\end{aligned}$$

Now, summing over all $K \in \mathcal{K}_v^{(u)}$ and using the inequality $\sum_{K\in\mathcal{K}_v^{(u)}}\frac{1}{k-1}\sum_{u'\in K\setminus\{u\}} z_{vu'} \leq \left|\mathcal{K}_v^{(u)}\right|\beta\alpha$ yields the following lower bound on the total LP-cost of the k-tuples in $\mathcal{K}_v^{(u)}$:

$$\begin{aligned}&\sum_{K\in\mathcal{K}_v^{(u)}}[(1-w_k)(1-x_K)+w_K x_K]\\ &\geq \sum_{K\in\mathcal{K}_v^{(u)}}[(1-w_K)(1-2z_{uv}-\frac{k-2}{k-1}\sigma)+z_{uv}-\beta\alpha]\\ &\geq \sum_{K\in\mathcal{K}_v^{(u)}}[(1-w_K)(1-z_{uv}-\frac{k-2}{k-1}\sigma-\beta\alpha)]\\ &\geq \sum_{K\in\mathcal{K}_v^{(u)}}[(1-w_K)[1-(k-1)\alpha-\beta\alpha]].\end{aligned}$$

Thus, charging each k-tuple in $\mathcal{K}_v^{(u)}$ a factor of $\frac{1}{1-(k-1)\alpha-\beta\alpha}$ times its LP-cost pays for the cluster-cost of all k-tuples in $\mathcal{K}_v^{(u)}$.

Case 2.2: *The cost of splitting k-tuples across clusters.* Again, we refer to such tuples as split tuples. Each split tuple K incurs a cluster-cost of w_K and an LP-cost of $x_K w_K + (1-x_K)(1-w_K)$. First, suppose that K is a split k-tuple with $v \in K$. Since K is split, there is $u' \in K \setminus \mathcal{N}'_\alpha(v)$ and thus we have $x_K \geq z_{vu'} > \alpha$, so charging $\frac{1}{\alpha}$ times the LP cost pays for such K. We still must pay for the split tuples K with $v \notin K$.

Let $S' \subseteq S \setminus \mathcal{N}'_\alpha(v)$ be such that $|S'| \leq k-1$. Furthermore, let $\mathcal{K}_v^{(S')}$ denote the set of split tuples K such that $v \notin K$ and $K \setminus \mathcal{N}'_\alpha(v) = S'$. According to the definition of S', for any split tuple K, there is a corresponding S'. We show that the total cluster-cost of the tuples in $\mathcal{K}_v^{(S')}$ is at most a constant time their total LP-cost.

Case 2.2.1: *There exists a vertex $u \in S'$ such that $z_{vu} \geq (1+\beta)\alpha$.* In this case, for every $K \in \mathcal{K}_v^{(S')}$, we can take some arbitrary $u' \in K \cap \mathcal{N}_\alpha(v)$ and obtain

$$x_K \geq z_{vu} - z_{vu'} \geq \beta\alpha,$$

since $u' \in \mathcal{N}_\alpha(v)$ implies $z_{vu} \leq \alpha$. Thus, in this case, charging $\frac{1}{\alpha\beta}$ times the LP-cost of each tuple in $\mathcal{K}_v^{(S')}$ pays for the cluster-cost of all tuples in $\mathcal{K}_v^{(S')}$.

Case 2.2.2: *For all $u \in S'$, $z_{vu} \leq (1+\beta)\alpha$.* Consider any $K \in \mathcal{K}_v^{(S')}$. Let $\tilde{S} = K \cap \mathcal{N}_\alpha(v)$, and $\sigma'_{S'} = \sum_{u \in S'} z_{vu}$, $\sigma''_{\tilde{S}} = \sum_{u \in \tilde{S}} z_{vu}$. We have the following bounds:

$$\begin{aligned} 1 - x_K &\geq 1 - \sum_{u \in K} z_{vu} = 1 - \Big(\sum_{u \in S'} z_{vu} + \sum_{u \in \tilde{S}} z_{vu}\Big) \\ &= 1 - (\sigma'_{S'} + \sigma''_{\tilde{S}}), \\ x_K &\geq \max_{u,u' \in K} [z_{vu} - z_{vu'}] \geq \max_{u \in S',\ u' \in \tilde{S}} [z_{vu} - z_{vu'}] \\ &\geq \frac{1}{|S'|}\sigma'_{S'} - \frac{1}{\left|\tilde{S}\right|}\sigma''_{\tilde{S}}. \end{aligned}$$

