Author/s:
Gan, J;Gleich, DF;Veldt, N;Wirth, A;Zhang, X

Title:
Graph clustering in all parameter regimes

Date:
2020-08-01

Citation:
Gan, J., Gleich, D. F., Veldt, N., Wirth, A. & Zhang, X. (2020). Graph clustering in all parameter regimes. Leibniz International Proceedings in Informatics, LIPIcs, 170, Schloss Dagstuhl. https://doi.org/10.4230/LIPIcs.MFCS.2020.39.

Persistent Link:
https://hdl.handle.net/11343/244123

Graph Clustering in All Parameter Regimes

Junhao Gan
School of Computing and Information Systems, The University of Melbourne, Victoria, Australia
junhao.gan@unimelb.edu.au

David F. Gleich
Department of Computer Science, Purdue University, West Lafayette, IN, USA
dgleich@purdue.edu

Nate Veldt
Center for Applied Mathematics, Cornell University, Ithaca, NY, USA
nveldt@cornell.edu

Anthony Wirth
School of Computing and Information Systems, The University of Melbourne, Victoria, Australia
awirth@unimelb.edu.au

Xin Zhang
School of Computing and Information Systems, The University of Melbourne, Victoria, Australia
xinz11@student.unimelb.edu.au

Abstract

Resolution parameters in graph clustering control the size and structure of clusters formed by solving a parametric objective function. Typically there is more than one meaningful way to cluster a graph, and solving the same objective function for different resolution parameters produces clusterings at different levels of granularity, each of which can be meaningful depending on the application. In this paper, we address the task of efficiently solving a parameterized graph clustering objective for *all* values of a resolution parameter. Specifically, we consider a new analysis-friendly objective we call LambdaPrime, involving a parameter $\lambda \in (0, 1)$. LambdaPrime is an adaptation of LambdaCC, a significant family of instances of the Correlation Clustering (minimization) problem. Indeed, LambdaPrime and LambdaCC are closely related to other parameterized clustering problems, such as parametric generalizations of modularity. They capture a number of specific clustering problems as special cases, including sparsest cut and cluster deletion. While previous work provides approximation results for a single value of the resolution parameter, we seek a set of approximately optimal clusterings for all values of λ in polynomial time.

More specifically, we show that when a graph has m edges and n nodes, there exists a set of at most m clusterings such that, for every $\lambda \in (0, 1)$, the family contains an optimal solution to the LambdaPrime objective. This bound is tight on star graphs. We obtain a family of $O(\log n)$ clusterings by solving the parametric linear programming (LP) relaxation of LambdaPrime at $O(\log n)$ λ values, and rounding each LP solution using existing approximation algorithms. We prove that this is asymptotically tight: for a certain class of ring graphs, for all values of λ, $\Omega(\log n)$ feasible solutions are required to provide a constant-factor approximation for the LambdaPrime LP relaxation. To minimize the size of the clustering family, we further propose an algorithm that yields a family of solutions of a size no more than twice of the minimum LP-approximating family.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Mathematics of computing → Approximation algorithms

Keywords and phrases Graph Clustering, Algorithms, Parametric Linear Programming

Digital Object Identifier 10.4230/LIPIcs.MFCS.2020.39

Related Version A full version of the paper is available at `https://arxiv.org/abs/1910.06435`.

Funding *Junhao Gan*: Supported in part by ARC DECRA DE190101118.
David F. Gleich: Supported in part by NSF awards IIS-1546488, CCF-1909528, and the NSF Center

for Science of Information STC, CCF-0939370, as well as DOE DE-SC0014543, NASA, and the Sloan Foundation.
Anthony Wirth: The Melbourne School of Engineering supported Nate Veldt's visit to Anthony Wirth.
Xin Zhang: Supported by a Melbourne Research Scholarship.

