A Branch-and-Cut Algorithm for Constrained Graph Clustering

Behrouz Babaki[1], Dries Van Daele[1], Bram Weytjens[2,3,4,5] Tias Guns[1,6]
[1]Department of Computer Science, KU Leuven, Belgium
[2]Department of Microbial and Molecular Systems, KU Leuven, Belgium
[3]Department of Plant Biotechnology and Bioinformatics, UGent, Belgium
[4]Bioinformatics Institute Ghent, Belgium
[5]Department of Information Technology, IDLab, UGent, Belgium
[6]Department of Business Technology and Operations, VUB, Belgium

Abstract—**Clustering is a well-defined problem class in data mining, and many variations of it exists. However, practitioners often have additional constraints or quality scores that are not supported by standard algorithms. This has lead to the study of constrained clustering, which investigates generic methods to handle a variety of constraints and objectives. In our bioinformatics work, we were faced with exactly such a problem: a graph clustering problem with strict connectivity requirements and an objective function based on penalties rather than density of the clustering. In this paper, we explain the problem including the constraints and quality measures, and propose to use generic Mixed Integer Programming to solve it. We propose two approaches to handle the connectivity requirement in particular: one defined over all simple paths in the graph explicitly, and one based on cutting planes that enforce connectivity only when needed. Our experiments show that these approaches are able to solve the problem well. We hence demonstrate the applicability of generic OR methods on this application-driven data mining problem.**

I. Introduction

Clustering is the task of partitioning a set of entities into homogeneous subsets. The quality of the clustering is typically determined by the *distance* between the entities. In graph clustering [1], each entity is assumed to be a node in a graph; this graph is typically not fully connected. The quality is then determined by the *density* of the entities within a cluster or by the *cut size* between the clusters, that is the number of edges shared between different clusters. The latter is for example frequently studied in the field of graph partitioning [2]. Graph clustering has applications in many domains including social network analysis and community detection, information networks, transportation and logistics, and bioinformatics [1].

As data mining is increasingly applied on more and more problems in different domains, it increasingly happens that existing clustering methods are not suited for the problem at hand. This is either because the problem domain imposes additional constraints that can not be expressed in these methods, or because the objective function has a non-standard form. This has lead to the field of *constrained clustering*, which studies clustering problems involving different constraints and objectives [3]. An increasingly popular way to handle a broad range of constraints and objectives is to use generic optimisation tools. In other words, to cast the problem as an optimisation problem and to use generic (discrete) optimisation solvers such as constraint programming [4], mixed integer programming [5], [6] or maximum satisfiability solvers [7]. Though scalability can be an issue, these solvers can intrinsically handle different objectives and constraints.

The problem we study in this paper is such a graph clustering problem that does not fit existing approaches. It is part of a bigger pipeline in computational cancer research, where the goal is to find *pathways* in gene interaction networks. There is hence an *interaction network* and it is a hard constraint that all nodes belonging to a cluster must be connected in this interaction network. To evaluate the quality there is a separate weighted *co-occurence* network and the nodes belonging to a cluster should have a low co-occurence penalty. Furthermore, small pathways are biologically less meaningful and hence the size of the clusters should also be maximized. The problem is hence a bi-objective graph clustering problem with hard connectivity constraints on a separate network. Existing methods are not able to handle such a complex setting, hence we study a mixed integer programming approach.

More specifically, our contributions are as follows: 1) we identify a new bi-objective constrained graph clustering with applications in bio-informatics and present an MIP formulation of the problem; 2) in order to better handle the large number of connectivity constraints, we propose a branch-and-cut approach that adds connectivity constraints as needed using the principle of node-cut sets. Our experiments demonstrate the effectiveness of the approach.

The rest of this paper is structured as follows: we first discuss related work. In section III we present the application that motivates this problem. Section IV introduces the formal problem definition and a MIP formulation with the connectivity requirement. In section VI, we introduce two methods for handling the connectivity requirement, including a cutting plane approach. Section VII contains the experiments after which we conclude.

II. Related work

There are several studies that apply mathematical programming to clustering and graph partitioning problems [8]. In addition to integer linear programming, semidefinite programming [9], [10], and quadratic programming [11] formulations have also been used for solving these problems. Techniques such as branch-and-cut [12], [13] and branch-and-price [14], [15] have been used to improve the performance.

There are several variants of the graph partitioning problem. When the underlying graph is a complete graph, this problem is sometimes called the *clique partitioning* problem. In most variants of the graph partitioning problem, the edges, nodes, or both are weighted. The number of clusters can be specified by the user or they can be decided by the algorithm.The two most prominent types of objective functions are 1) the total weight of the edges that have endpoints in different clusters and 2) the total weight of the edges that have endpoints in the same cluster. Depending on the meaning of the weights, these function are minimized or maximized. In either case, these objectives are meant to increase the homogeneity of the clusters.

Several types of constraints are common to the graph partitioning problem. The most widely used constraint is the *balance* constraint that requires the number of nodes in all clusters to be almost equal [16]. Other types of constraints include constraints on the size of clusters [11], and constraints on the total weight of nodes in a cluster [12].