Combining these bounds yields the following lower bound on the LP-cost of K.

$$\begin{aligned} &(1-w_K)(1-x_K) + w_K x_K \\ &\geq (1-w_K)(1-\sigma'_{S'} - \sigma''_{\tilde{S}}) + w_K \left(\frac{\sigma'_{S'}}{|S'|} - \frac{\sigma''_{\tilde{S}}}{\left|\tilde{S}\right|}\right) \qquad (13) \\ &= w_K \left[\frac{|S'|+1}{|S'|}\sigma'_{S'} + \frac{\left|\tilde{S}\right|-1}{\left|\tilde{S}\right|}\sigma''_{\tilde{S}} - 1\right] + 1 - \sigma'_{S'} - \sigma''_{\tilde{S}}. \end{aligned}$$

The map $K \mapsto (K \cap \mathcal{N}_\alpha(v))$ induces a bijection between $\mathcal{K}_v^{(S')}$ and $\binom{\mathcal{N}_\alpha(i)}{k-|S'|}$. By using $z_{vu'} \leq \alpha\beta$ for $u' \in K \cap \mathcal{N}_\alpha(v)$, we have

$$\begin{aligned} \sum_{K \in \mathcal{K}_v^{(S')}} \sigma''_{\tilde{S}} &= \sum_{K \in \mathcal{K}_v^{(S')}} \sum_{u' \in K \cap \mathcal{N}_\alpha(v)} z_{vu'} \qquad (14) \\ &= \frac{k-|S'|}{|\mathcal{N}_\alpha(v)|}\binom{|\mathcal{N}_\alpha(v)|}{(k-|S'|)} \sum_{u' \in \mathcal{N}_\alpha(v)} z_{vu'} \\ &\leq \sum_{K \in \mathcal{K}_v^{(S')}} \alpha\beta\,|K \cap \mathcal{N}_\alpha(v)|\,. \end{aligned}$$

Since $\alpha, \beta \leq 1/k$, we also have

$$\begin{aligned} 1 - \sigma'_K - \alpha\beta\left|\tilde{S}\right| &\geq 1 - \left|S'\right|(1+\beta)\alpha - \left|\tilde{S}\right|\beta\alpha \\ &\geq 1 - (k-1)(1+\beta)\alpha - \beta\alpha \geq 0. \qquad (15) \end{aligned}$$

Therefore, summing inequality (13) over all $K \in \mathcal{K}_v^{(S')}$ gives the following lower bound on the total LP-cost of all tuples in $\mathcal{K}_v^{(S')}$. Denote $Q \triangleq \frac{|S'|+1}{|S'|}\sigma'_{S'} + \frac{\left|\tilde{S}\right|-1}{\left|\tilde{S}\right|}\sigma''_{\tilde{S}} - 1$, and then

$$\begin{aligned} &\sum_{K \in \mathcal{K}_v^{(S')}} [(1-w_K)(1-x_K) + w_K x_K] \\ &\quad\geq \sum_{K \in \mathcal{K}_v^{(S')}} \left(w_K Q + 1 - \sigma'_{S'} - \sigma''_{\tilde{S}}\right) \\ &\quad\geq \sum_{K \in \mathcal{K}_v^{(S')}} \left(w_K Q + 1 - \sigma'_{S'} - \alpha\beta|\tilde{S}|\right) \\ &\quad\geq \sum_{K \in \mathcal{K}_v^{(S')}} w_K \left(Q + 1 - \sigma'_{S'} - \alpha\beta|\tilde{S}|\right) \\ &\quad\geq \sum_{K \in \mathcal{K}_v^{(S')}} w_K \left[\frac{1}{|S'|}\sigma'_{S'} + \frac{\left|\tilde{S}\right|-1}{\left|\tilde{S}\right|}\sigma''_{\tilde{S}} - \alpha\beta|\tilde{S}|\right] \\ &\quad\geq \sum_{K \in \mathcal{K}_v^{(S')}} w_K[\alpha + 0 - (k-1)\alpha\beta], \end{aligned}$$

where the second inequality is due to (14) and the third inequality follows from (15) and $w_K \leq 1$.