1 Introduction

Graph clustering is a task of grouping nodes of a graph into sets of nodes that share more edges with each other than the rest of the graph. This often involves, implicitly or explicitly, a trade-off between cluster size and edge-density. Hence there are a number of objective functions for graph clustering that rely on a tunable resolution parameter that controls this trade-off when optimizing the objective. Solving such an objective for a range of parameters reveals different types of clustering structure in the same graph. Applications include hierarchical clustering [17] and the detection of robust clusterings that remain optimal over a range of parameter settings [1].

1.1 Parametric Graph Clustering

Our work specifically considers a simple parametric clustering objective we call LambdaPrime. This objective can be viewed as a slight variation of the LambdaCC graph clustering framework [23], which is itself a parameterized variant of Correlation Clustering [2], with resolution parameter $\lambda \in (0,1)$. Formally, for a graph $G = (V, E)$, resolution parameter λ, and a clustering $\mathcal{C}$, the LambdaCC objective cost for $\mathcal{C}$ can be written as:

$$\textbf{LamCC}(\mathcal{C}, \lambda) = \sum_{S \in \mathcal{C}} \left(\frac{1}{2}(1-\lambda)\,\textbf{cut}(S) + \lambda \left[\binom{|S|}{2} - |E_S| \right] \right), \tag{1}$$

where $S \in \mathcal{C}$ denotes an individual cluster in $\mathcal{C}$, $\textbf{cut}(S)$ is the number of edges incident on exactly one node in S and E_S is the set of edges in E whose both end vertices are in S.

Intuitively, the value λ controls the size and density of clusters formed by optimizing the objective. When $\lambda = 1$, an optimal clustering is the one consisting of only singletons while when $\lambda = 0$, an optimal clustering groups all nodes into one cluster. Since both of these cases result in trivial clusterings, we focus on the parameter range of $(0, 1)$. Several other well-studied objective functions for graph-clustering are captured as special cases of LambdaCC for particular values of λ. These include sparsest cut [11], cluster deletion [19], and modularity clustering [15].

In our work we focus on a slight variant of the LambdaCC objective, which we call LambdaPrime. For a clustering $\mathcal{C}$, the LambdaPrime objective is given by

$$\textbf{LamPrime}(\mathcal{C}, \lambda) = \sum_{S \in \mathcal{C}} \left(\frac{1}{2}\,\textbf{cut}(S) + \lambda \binom{|S|}{2} \right). \tag{2}$$

This is exactly $\lambda|E|$ greater than the LambdaCC objective, which is constant with respect to the cluster assignment. Therefore, these two objectives share the same set of optimal solutions for the same λ. We focus on proving results for the LambdaPrime objective, as it leads to the cleanest exposition of our techniques and main results for parametric graph clustering, without fundamentally changing the nature of these results. We discuss the relationship between LambdaPrime and LambdaCC in more depth in Section 2.

A number of other closely related graph clustering objective functions also rely on tunable resolution parameters [1, 5, 17, 18, 20]. Many of these can be viewed as generalizations of the popular modularity clustering objective [15]. In this paper we broadly refer to these as *parametric* graph clustering objective functions. The modularity objective is NP-hard to approximate to within any multiplicative factor [7], and thus techniques typically used for optimizing generalizations of modularity are heuristics with no approximation guarantee [1, 5, 10, 17, 20, 18, 21]. However, LambdaPrime corresponds to a linear transformation of the modularity objective with a resolution parameter, and permits several algorithmic guarantees by techniques developed originally for Correlation Clustering [2]. In particular, on an n-node graph, for every value of λ, a standard weighted Correlation Clustering algorithm [4, 6] finds an $O(\log n)$ approximation of LambdaPrime. Several other linear programming based algorithms for the related LambdaCC problem have been developed as well for different parameter regimes [9, 23], and are transferable to LambdaPrime as well.