The main difference of our problem with existing ones is that we have two graphs, where the edges of the co-occurrence graph are weighted, but the goal is to minimize their total value. The connectivity of the nodes in the interaction network are required and must hence be added as hard constraints. The graph partitioning problem of [17] also requires such constraints. However, they do not assume that the number of clusters is given, and their encoding of the clustering problem is quite different from ours.

III. Motivating application

In cancer research, tumor tissue is collected from patients for further study. Such a tumor is basically a cell in which one or more genes have mutated. Normally that cell would be destroyed by the immune system, but in case of a tumor that cell has managed to survive and may even be growing (out of control). Genes and mutated genes can be identified in a tumor by sequencing the DNA of the tissue. Nowadays, this sequencing has become fairly commonplace, enabling the genome-wide measurement of mutated genes across large groups of cancer patients.

The key challenges when interpreting these data are to detect the (mutated) genes that affect the creation and development of cancer and to gain an understanding of their interaction. Initially, the main focus by the community was solely on the detection of key driver genes. However, there are typically many mutated genes making it challenging to detect rare ones with high statistical significance. For this reason, there is recently a rise in methods aiming to exploit the information contained in the human interaction network [18], [19]. This network expresses which genes interact with each other. This can be used to verify that a set of genes interact with each other.

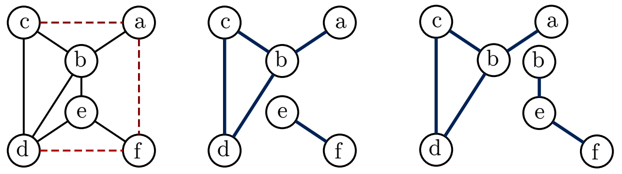

Fig. 1: (left) A small subgraph extracted from the interaction network. Dashed lines represent the can-not-link constraints. (middle) the pathways obtained from non-overlapping clusters (right) the pathways obtained from overlapping clusters.

In our setting we start by considering a subgraph of the human interaction network containing only those genes and interactions that were identified as being highly relevant to breast cancer. Each node represents a gene or gene product, and each edge represents an interaction between a pair of nodes. The graph may consist of multiple connected components. It is our goal to separate distinct *pathways* from each of these components, where a pathway is a set of genes that interact with each other.

By incorporating such biological pathway information, a superior selection and understanding of genes and their interactions is possible. A difficult challenge however is how to determine that two genes are more likely to belong to the same or to a different pathway. Fortunately, it is known that the creation and growth of tumors follows a clonal *evolutionary* model. Following this model, it is considered unlikely that a tumor would disrupt and mutate different genes within a single pathway. Indeed, it is sufficient to disrupt one gene to disrupt the pathway, and hence evolutionarily there is little incentive to disrupt others as well. As a consequence, it is less common that multiple mutated genes of a single pathway are observed within the same patient. We can hence compute a co-occurrence penalty score for each pair of genes, based on the harm associated with the mutations and the number of patients for whom that pair was observed.

While obtaining pathways with a low penalty is important, one should not go as far as dividing the genes into small clusters. For this reason, we will aim to balance the size of the (smallest) pathway with the amount of penalty the pathways incur.

IV. Problem and MIP formulation

From a computational point of view, the input to our problem are a set of genes and two graphs over those genes: an unweighted interaction graph and a weighted co-occurrence graph. We first review some basic graph concepts before formally defining the problem.

A graph $G = (V, E)$ consists of a set of nodes G and a set of edges E. Each edge $e \in E$ is a tuple $(u, v), u, v \in V$ and the graph is *undirected* if the ordering of the edges does not matter. The graph is *weighted* if each edge $e \in E$ has a corresponding weight $w(e)$. A graph is *simple* if it does not have any loops or double edges. A graph is *connected* if there is a path between each pair of its nodes. The nodes on a path except the first and last ones are called the *intermediate nodes*. A *simple path* is a path with no cycles. Another important concept is that of an induced subgraph. Given a set of nodes $V' \subseteq V$, the *induced subgraph* of G on V' is a graph $G[V']$ that contains only the nodes in V' and only the edges of G that have both endpoints in V'.

1) Problem definition: Our problem is as follows: given a set of genes V, a simple undirected graph $G_i = (V, E_i)$ representing the interaction network and a simple weighted undirected graph $G_o = (V, E_o)$ with weight function w_o representing the co-occurence network. The goal is to partition V into k different groups $V_1, \ldots, V_k$, where k is assumed given. Each cluster induces a subgraph on the interaction graph G_i and that subgraph *must be connected*. The quality of a clustering is determined by the size and co-occurence penalty of the clusters. The two are linearly combined using γ to obtain the following objective that must be maximized:

$$f(V_1, \ldots, V_k) = min_{c=1}^{k}|V_c| - \gamma \sum_{c=1}^{k} \sum_{e \in G_o[V_c]} w_o(e) \quad (1)$$

The first component represents the size of the smallest cluster while the second component represents the sum of weighted edges in the induced subgraph of the co-occurence graph G_o on the cluster V_c. γ can be used to balance the size component to the co-occurence penalty computed and is assumed given.