Therefore, charging a factor of $\frac{1}{\alpha(1-(k-1)\beta)}$ times the LP-cost for each tuple in $\mathcal{K}_v^{(S')}$ pays for the cluster-cost of all tuples in $\mathcal{K}_v^{(S')}$.

In summary, if $\alpha, \beta \leq 1/k$, then charging each tuple a factor of c times its LP cost, where

$$c = \max\left\{\frac{1}{\beta\alpha}, \frac{1}{1-(k-1)\alpha}, \frac{1}{1-(k-1)\alpha-\beta\alpha}, \frac{1}{\alpha[1-(k-1)\beta]}\right\} = \frac{1}{\alpha\beta},$$

suffices to compensate for the cluster-cost of all tuples.

For the MMCC problem, the proof of Theorem III.2 (Appendix) may be generalized by independently handling tuples of fixed sizes. However, to obtain a tighter approximate constant then the one presented in Theorem III.4, we show next how to modify the corresponding analysis for Case 2.2.2.

The analysis of Case 2.2.2 for mixed motifs proceeds as follows. Define $S^* = \{u \in S \setminus \mathcal{N}'_\alpha(v), z_{vu} \leq (1+\beta)\alpha\}$ and $\bar{\sigma} = \frac{1}{|S^*|}\sum_{u \in S^*} z_{vu} \leq (1+\beta)\alpha$. For $S' \subseteq S^*$ of size $|S'| \leq k-1$, and for all $u \in S'$, it holds that $z_{vu} < (1+\beta)\alpha$.

Let $\mathcal{K}_v^{(S')}$ be the set of all k-tuples K such that $K \setminus \mathcal{N}'_\alpha(v) = S'$ and $v \notin K$. We need to find a constant c such that

$$\begin{aligned} &\sum_{u' \in \mathcal{N}_\alpha(v)} \sum_{u \in S^*} w_{u'u} + \lambda \sum_{S' \subseteq S^*} \sum_{K \in \mathcal{K}_v^{(S')}} w_K \\ &\leq c \left\{ \sum_{u' \in \mathcal{N}_\alpha(v)} \sum_{u \in S^*} [w_{u'u} z_{u'u} + (1-w_{u'u})(1-z_{u'u})] \right. \\ &\left. \quad + \lambda \sum_{S' \subseteq S^*} \sum_{K \in \mathcal{K}_v^{(S')}} [w_K x_K + (1-w_K)(1-x_K)] \right\}. \end{aligned}$$

Recall that $\tilde{S} = K \setminus S$, and that $\sigma'_{S'} = \sum_{u \in S'} z_{vu}$ and $\sigma''_{\tilde{S}} = \sum_{u' \in \tilde{S}} z_{vu'}$. Using the same method as the one outlined in

the derivations of (13) and (14), and observing that $\sigma''_{\tilde{S}} \geq 0$, we obtain

$$\sum_{S' \subseteq S^*} \sum_{K \in \mathcal{K}_v^{(S')}} [w_K x_K + (1 - w_K)(1 - x_K)] \tag{16}$$

$$\geq \sum_{S' \subseteq S^*} \sum_{K \in \mathcal{K}_v^{(S')}} \left[w_K \left(\frac{|S'|+1}{|S'|} \sigma'_{S'} - 1 \right) + 1 - \sigma'_{S'} - \alpha\beta|\tilde{S}| \right] \tag{17}$$

$$\geq \sum_{t=1}^{k-1} \sum_{S' \subseteq S^*, |S'|=t} \left[(1 - \sigma'_{S'} - \alpha\beta|\tilde{S}|)|\mathcal{K}_v^{(S')}| + \left(\frac{t+1}{t} \sigma'_{S'} - 1 \right) \sum_{K \in \mathcal{K}_v^{(S')}} w_K \right], \tag{18}$$

where the sum of the coefficients in front of the term $\alpha\beta$ equals $-\sum_{t=1}^{k-1}(k-t)\binom{|\mathcal{N}_\alpha(v)|}{k-t}\binom{|S^*|}{t}$.