1.2 Clustering in All Parameter Regimes

Optimizing a parametric clustering objective over a wide range of resolution parameters is a useful approach for detecting different types of clustering structure in the same network [1, 5, 23]. Although approximation algorithms for LambdaPrime can be directly derived from existing work on LambdaCC and Correlation Clustering, these only provide approximation guarantees for a single fixed value of λ. In this paper, we focus on finding *families* of clusterings that come with rigorous optimality guarantees for an entire range of parameter values of a parametric graph clustering objective. More precisely, we say that a family of clusterings solves (or approximates) a parametric objective *in all parameter regimes* if, for every value of the resolution parameter, the family contains a solution (or approximate solution) to the objective. In our work, we seek families satisfying guarantees both in terms of the approximation factor, as well as in terms of the number of clusterings needed to attain such an approximation factor for all values of a resolution parameter.

1.3 Our Contributions

We provide new lower bounds and techniques for exactly or approximately solving the LambdaPrime objective in all parameter regimes. Section 2 formally introduces the objective and outlines the region of parameters that is not trivial to find an optimal clustering. We then tackle the task of finding a *small* family of clusterings that contains an approximately optimal clustering to every parameter, in a *reasonable* amount of time. We begin in Section 3 by examining the objective itself and proving that for any graph with n nodes and m edges, there exists a set of m or fewer clusterings that contains an optimal LambdaPrime clustering for every value of λ. However, since obtaining an optimal clustering for a particular λ is NP-hard, our primary contributions pertain to finding approximately optimal clusterings for the LambdaPrime framework using linear programming relax-and-round techniques. We show that for the relaxed LP objective, we can produce a family of solutions of size $O(\log n)$ by solving an LP at each of the $O(\log n)$ λ values. One of our central results is to show that this bound is tight on ring graphs, presented at Section 3.3. Although this result indicates that a logarithmic number of clusterings is needed in the worst case, our $O(\log n)$ upper bound is not tight in all cases. Therefore, in Section 4 we introduce a technique which, for any input graph, returns a family of clusterings that contains at most two times the minimum number of LP solutions needed to obtain a $(1+\varepsilon)$-approximation in all parameter regimes.

2 The LambdaPrime Objective

LambdaPrime is an objective function for clustering graphs based on a tunable resolution parameter, $\lambda \in (0, 1)$. The objective seeks to minimize the number of edges crossing between clusters, subject to a regularization term that controls cluster size. The LambdaPrime minimization problem can be expressed formally as an integer linear program (ILP):

$$\begin{array}{lll} \text{minimize } \mathbf{OPT}(\lambda) & = \sum_{(i,j)\in E} x_{ij} + & \sum_{i<j} \lambda(1 - x_{ij}) \\ \text{subject to} & x_{ij} \leq x_{ik} + x_{jk} & \text{for all } i, j, k \\ & x_{ij} \in \{0, 1\} & \text{for all } i, j\,. \end{array} \tag{3}$$

The variable x_{ij} represents the binary *distance* between nodes in a clustering. If $x_{ij} = 1$, nodes i and j are in different clusters, whereas $x_{ij} = 0$ indicates they are clustered together. In this way, clusterings of a graph are in one-to-one correspondence with feasible solutions to the ILP. The number of clusters to form is not determined ahead of time, but is implicitly controlled by λ and which $(i, j) \in E$.

An instance of LambdaPrime corresponds specifically to a signed graph in which *some* node pairs (those in E) share a positive edge with weight 1, and *all* node pairs share a negative edge with weight λ. A pair of nodes could share two edges. Consistent with the minimization variant of Correlation Clustering [2], separating nodes that have a positive edge results in a penalty of 1, while placing a pair of nodes in the same cluster results in a penalty of λ. The LambdaCC objective can also be expressed by an ILP with the same set of constraints and the following slightly different objective:

$$\text{minimize} \quad \sum_{(i,j)\in E}(1 - \lambda)x_{ij} + \sum_{(i,j)\notin E} \lambda(1 - x_{ij}). \tag{4}$$

Since the LambdaPrime objective is greater than LambdaCC by an additive term $\lambda|E|$, these objectives will differ in terms of approximation factors. Although LambdaCC is the harder objective to approximate, all of our results can also be adapted to apply to LambdaCC. In some cases the analogous results for LambdaCC involve slighty different constants, though this does not change the fundamental nature of our results, since our approximation factors and the size of clustering families we produce are both logarithmic. We focus here on on the LambdaPrime objective for ease of exposition. In particular, unlike LambdaCC, the LambdaPrime score is monotonically increasing with λ, which simplifies several explanations and proof details. In the full version of this paper [8], we provide further details for how to adjust our results so that they apply to LambdaCC.