2) Mixed integer programming: We can formulate this problem as an integer linear program. The main decision variables in this formulation are those that determine the assignment of nodes to the clusters. We will use binary variables x_{ic} indicates whether or not node i is included in cluster c. Each cluster must be assigned to exactly one cluster. This can be enforced by the following set of constraints:

$$\sum_{c=1}^{k} x_{ic} = 1 \quad \forall i \in \{1, \ldots, |V|\} \quad (2)$$

To model the first component of the objective in Eq. 1 we introduce an integer variable s which represents the size of the smallest cluster. The domain of s is $\{0, \ldots, \lfloor\frac{|V|}{k}\rfloor\}$. The following constraints ensure that s is smaller or equal than the size of each cluster.

$$s \leq \sum_{i=1}^{|V|} x_{ic} \quad \forall c \in \{1, \ldots, k\} \quad (3)$$

As s is included in the objective function it will be maximized and hence take the value of the size of the smallest cluster during optimisation.

For the second component, we introduce additional variables to model the edges in the induced subgraph $G_o[V_c]$ of each cluster. More specifically, we introduce binary variables y_{ijc} which are equal to one if and only if nodes i and j are both included in cluster V_c. To enforce this property, we add the following constraints to the model:

$$y_{ijc} \geq x_{ic} + x_{jc} - 1 \quad \forall c \in \{1, \ldots, k\}, \forall (i,j) \in E_o \quad (4)$$

When both x_{ic} and x_{jc} are equal to one, this constraint forces y_{ijc} to be equal to one. Otherwise, y_{ijc} can be either zero or one. However, as we will see, y_{ijc} is included in the objective function with a negative coefficient. Hence the optimization procedure will automatically fix y_{ijc} to zero in such cases.

Our formulation so far does not ensure that the nodes in each cluster are connected. For now assume that constraint $\mathsf{connected}(x_{1c}, \ldots, x_{|V|c})$ enforces the connectivity of cluster c. We will discuss the exact formulation of this constraint in the next section.

The complete model that we defined is hence the following, where $w_o((i,j)) = w_o(e)$ for $e = (i,j)$:

$$\text{maximize} \quad s - \gamma \sum_{c=1}^{k} \sum_{(i,j) \in E_o} w_o((i,j)) * y_{ijc} \quad (5)$$

s.t.

$$\sum_{c=1}^{k} x_{ic} = 1 \quad \forall i \in \{1, \ldots, |V|\} \quad (6)$$

$$y_{ijc} \geq x_{ic} + x_{jc} - 1 \quad \forall c \in \{1, \ldots, k\}, \forall (i,j) \in E_o \quad (7)$$

$$s \leq \sum_{i=1}^{|V|} x_{ic} \quad \forall c \in \{1, \ldots, k\} \quad (8)$$

$$\mathsf{connected}(x_{1c}, \ldots, x_{|V|c}) \quad \forall c \in \{1, \ldots, k\} \quad (9)$$

$$x_{ic} \in \{0, 1\} \quad \forall c \in \{1, \ldots, k\}, \forall i \in \{1, \ldots, |V|\} \quad (10)$$

$$y_{ijc} \in \{0, 1\} \quad \forall c \in \{1, \ldots, k\}, \forall (i,j) \in E_o \quad (11)$$

$$s \in \{0, \ldots, \lfloor\frac{|V|}{k}\rfloor\} \quad (12)$$

V. Extensions and improvements

We choose to use a generic discrete constraint solver to solver our non-traditional clustering problem. In the following we discuss three extensions to the above formulation that are possible thanks to the use of generic solvers and there ability to handle different types of constraints.

A. Overlapping clusters

It is known that genes can occur in multiple pathways, and hence we may wish to allow for nodes to be included in more than one cluster. To apply this modification to our formulation, we only need to replace constraints (2) with the following inequalities:

$$\sum_{j=1}^{k} x_{ij} \geq 1 \quad i \in \{1, \ldots, |V|\} \quad (13)$$

B. Breaking symmetries

Clustering problems often have an inherent symmetry which is due to the fact that the labels of clusters are arbitrary. This means that for each solution, there are $k!$ equivalent solutions that only differ by the cluster labels. We can strengthen our formulation by breaking these symmetries. [20] suggests two measures to reduce these symmetries: 1) assign label 1 to the cluster that contains node 1. 2) assign labels to other clusters

Algorithm 1 Computing the set of Pareto optimal solutions

```
1: $\mathcal{P} \leftarrow \emptyset$                    ▷ the set of Pareto optimal solutions
2: $m \leftarrow 0$                    ▷ Minimum size of the previous solution
3: repeat
4:     $solution \leftarrow$ MINIMIZEPENALTIES$(V, G_i, G_c, k, m)$
5:     $\mathcal{P} \leftarrow \mathcal{P} \cup solution$
6:     $m \leftarrow$ size of the smallest cluster in $solution$
7: until no $solution$ was found
8: return $\mathcal{P}$
```

in the increasing order of their sizes. This translates to the following constraints:

$$x_{11} = 1 \tag{14}$$

$$\sum_{i=1}^{|V|} x_{ic} \leq \sum_{i=1}^{|V|} x_{i(c+1)} \qquad c \in \{2, \ldots, k-1\} \tag{15}$$

These constraints do not eliminate all symmetries (especially in the case of overlapping clusters) but still lead to improvements in performance in practice.