Using the same approach as for the derivations when $k = 2$, we have

$$\sum_{u \in S^*} \sum_{u' \in \mathcal{N}_\alpha(v)} [w_{u'u} z_{u'u} + (1 - w_{u'u})(1 - z_{u'u})] \tag{19}$$

$$\geq \sum_{u \in S^*} [(1 - z_{vu} - \alpha\beta)|\mathcal{N}_\alpha(v)| + (2z_{vu} - 1) \sum_{u' \in \mathcal{N}_\alpha(v)} w_{uu'} \Big], \tag{20}$$

where the sum of the coefficients in front of the term $\alpha\beta$ equals $-|\mathcal{N}_\alpha(v)|\,|S^*|$.

Next, define two constants r and r' based on

$$r = \frac{(k-2)\,|\mathcal{N}_\alpha(v)|\,|S^*|}{|\mathcal{N}_\alpha(v)|\,|S^*| + \lambda \sum_{t=1}^{k-1}(k-t)\binom{|\mathcal{N}_\alpha(v)|}{k-t}\binom{|S^*|}{t}},$$
$$r' = (k-2) - r,$$

so that they satisfy

$$|\mathcal{N}_\alpha(v)||S^*|r' = \lambda \sum_{t=1}^{k-1}(k-t)\binom{|\mathcal{N}_\alpha(v)|}{k-t}\binom{|S^*|}{t} r.$$

By choosing $\alpha \leq \frac{1}{k}$ and $\beta \leq \frac{1}{k-r}$, we can verify that, for $1 \leq t \leq k-1$, any $S' \subseteq S^*$, $|S'| = t$, and $u \in S^*$,

$$1 - \sigma'_{S'} - (k-t-r)\alpha\beta$$
$$\geq 1 - t\alpha(1+\beta) - (k-t-r)\alpha\beta \tag{21}$$
$$\geq 1 - (k-1)\alpha(1+\beta) - (1-r)\alpha\beta \geq 0, \tag{22}$$
$$1 - z_{vu} - (1+r')\alpha\beta \tag{23}$$
$$\geq 1 - \alpha(1+\beta) - (k-1-r)\alpha\beta \geq 0. \tag{24}$$

Combining inequalities (16) and (19) and inserting r and r' into the expressions, we obtain

$$\sum_{u \in S^*} \sum_{u' \in \mathcal{N}_\alpha(v)} [w_{u'u} z_{u'u} + (1 - w_{u'u})(1 - z_{u'u})] + \lambda \sum_{S' \subseteq S^*} \sum_{K \in \mathcal{K}_v^{(S')}} [w_K x_K + (1 - w_K)(1 - x_K)]$$

$$\geq \sum_{u \in S^*} \Big[(1 - z_{vu} - (1+r')\alpha\beta)|\mathcal{N}_\alpha(v)| + (2z_{vu} - 1) \sum_{u' \in \mathcal{N}_\alpha(v)} w_{uu'} \Big] + \lambda \sum_{t=1}^{k-1} \sum_{S' \subseteq S^*, |S'|=t} \Big[(1 - \sigma'_{S'} - (k-t-r)\alpha\beta)|\mathcal{K}_v^{(S')}| + \left(\frac{t+1}{t} \sigma'_{S'} - 1 \right) \sum_{K \in \mathcal{K}_v^{(S')}} w_K \Big]$$

$$\geq \sum_{u \in S^*} \left\{ \left[z_{vu} - (1+r')\alpha\beta \right] \sum_{u' \in \mathcal{N}_\alpha(v)} w_{uu'} \right\} + \lambda \sum_{t=1}^{k-1} \sum_{S' \subseteq S^*, |S'|=t} \left\{ \left[\frac{1}{t} \sigma'_{S'} - (k-t-r)\alpha\beta \right] \sum_{K \in \mathcal{K}_v^{(S')}} w_K \right\}$$
$$\geq \alpha' \sum_{u' \in \mathcal{N}_\alpha(v)} \sum_{u \in S^*} w_{u'u} + \lambda \sum_{S' \subseteq S^*} \sum_{K \in \mathcal{K}_v^{(S')}} w_K,$$

where the second inequality is due to inequalities (22) and (24), and $w_{uu'}, w_K \leq 1$ and $\alpha' \triangleq \min\{\alpha - (1+r')\alpha\beta, \alpha - (k-1-r)\alpha\beta\}$