2.1 Relation to Other Clustering Objectives

The optimal solutions of LambdaCC (and thus the optimal solutions of LambdaPrime) interpolate between solutions to the sparsest cut and cluster deletion objectives [23]. The scaled sparsest cut of a graph $G = (V, E)$ is the bipartition $\{S, \bar{S}\}$ that minimizes the ratio cut objective $\mathbf{cut}(S)/(|S||\bar{S}|)$. Cluster deletion is the problem of partitioning G into cliques in a way that minimizes the number of edges between cliques. Deriving from previous work on LambdaCC [23], we formalize the relationship between these two objectives for LambdaPrime in the following theorem.

▶ **Theorem 1.** *Let $G = (V, E)$ be a graph, and define $\lambda^* = \min_{S\subset V} \boldsymbol{cut}(S)/(|S||\bar{S}|)$.*

- *For any $\lambda \leq \lambda^*$, placing all nodes in one cluster optimizes the LambdaPrime objective.*
- *For any $\lambda \in (\lambda^*, 1)$, the optimal LambdaPrime clustering will contain at least two clusters. There exists some $\lambda > \lambda^*$ such that the optimal LambdaPrime clustering will be the bipartition $\{S^*, \bar{S}^*\}$ which minimizes $\boldsymbol{cut}(S)/(|S||\bar{S}|)$.*

- *For any $\lambda \in (|E|/(1+|E|), 1)$, the optimal LambdaPrime clustering will be optimal for the cluster deletion problem. In other words, all clusters will be cliques, and the number of edges crossing between clusters will be minimized.*

For any connected graph, $\lambda^* \geq 4/n^2$, with equality for a graph that can be split into two equal-sized sets, with a single edge between the sets. Thus, when searching for clusterings that optimize LambdaPrime, it suffices to consider $\lambda \in \left(\frac{4}{n^2}, 1\right)$. LambdaPrime is also related to other generalized clustering objectives that rely on tunable resolution parameters, including the constant Potts model [20], and generalizations of the modularity objective that include a resolution parameter [1, 18]. For the appropriate parameter settings, these objectives are equivalent at optimality, though different in terms of approximations.

2.2 Approximations via Linear Programming

Since LambdaPrime corresponds to a special weighted variant of Correlation Clustering, there is a $O(\log n)$ approximation guarantee for the objective for any value of the parameter λ [6]. This is obtained by solving and rounding the LP relaxation of (3), where we replace the binary constraint $x_{ij} \in \{0, 1\}$ with the linear constraint $0 \leq x_{ij} \leq 1$. Although better approximation guarantees exist for $\lambda \geq 1/2$, in the worst case, the LP relaxation has an $\Omega(\log n)$ integrality gap, which can be shown by slightly adapting the integrality-gap proof for LambdaCC [9]. In this paper, our goal is not to obtain new approximation guarantees for fixed values of λ. Instead, we show how to obtain a family of solutions that is approximately optimal, for *all* values of the parameter, via a small number of LP solves.