C. Obtaining the set of Pareto optimal solutions

In the above formulation we used the commonly employed method of reducing a bi-objective optimisation problem to a single-objective one through the use of a balancing parameter γ. An alternative solution is to use a bi-objective optimisation approach to compute the set of *Pareto optimal* solutions. A solution of a bi-objective optimization problem is Pareto optimal if there is no other solution with a better quality with respect to both objectives. Obtaining the set of Pareto optimal solutions for other types of bi-objective constrained clustering has been studied before [21]. As in that work, we use a method based on the ϵ-constraint algorithm [22] to obtain the Pareto optimal solutions. To do this, we remove the size component from the objective leaving only the co-occurence part. Then we iteratively solve this modified problem, each time adding a constraint such that the size of the smallest cluster is larger than that found in the previous iteration (m):

$$\sum_{i=1}^{|V|} x_{ic} > m \quad \forall c \in \{1, \ldots, k\} \tag{16}$$

The approach is depicted in Algorithm 1.

VI. Enforcing connectivity

The remaining issue to work out is how to represent the $\mathsf{connected}(x_{1j}, \ldots, x_{|V|j})$ constraint. We investigate two different approaches.

For a cluster to be connected, there should exist for each pair of points belonging to the cluster at least one path between these two nodes such that all nodes on this path also belong to the cluster. In section VI-A we enforce this condition by explicitly enumerating all simple paths between each pair of non-adjacent nodes in the graph and adding variables and constraints for these paths. However, the total number of paths can be exponential leading to very large models to solve.

A way to avoid having to ground out constraints for all the possible paths is to incrementally add only those constraints needed. The *cutting plane* algorithm is a method that allows this exactly. In section VI-B we introduce another formulation for the connectivity constraint based on node-cut sets, which also has a worst-case exponential number of constraints but which lends itself well to an incremental cutting plane method.

A. Enumerating all simple paths

Consider a simple path between the nodes u and v. Let I denote the set of indices of the intermediate nodes on this path. Assume that binary variable y_c indicates that all these nodes belong to cluster c. A standard translation of the relation $(y_c = 1) \Leftrightarrow \wedge_{i \in I}(x_{ic} = 1)$ to linear constraints gives the following inequality:

$$0 \leq \sum_{i \in I} x_{ic} - |I| y_c \leq |I| - 1$$

In general, let P_{uv} denote the set of all simple paths between nodes u and v in the interaction graph. For a path $P_r \in P_{uv}$, let I_r denote the set of indices of intermediate nodes of P_r. We introduce binary variable y_{rc} to indicate that all these nodes are assigned to cluster c. This relationship is enforced by the following constraints:

$$0 \leq \sum_{i \in I_r} x_{ic} - |I_r| y_{rc} \leq |I_r| - 1 \qquad \forall u, v \in V, (u,v) \notin E,$$
$$\forall r \in \{1, \ldots, |P_{uv}|\}, \forall c \in \{1, \ldots, k\} \tag{17}$$

Finally, to enforce the condition $(x_{uc} = 1 \wedge x_{vc} = 1) \Rightarrow \vee_r (y_{rc} = 1)$, we add the following constraints to the model:

$$x_{uc} + x_{vc} - 1 \leq \sum_{r=1}^{|P_{uv}|} y_{rc} \qquad \forall u, v \in V, (u,v) \notin E,$$
$$\forall c \in \{1, \ldots, k\} \tag{18}$$

These constraints ensure there exists at least one path in the interaction graph between every two nodes in the same cluster.

B. A cutting plane approach

In the cutting plane method two steps are iteratively repeated: 1) A model that includes only a subset of the constraints is solved. 2) A constraint that is violated by the current solution, a cut, is added to the model. These steps are repeated until no constraint is violated. To use the cutting plane algorithm, we need an oracle that given an assignment $\mathbf{x}$ can check if $\mathbf{x}$ satisfies all constraints and if not, finds a constraint that is violated by $\mathbf{x}$. Since in the latter case the added constraint separates $\mathbf{x}$ from the feasible region, the problem solved by the oracle is called the *separation* problem. In a *branch and cut* algorithm, cutting planes are added throughout the branch and bound tree.

A cutting plan approach with the formulation of connectivity from the previous section would require us to add both variables (the y_{rc} ones) and constraints (17) and (18) for each cut. Instead, we adopt the approach of [23] which defines the connectivity constraints in terms of *node-cut sets*. The

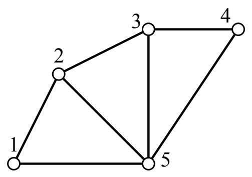

Fig. 2: The sets {2, 5} and {3, 5} belong to $\Gamma(1, 4)$, but the set {2, 3, 5} does not because it is not minimal.

advantage is that no extra variables need to be introduced, and that between two nodes the connectivity can be broken incrementally with individual constraints. The following definition and theorem are taken from [23].

Definition 1 (Node-cut set)**.** *Given nodes $u, v \in V$ that are not adjacent ($(u, v) \notin E$), a set of nodes $S \subseteq V \setminus \{u, v\}$ is a* node-cut set *separating u and v (or simply a uv*-node cut*) if all paths between u and v intersect S.*

There is hence at least one node from every path between u and v in the uv-node cut. A uv-node cut is *minimal* if it is not a uv-node cut after removing any of its nodes. For a pair of non-adjacent nodes u and v, we denote by $\Gamma(u, v)$ the set of all minimal uv-node cut sets. Figure 2 shows examples of minimal node-cut sets. The following theorem relates the connectivity of a graph to its minimal node-cut sets.