Therefore, charging a factor of $\min\{\alpha - (1+r')\alpha\beta, \alpha - (k-1-r)\alpha\beta\} = \alpha - (k-1-r)\alpha\beta$ times the LP-cost for all pairs (u, u') such that $u' \in \mathcal{N}_\alpha(v)$ and $u \in S^*$, and for all k-tuples K such that $K \setminus \mathcal{N}_\alpha(v) \subseteq S^*$ compensates for splitting all such pairs and k-tuples during clustering.

Combining all cases described in Table (III) shows that if $\alpha \leq 1/k, \beta \leq 1/(k-r)$, then charging each pair and k-tuple a factor of c times its LP cost, where

$$c = \max\{\frac{1}{\beta\alpha}, \frac{1}{1-(k-1)\alpha}, \frac{1}{1-(k-1)\alpha-\beta\alpha}, \frac{1}{\alpha[1-(k-1-r)\beta]}\} = \frac{1}{\alpha\beta} \geq k(k-r),$$

suffices to compensate for the cluster-cost of all pairs and tuples.

Note that, however, r depends on $|S^*|$ and $\mathcal{N}_\alpha(v)$ and these values are not known a priori and they may change over different iterations. Hence, we need to find a universal lower bound for r. Since $|S^*| + |\mathcal{N}_\alpha(v)| \leq n$, a simple bound of the form may be obtained according to

$$r \geq \frac{(k-2)\,|\mathcal{N}_\alpha(v)|}{|\mathcal{N}_\alpha(v)| + \lambda \sum_{t=1}^{k-1} |\mathcal{N}_\alpha(v)| \binom{|\mathcal{N}_\alpha(v)|-1}{k-t-1}\binom{|S^*|}{t}} \geq \frac{k-2}{1+\lambda n^{k-1}} = r_0.$$

Therefore, if $\alpha \leq 1/k, \beta \leq 1/(k-r_0)$, one can achieve the constant approximation factor $c = 1/\alpha\beta$.

First, there are $\binom{n}{i}$ ways to choose $K_1 \cup K_2$ with size i for $k+1 \leq i \leq 2k-1$. Then, for the set $K_1 \cup K_2$, there are $\binom{i}{k}$ ways to choose K_1. Given $K_1 \cup K_2$ and K_1, there are $\binom{k}{2k-i}$ ways to choose K_2. Since K_1 and K_2 are symmetric, the obtained count should be divided by 2. Then, there are $\binom{i}{k} - 2$ ways to choose K_3 from $K_1 \cup K_2$ while we keep $K_3 \neq K_1$ and $K_3 \neq K_2$. Therefore, we obtain the value of $|\Upsilon|$ given in Equation (3).

REFERENCES

[1] P. Li, H. Dau, G. Puleo, and O. Milenkovic, "Motif clustering and overlapping clustering for social network analysis," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.

[2] N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Mach. Learn.*, vol. 56, nos. 1–3, pp. 89–113, Jun. 2004.

[3] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: Ranking and clustering," *J. ACM*, vol. 55, no. 5, p. 23, Oct. 2008.

[4] M. Charikar, V. Guruswami, and A. Wirth, "Clustering with qualitative information," in *Proc. IEEE 44th Annu. Symp. Found. Comput. Sci.*, Oct. 2003, pp. 524–533.

[5] I. Chien, C.-Y. Lin, and I.-H. Wang, "Community detection in hypergraphs: Optimal statistical limit and efficient algorithms," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2018, pp. 871–879.

[6] P. Li and O. Milenkovic, "Inhomogeneous hypergraph clustering with applications," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 2308–2318.

[7] P. Li and O. Milenkovic, "Submodular hypergraphs: P-Laplacians, cheeger inequalities and spectral clustering," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4026–4034.