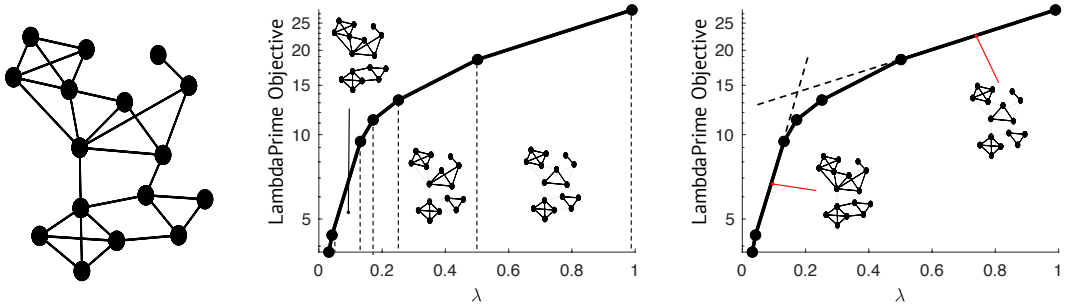
If we do not treat λ as a fixed value, objective (3) corresponds to a parametric integer linear program in λ. Let $\mathbf{OPT}(\lambda)$ denote the optimal ILP score at a certain value of λ: the **OPT** function is known to be concave and piecewise-linear in λ [3]. The *breakpoints* of a parametric ILP are values of the parameter λ at which a slope change occurs. In this context, a slope change corresponds to a parameter λ where the optimal clustering for LambdaPrime changes. Similarly, the LP relaxation of (3) is a parametric linear program, whose solution we denote by $\mathbf{LP}(\lambda)$. Since for a fixed LP solution the objective value is linear in λ, same as the ILP, $\mathbf{LP}(\lambda)$ is also concave and piecewise linear in terms of λ.

Breakpoints for the parametric LP are places at which the optimal feasible solution changes. Previous work on parametric programming has shown that in the worst case, parametric integer programs and parametric linear programs may have an exponential number of breakpoints [3, 14]; we prove here that this is not the case for LambdaPrime ILP, but the upper bound on LambdaPrime LP objective is still open.

2.3 Concave Function Approximation

Finding an optimal solution for either **OPT** or **LP** at a single value of λ corresponds to evaluating a function at a single point. Approximating either function over a range of λ values can be achieved by approximating a concave, piecewise-linear function with another concave and piecewise-linear function constructed from a set of clusterings (or feasible LP solutions in the case of the LP relaxation).

Figure 1 displays the $\mathbf{OPT}(\lambda)$ function for a small synthetic graph. Each linear piece in the plot corresponds to a clustering that remains optimal over a range of λ values. In addition to being concave and piecewise linear, note that **OPT** is strictly increasing in λ, which will be the case for the LambdaPrime objective (3) and its LP relaxation on every graph. Due to the size and structure of the graph in Figure 1, solutions to the LambdaPrime ILP and LP are in fact the same, i.e., $\mathbf{OPT}(\lambda) = \mathbf{LP}(\lambda)$ for all $\lambda \in (0, 1)$. Typically this



(a) A small graph. **(b)** LambdaPrime $\mathbf{OPT}(\lambda)$. **(c)** Clusterings → linear pieces.

Figure 1 The concave, piecewise-linear $\mathbf{OPT}(\lambda)$ of objective costs for a small synthetic graph. The linear pieces form an approximation of $\mathbf{OPT}(\lambda)$.

will not be the case in practice. For larger graphs, it will be prohibitively expensive to compute ILP solutions, but solving the LP relaxation can still be accomplished in polynomial time, which is more reasonable in practice. In recent work, a subset of the current authors showed how the linear programming relaxation of Correlation Clustering (and by extension LambdaPrime) can be solved in practice using memory-efficient projection methods [22].

Given a family of clusterings, $\mathscr{C}$, we can define a new piecewise-linear function that approximates the LambdaPrime objective by identifying the clustering in $\mathscr{C}$ that best approximates LambdaPrime for a certain range of λ values. In Figure 1 we illustrate this idea by extracting a sub-family of the optimal clusterings for the same small synthetic graph. The new approximate function has a smaller number of linear pieces, since we have selected a strict sub-family of clusterings, and upper bounds the function $\mathbf{OPT}$. In general, the same principle holds for approximating the LambdaPrime LP relaxation. We will typically approximate the LP relaxation of a graph for all λ's by finding a subfamily of feasible solutions, each of which *exactly* minimizes the LP for some λ, and corresponds to one of the linear pieces of the function $\mathbf{LP}$. Done carefully, the resulting piecewise-linear curve remains a good approximation for $\mathbf{LP}$, despite containing far fewer linear pieces.