Theorem 1. *Given $U \subseteq V$ and a pair of non-adjacent nodes $u, v \in U$, there exists a path between u and v in the subgraph induced by U if and only if all uv-node cuts S are such that $S \cap U \neq \emptyset$.*

For every uv-node cut S this theorem states thatto ensure that cluster c is connected, at least one node in the node cut must belong to the cluster:

$$(x_{uc} = 1 \land x_{vc} = 1) \Rightarrow \lor_{w \in S}(x_{wc} = 1)$$

By translating this condition into linear constraints we can formulate the constraint $\mathsf{connected}(x_{1c}, \ldots, x_{|V|c})$ as follows:

$$\sum_{w \in S} x_{wc} \geq x_{uc} + x_{vc} - 1 \qquad \forall (u, v) \in V, (u, v) \notin E, S \in \Gamma(u, v) \quad (19)$$

The constraint set (19) contains an exponential number of constraints. Following the cutting plane method, we will iteratively add some of the violated constraints to the model. A common practice is to add one or some of the constraints *most violated* by the current solution in each iteration.

Given a solution $\mathbf{x}^*$ of the problem, let us denote the value of x_{ic} variables in this solution by x^*_{ic} and the vector of variables corresponding to cluster c by $\mathbf{x}^*_c$. In this respect, a first observation is that $\sum_{w \in S} x^*_{wc} \geq 0$ is always true as the x^* are Boolean variables. Hence, the constraint can only be violated if $x^*_{uc} + x^*_{vc} - 1 > 0$, that is, if both variables belong to the same cluster. If that is the case, then the constraint can only be violated if $\sum_{w \in S} x^*_{wc} = 0$, that is, if none of the nodes in the cut set are in the cluster. If no such constraint can be found that the connectivity constraint is satisfied.

The same principle can be used on real-valued solutions (as computed by the MIP solver when solving the linear relaxation). The most violated constraint for a non-adjacent pair (u, v) is now the constraint for which $x^*_{uc} + x^*_{vc} - 1 > 0$ and that minimizes $\sum_{w \in S} x^*_{wc}$. To add a cut of (u, v) in the cutting plane algorithm, the goal is hence to find the node-cut set S^*,

$$S^* = \operatorname*{argmin}_{S \in \Gamma(u,v)} \sum_{w \in S} x^*_{wc} \quad (20)$$

It is shown in [23] that the solution for equation 20 can be computed efficiently: If we use x^*_{ic} as the capacity of node i, the separation problem reduces to finding the minimum capacity node cut separating u and v. This problem can be solved using any standard *min-cut* algorithm. A summary of the cut generation procedure is presented in algorithm 2, where for each cluster the most violated constraint among its node-pairs is added.

Algorithm 2 The cut-generation procedure

```
1: $\mathcal{C} \leftarrow \emptyset$                          ▷ Set of constraints to add
2: for $c \in \{1, \ldots, k\}$ do
3:     for $u, v$ such that $(u, v) \notin E_i$ and $x^*_{uc} + x^*_{vc} > 1$ do
4:         $S^* \leftarrow$ min-cut$(u, v, \mathbf{x}^*_c, G_i)$
5:         $\mathcal{C} \leftarrow \mathcal{C} \cup \{\sum_{w \in S^*} x_{wc} \geq x_{uc} + x_{vc} - 1\}$
6:     end for
7: end for
8: Add constraints $\mathcal{C}$ to the model
```

VII. Experiments

We ran experiments on quad-core Linux machines with 32 GB of memory. We implemented our branch-and-cut algorithm using the Python interface of *Gurobi-7*[1] and allowed it to use all 4 cores. The code and data are available online [2].

To solve the separation problem, we used the min-cut/max-flow algorithm from the *NetworkX-1.11* library [24]. In order to use this algorithm, we first replaced each undirected edge by two opposite directed edges of infinite capacity. Then, we replaced each node v by two nodes $v_{\text{in}}, v_{\text{out}}$ connected by two opposite edges, with capacities equal to the capacity of v. We obtain the minimum node cut separating u and v by computing the minimum cut between u_{out} and v_{in} in this graph.

We followed the recipe of [23] for adding the cuts: cuts to the linear relaxation were only added at the root node of the branch-and-cut tree. Moreover, when adding cuts to the relaxation, we monitored the change in the value of the objective function. If this value improved less than 5% in 10 consecutive rounds, we stopped adding cuts. Outside the root node, we only add cuts when an integer solution is found. We normalized the two components of the objective function. We multiplied the size variable by $1/n$ in overlapping clustering and by k/n in non-overlapping clustering. We divided the second component by the sum of edge weights.

The graphs used in our experiments were extracted from the HINT+HI2012 protein-protein interaction network [25],

[1]www.gurobi.com
[2]https://github.com/Behrouz-Babaki/graph_clustering

TABLE I: instance properties

instance	#simple paths	#nodes	#edges
200_1000	93	40	46
200_1250	1508	52	68
250_750	45	30	32
250_1000	1040	69	81
300_750	59	44	46
350_500	12	13	12
350_750	73	58	60
450_750	105	90	92

[26]. We consider this selection of subgraphs representative of the graphs encountered in our application domain. Table I highlights some of the properties of our selected instances. Several of these instances contain a very low number of edges compared to nodes. As a direct result of this, they contain a very small number of paths. We have also selected a few graphs that contain more paths. These graphs also have a number high-degree nodes. Such hubs are fairly common in scale-free networks, of which protein-protein interaction networks are a common example.