[8] C.-Y. Lin, I. E. Chien, and I.-H. Wang, "On the fundamental statistical limit of community detection in random hypergraphs," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2017, pp. 2178–2182.

[9] D. Ghoshdastidar and A. Dukkipati, "Consistency of spectral partitioning of uniform hypergraphs under planted partition model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 397–405.

[10] A. R. Benson, D. F. Gleich, and J. Leskovec, "Tensor spectral clustering for partitioning higher-order network structures," in *Proc. SIAM Int. Conf. Data Mining*, Vancouver, BC, Canada, 2015, pp. 118–126.

[11] G. J. Puleo and O. Milenkovic, "Correlation clustering with constrained cluster sizes and extended weights bounds," *SIAM J. Optim.*, vol. 25, no. 3, pp. 1857–1872, 2015.

[12] G. J. Puleo and O. Milenkovic, "Correlation clustering and biclustering with locally bounded errors," *IEEE Trans. Inf. Theory*, vol. 64, no. 6, pp. 4105–4119, Jun. 2018.

[13] N. Veldt, D. F. Gleich, and A. Wirth, "A correlation clustering framework for community detection," in *Proc. World Wide Web Conf.* Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2018, pp. 439–448.

[14] A. R. Benson, D. F. Gleich, and J. Leskovec, "Higher-order organization of complex networks," *Science*, vol. 353, no. 6295, pp. 163–166, 2016.

[15] D. F. Gleich, N. Veldt, and A. Wirth, "Correlation clustering generalized," 2018, *arXiv:1809.09493*. [Online]. Available: https://arxiv.org/abs/1809.09493

[16] X. Pan, D. Papailiopoulos, S. Oymak, B. Recht, K. Ramchandran, and M. I. Jordan, "Parallel correlation clustering on big graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 82–90.

[17] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropolog. Res.*, vol. 33, no. 4, pp. 452–473, 1977.

[18] M. R. Garey and D. S. Johnson, *Computers and Intractability*, vol. 29. New York, NY, USA: Freeman, 2002.

Pan Li is a postdoctoral research fellow in the Department of Computer Science at Stanford University and will join the Department of Computer Science at Purdue University as an assistant professor in Fall, 2020. He earned his master's degree from Tsinghua University in 2015 and his PhD from the University of Illinois at Urbana-Champaign in 2019.

Gregory J. Puleo is an assistant professor in the Department of Mathematics and Statistics at Auburn University. He earned his PhD in 2014 from the University of Illinois at Urbana-Champaign.

Olgica Milenkovic is a professor of Electrical and Computer Engineering at the University of Illinois, Urbana-Champaign (UIUC), and Research Professor at the Coordinated Science Laboratory. She obtained her Masters Degree in Mathematics in 2001 and PhD in Electrical Engineering in 2002, both from the University of Michigan, Ann Arbor. Prof. Milenkovic is heading a group focused on addressing unique interdisciplinary research challenges spanning the areas of algorithm design and computing, bioinformatics, coding theory, machine learning and signal processing. Her scholarly contributions have been recognized by multiple awards, including the NSF Faculty Early Career Development (CAREER) Award, the DARPA Young Faculty Award, the Dean's Excellence in Research Award, and several best paper awards. In 2013, she was elected a UIUC Center for Advanced Study Associate and Willett Scholar while in 2015 she became a Distinguished Lecturer of the Information Theory Society. She is an IEEE Fellow and has served as Associate Editor of the IEEE TRANSACTIONS OF COMMUNICATIONS, the IEEE TRANSACTIONS ON SIGNAL PROCESSING, the IEEE TRANSACTIONS ON INFORMATION THEORY, and the IEEE TRANSACTIONS ON MOLECULAR, BIOLOGICAL AND MULTI-SCALE COMMUNICATIONS. In 2009, she was the Guest Editor-in-Chief of a special issue of the IEEE TRANSACTIONS ON INFORMATION THEORY ON MOLECULAR BIOLOGY AND NEUROSCIENCE, while in 2019 she served as Guest Editor-in-Chief of a special dedicated to the interdisciplinary work of V.I. Levenshtein.