3 Approximate Solutions in All Parameter Regimes

In this section, we provide a detailed discussion on our approach of obtaining a set family of clusterings that contains an approximately optimal solution to the LambdaPrime objective for every value of λ in $(4/n^2, 1)$ (a range justified in Section 2.1). We begin by exploring the characteristics of the concave and piecewise-linear function we wish to approximate, first for the ILP objective and then for the LP relaxation.

Algorithmically, our approach to the task is to first compute a family of solutions for the LambdaPrime LP, and then round those solutions to clusterings. In obtaining the solutions, we wish to control both the size of the clustering family and the runtime. In general, an exponential number of solutions may be needed to optimally solve a parametric LP in all parameter regimes [14]. In Theorem 3 we obtain an approximating family of clusterings of size $O(\log n)$ by solving $O(\log n)$ LPs.

3.1 Bounding the Size of Optimal Solution Families

We begin with a bound on the number of clusterings needed to *optimally* solve LambdaPrime in all parameter regimes.

▶ **Theorem 2.** *Given a graph $G = (V, E)$, there exists a family of $|E|$ or fewer clusterings which, for every value of $\lambda \in (0, 1)$, contains an optimal LambdaPrime clustering for that λ. On star graphs, this bound is tight.*

Proof. Let $0 < \lambda_1 < \lambda_2 < \cdots < \lambda_k < 1$ denote the breakpoints of the ILP, and also let $\lambda_0 = 0$ and $\lambda_{k+1} = 1$. For $t = 0, 1, 2, \ldots, k$, let $\mathbf{x}^t = (x_{ij}^t)$ denote the feasible ILP solution that is optimal in the range $\lambda \in [\lambda_t, \lambda_{t+1}]$. Each $\mathbf{x}^t$ encodes a clustering $\mathcal{C}_t$ of G that is optimal in this range and corresponds to a linear piece of **OPT**. For each clustering $\mathcal{C}_t$, define the total number of positive and negative mistakes, respectively,

$$P_t = \sum_{(i,j)\in E} x_{ij}^t = \frac{1}{2} \sum_{S\in\mathcal{C}_t} \mathbf{cut}(S), \qquad N_t = \sum_{i<j} (1 - x_{ij}^t) = \sum_{S\in\mathcal{C}_t} \binom{|S|}{2},$$

so that the LambdaPrime objective for an arbitrary λ is $\mathbf{LamPrime}(\mathcal{C}_t, \lambda) = P_t + \lambda N_t$. We then show $P_t < P_{t+1}$ for $t = 0, 1, \ldots (k-1)$. Since $\mathcal{C}_t$ is optimal over $\lambda \in [\lambda_t, \lambda_{t+1}]$ and $\mathcal{C}_{t+1}$ is optimal for $\lambda \in [\lambda_{t+1}, \lambda_{t+2}]$, both clusterings are optimal at λ_{t+1}, and therefore

$$P_t + \lambda_{t+1} N_t = P_{t+1} + \lambda_{t+1} N_{t+1}.$$

If $P_t = P_{t+1}$, then $N_t = N_{t+1}$, contradicting the fact that these clusterings are optimal for *different* parameter ranges. If $P_t > P_{t+1}$, then $N_t < N_{t+1}$, which would imply that for $\lambda > \lambda_{t+1}$, $\mathcal{C}_t$ would be a better approximation than $\mathcal{C}_{t+1}$, a contradiction to the fact that **OPT** is a concave, increasing, and piecewise-linear function. Thus, $P_t < P_{t+1}$. Since the graph is unweighted, there are at most $|E|$ possible values for P_t for $t = 0, 1, \ldots, k$, and therefore at most $|E|$ clusterings in an optimal family. This concludes the upper bound proof.