A. Results and discussion

1) Scalability of the two approaches: We compare the runtime of the model formulation using all paths (ENUM) with that of using branch-and-cut (BNC), for a number of datasets and k values and averaged over $\gamma \in \{0.1, 0.25, 0.33, 0.5, 1, 2, 3, 4, 5, 10\}$. Table II shows the results. In each instance, one of the algorithms outperforms the other one. This divides the instances into two groups. Instances for which enumerating all paths has a better performance are those which have a similar number of nodes and edges (see table I). As a result, the total number of simple paths in these graphs is small and the optimization model has a reasonable size. On the other hand, for two of the instances the branch-and-cut method provides a clear advantage. The total number of simple paths for these instances is considerably larger than the others. Hence, for these instances the extra effort for solving the separation problem pays off.

2) Impact of γ: The γ parameter is a way of balancing the minimum size of the clusters with the total co-occurrence penalty. We showcase the effect of the parameter on these two components of the objective in Figure 3 for instance 250_750. As γ approaches 0 most weight is given to the size leading to increasing larger minimum clusters but also a sharp incline in the total co-occurence penalty. For too high γ values, ≥ 4 in this case, the size of the clusters can drop to values of 1 or 2, which are not meaningful. In this case, a γ between 1 and 3 seems most sensible.

3) Pareto optimal set: As explained in Section V-C one can also use generic solvers to compute the Pareto-optimal set directly instead of having to determine a γ parameter. In Figure 4 for the non-overlapping setting we can see that a few smallest cluster values are skipped because not optimal, and that the there is a gradual increase in total violations as the minimum size increases, with an increase in incline near the end (there are 30 genes in this dataset). This can be used to select an appropriate solution among the Pareto optimal ones.

TABLE II: Average runtimes of overlapping and non-overlapping clustering by enumerating all simple paths (AllPaths) and branch and cut (BnC). Timed-out experiments are counted as 600 seconds (–).

		overlapping		*non-overlapping*	
instance	*k*	*AllPaths*	*BnC*	*AllPaths*	*BnC*
200_1000	2	**0.830**	12.320	**0.319**	13.398
	3	**3.211**	16.255	**4.474**	15.007
	4	**14.846**	28.714	**15.578**	52.262
200_1250	2	225.735	**43.740**	113.210	**38.849**
	3	582.541	**52.950**	298.390	**72.643**
	4	–	**188.287**	595.930	**205.454**
250_1000	2	487.222	**78.243**	370.898	**69.481**
	3	587.700	**121.677**	–	**217.079**
	4	–	**206.952**	–	–
250_750	2	**0.166**	4.058	**0.025**	3.873
	3	**0.321**	6.433	**0.387**	3.965
	4	**0.737**	10.564	**0.698**	7.374
300_750	2	**0.333**	12.232	**0.209**	10.546
	3	**0.761**	20.365	**0.812**	16.924
	4	**1.728**	28.267	**3.598**	50.836
350_500	2	**0.004**	0.093	**0.003**	0.151
	3	**0.006**	0.129	**0.021**	0.257
	4	**0.007**	0.181	**0.039**	0.211
350_750	2	**0.503**	44.842	**0.591**	20.130
	3	**1.971**	113.311	**2.016**	58.417
	4	**7.892**	323.188	**9.841**	309.998
450_750	2	**1.560**	115.541	**1.334**	89.672
	3	**10.686**	361.855	**12.655**	320.706
	4	**90.516**	430.954	**75.395**	–

4) Biological validation: Research increasingly shows that a single type of cancer, for example breast cancer, is not one homogeneous disease. Instead, it can be divided in different *subtypes* that display different harmful effects, and in turn require different treatment. In order to validate the results of the clustering approach, we looked at the PAM50 tumor subtype classification [27], which was previously published for the patients in our records by the Cancer Genome Atlas Network [28]. PAM50 subtypes are determined by looking at the expression levels of 50 specific genes from a breast cancer sample in order to assign an *intrinsic* subtype to the patients tumor. As this is based on expression data rather than the mutational gene data used in our clustering approach, correlation between the PAM50 subtypes and the identified clusters would be interesting as this would enable us to (partially) subtype breast cancer tumors based on gene mutation data. Furthermore, since the PAM50 tumor subtype classification has prognostic significance, correlation between the clusters

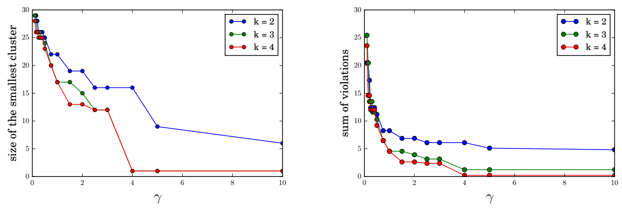

Fig. 3: impact of γ on smallest cluster size (left) and total co-occurrence penalty (right) for instance 250_750.

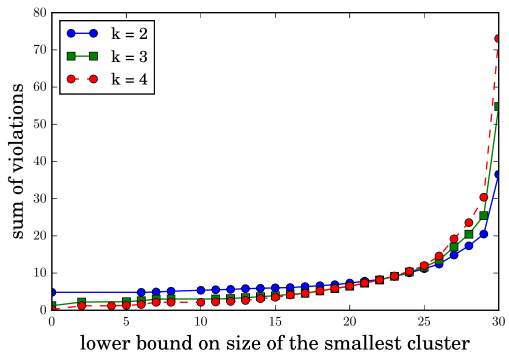

Fig. 4: The Pareto optimal set for overlapping clustering on instance `250_750` with three values for number of clusters.

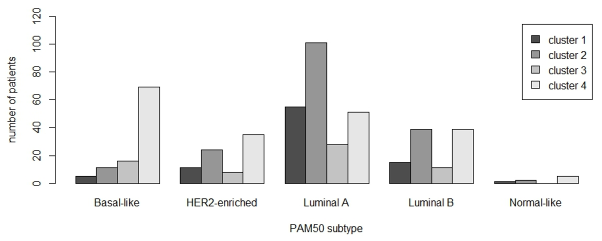

Fig. 5: Number of patients which are a member of each cluster, per PAM50 subtype.

and the PAM50 tumor subtypes would largely validate our clustering approach as being biologically relevant in a real-world cancer setting. We analyzed the data by partitioning it into 4 non-overlapping clusters with a γ of 0.5. We then assessed every patient in our dataset for membership of one or multiple clusters. A patient was considered a member of a specific cluster when that patient had at least one deleterious mutation (defined as a PHRED-scaled CADD score of at least 20 [29]). The results of this analysis are depicted in Figure 5 and Figure 6. We performed a χ-square goodness of fit test on the distribution of the basal-like PAM50 subtype from Figure 5 and on both the distributions of cluster 1 and cluster 2 from Figure 6 to test whether they deviated from a random assignment of patients to clusters. All three distributions deviated significantly from the random case (Basal-like: $\chi-squared = 103.48$, $df = 3$, $p-value < 2.2e-16$; cluster 1: $\chi - squared = 15.507$, $df = 4$, $p - value = 0.003757$; Cluster 2: $\chi - squared = 22.388$, $df = 4$, $p - value = 0.0001678$). Based on these results, and the observations from figure 5 and figure 6, two interesting conclusions could be drawn: 1) patients with a basal-like PAM50 tumor subtype were very likely to have a mutation in clustered pathway 4 and 2) patients with a mutation in pathway 1 or 2 were more likely to have a luminal A subtype. This shows that the identified gene clusters / pathways have at least some correlation with the PAM50 subtypes and could thus be useful in patient subtyping and exploring subsequent treatment options, although more research would be needed to confirm this. As such, our subtyping method is able to generate meaningful biological results in a cancer subtyping setting.

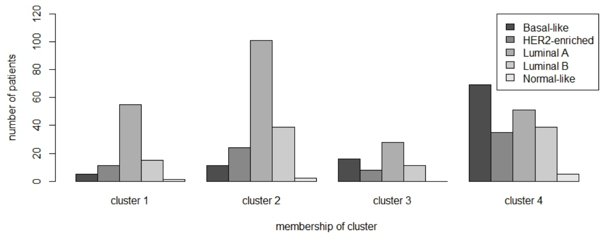

Fig. 6: Number of patients with specific PAM50 subtype, per cluster.

VIII. Conclusions and Future Work

Motivated by a problem in bio-informatics, we presented a novel graph clustering problem involving two graphs, a co-occurrence graph whose weighted edges are part of the objective function, and an interaction graph with hard connectivity constraints. We propose two methods for handling the (potentially exponential in number) connectivity constraints, one based on enumerating all simple paths and the other being a cutting plane approach. We also present a number of extensions such as a bi-objective Pareto optimisation method to balance minimum cluster size and total penalty. Computational experiments show the properties of the different proposed methods, and a validation experiment on a separate biological data source demonstrates the potential of our proposed approach.

References

[1] S. E. Schaeffer, "Graph clustering," *Computer science review*, vol. 1, no. 1, pp. 27–64, 2007.

[2] A. Buluç, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz, "Recent advances in graph partitioning," in *Algorithm Engineering*, ser. Lecture Notes in Computer Science, 2016, vol. 9220, pp. 117–158.

[3] S. Basu, I. Davidson, and K. Wagstaff, *Constrained clustering: Advances in algorithms, theory, and applications*. CRC Press, 2008.

[4] T. Dao, K. Duong, and C. Vrain, "A declarative framework for constrained clustering," in *ECML/PKDD (3)*, ser. Lecture Notes in Computer Science, vol. 8190. Springer, 2013, pp. 419–434.

[5] S. Gilpin, S. Nijssen, and I. N. Davidson, "Formalizing hierarchical clustering as integer linear programming," in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA.*, 2013.

[6] B. Babaki, T. Guns, and S. Nijssen, "Constrained clustering using column generation," in *Integration of AI and OR Techniques in Constraint Programming - 11th International Conference, CPAIOR 2014, Cork, Ireland, May 19-23, 2014. Proceedings*, 2014, pp. 438–454.

[7] J. Berg and M. Järvisalo, "Cost-optimal constrained correlation clustering via weighted partial maximum satisfiability," *Artif. Intell.*, vol. 244, pp. 110–142, 2017.

[8] P. Hansen and B. Jaumard, "Cluster analysis and mathematical programming," *Math. Program.*, vol. 79, pp. 191–215, 1997.

[9] M. Armbruster, M. Fügenschuh, C. Helmberg, and A. Martin, "A comparative study of linear and semidefinite branch-and-cut methods for solving the minimum graph bisection problem," in *IPCO*, ser. Lecture Notes in Computer Science, vol. 5035. Springer, 2008, pp. 112–124.

[10] A. Lisser and F. Rendl, "Graph partitioning using linear and semidefinite programming," *Math. Program.*, vol. 95, no. 1, pp. 91–101, 2003.

[11] N. Fan and P. M. Pardalos, "Linear and quadratic programming approaches for the general graph partitioning problem," *J. Global Optimization*, vol. 48, no. 1, pp. 57–71, 2010.

[12] C. E. Ferreira, A. Martin, C. C. de Souza, R. Weismantel, and L. A. Wolsey, "The node capacitated graph partitioning problem: A computational study," *Math. Program.*, vol. 81, pp. 229–256, 1998.

[13] M. Grötschel and Y. Wakabayashi, "A cutting plane algorithm for a clustering problem," *Math. Program.*, vol. 45, no. 1-3, pp. 59–96, 1989.

[14] A. Mehrotra and M. A. Trick, "Cliques and clustering: A combinatorial approach," *Oper. Res. Lett.*, vol. 22, no. 1, pp. 1–12, 1998.

[15] X. Ji and J. E. Mitchell, "Branch-and-price-and-cut on the clique partitioning problem with minimum clique size requirement," *Discrete Optimization*, vol. 4, no. 1, pp. 87–102, 2007.

[16] M. Labbé and F. A. Özsoy, "Size-constrained graph partitioning polytopes," *Discrete Mathematics*, vol. 310, no. 24, pp. 3473–3493, 2010.

[17] S. Benati, J. Puerto, and A. M. Rodrguez-Cha, "Clustering data that are graph connected," *European Journal of Operational Research*, 2017.

[18] M. D. Leiserson, F. Vandin, H.-T. Wu, J. R. Dobson, J. V. Eldridge, J. L. Thomas, A. Papoutsaki, Y. Kim, B. Niu, M. McLellan *et al.*, "Pan-cancer network analysis identifies combinations of rare somatic mutations across pathways and protein complexes," *Nature genetics*, vol. 47, no. 2, pp. 106–114, 2015.

[19] S. Pulido-Tamayo, B. Weytjens, D. De Maeyer, and K. Marchal, "Ssa. me detection of cancer mutual exclusivity patterns by small subnetwork analysis," *bioRxiv*, p. 034124, 2015.

[20] H. D. Sherali and J. Desai, "A global optimization rlt-based approach for solving the hard clustering problem," *J. Global Optimization*, vol. 32, no. 2, pp. 281–306, 2005.

[21] T. Guns, T. Dao, C. Vrain, and K. Duong, "Repetitive branch-and-bound using constraint programming for constrained minimum sum-of-squares clustering," in *ECAI*, ser. Frontiers in Artificial Intelligence and Applications, vol. 285. IOS Press, 2016, pp. 462–470.

[22] V. T'Kindt and J. Billaut, *Multicriteria Scheduling - Theory, Models and Algorithms (2. ed.).* Springer, 2006.

[23] R. Carvajal, M. Constantino, M. Goycoolea, J. P. Vielma, and A. Weintraub, "Imposing connectivity constraints in forest planning models," *Operations Research*, vol. 61, no. 4, pp. 824–836, 2013.

[24] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference (SciPy2008)*, 2008, pp. 11–15.

[25] J. Das and H. Yu, "Hint: High-quality protein interactomes and their applications in understanding human disease," *BMC systems biology*, vol. 6, no. 1, p. 92, 2012.

[26] H. Yu, L. Tardivo, S. Tam, E. Weiner, F. Gebreab, C. Fan, N. Svrzikapa, T. Hirozane-Kishikawa, E. Rietman, X. Yang *et al.*, "Next-generation sequencing to generate interactome datasets," *Nature methods*, vol. 8, no. 6, pp. 478–480, 2011.

[27] J. S. Parker, M. Mullins, M. C. Cheang, S. Leung, D. Voduc, T. Vickery, S. Davies, C. Fauron, X. He, Z. Hu *et al.*, "Supervised risk predictor of breast cancer based on intrinsic subtypes," *Journal of clinical oncology*, vol. 27, no. 8, pp. 1160–1167, 2009.

[28] C. G. A. Network *et al.*, "Comprehensive molecular portraits of human breast tumors," *Nature*, vol. 490, no. 7418, p. 61, 2012.

[29] M. Kircher, D. M. Witten, P. Jain, B. J. O'roak, G. M. Cooper, and J. Shendure, "A general framework for estimating the relative pathogenicity of human genetic variants," *Nature genetics*, vol. 46, no. 3, pp. 310–315, 2014.