We then show that the bound is tight for star graphs. Consider an n-node star graph where the node connecting to every other node is the central node, and we refer to all other nodes as outer nodes. The optimal sparsest cut solution places one outer node with the central node, and all other nodes in a second cluster. For cluster deletion, the optimal solution places one outer node with the central node, and each other node in a singleton cluster. The LambdaPrime ILP interpolates between two solutions. The minimum scaled sparsest cut of the star graph is $\lambda^* = \frac{1}{n-1}$, so this will be the first breakpoint of the ILP. The final breakpoint is at $\lambda = \frac{1}{2}$, above which point the cluster deletion solution is optimal. As λ decreases from $\lambda = \frac{1}{2}$ to $\lambda = \frac{1}{n-1}$, the optimal solution will add outer nodes one by one to the cluster containing the central node. There will be exactly $|E|$ such clusterings, counting down until all outer nodes have been merged with the central node. ◀

Theorem 2 tells us that even though there are an exponential number of ways to cluster a graph, a linear number of clusterings characterizes an optimal family for LambdaPrime. Furthermore, since the theorem is specifically proven for optimal clusterings, the same result also holds for related parametric clustering objectives that correspond to linear transformations of LambdaPrime and its weighted variants, including LambdaCC [23], Potts models from statistical mechanics [20, 17], and variants of modularity clustering with a resolution parameter [1, 18]. Finally, this theorem proves that the parametric integer linear program (ILP) corresponding to the LambdaPrime objective, a piecewise-linear function, has a linear number of *breakpoints*. This is significant given that in general, parametric ILPs may contain an exponential number of breakpoints [3].

3.2 Obtaining Approximate Solutions

Although we have shown that the LambdaPrime ILP has at most $|E|$ breakpoints, this proof does not hold for the LP relaxation, and we have no guarantee that the number of LP breakpoints is not exponential [14]. Without a polynomial bound on the number of breakpoints, any algorithm which relies on a family of optimal solutions for the LP relaxation for all values of λ is not guaranteed to run in polynomial time. We overcome this challenge by bounding the number of solutions needed to *approximate* the LP in all regimes.

▶ **Theorem 3.** *For any $\varepsilon > 0$, there exists a (poly-time computable) family of $O(\log_{1+\varepsilon} n)$ feasible LP solutions which, for every value of λ, contains a $(1+\varepsilon)$-approximate solution to the LambdaPrime LP relaxation.*

We can round each LP solution in the family obtained in Theorem 3 using existing techniques [4], which guarantees we have an $O(\log n)$-approximation to the LambdaPrime objective. This theorem can be viewed as an application of the results of Magnanti and Stratila [13] on concave function approximation, specifically to the problem of parametric graph clustering. We first prove a lemma showing how well an optimal solution for the LP at one value of λ approximates the LP when a nearby resolution parameter is used. A solution to a minimization problem is a *δ-approximate* if the objective cost of it is no more than δ times the cost of the optimal.

▶ **Lemma 4.** *Let (x_{ij}^t) and (x_{ij}^{t+1}) be optimal solutions to the LambdaPrime ILP (respectively the LP relaxation) for resolution parameters $\lambda_t < \lambda_{t+1}$. Let $\delta = \lambda_{t+1}/\lambda_t$. Then (x_{ij}^t) is a δ-approximate solution for the LambdaPrime LP when λ_{t+1} is used, and (x_{ij}^{t+1}) is a δ-approximate solution for the LP when λ_t is used.*

Proof. For $k \in \{t, t+1\}$, define $P_k = \sum_{(i,j)\in E} x_{ij}^k$, total weights on positive violations and $N_k = \sum_{i<j}(1-x_{ij}^k)$, total weights on negative violations, so that the LambdaPrime score for (x_{ij}^k) at an arbitrary value of λ is $P_k + \lambda N_k$, a quantity we denote by $A_k(\lambda)$. Since (x_{ij}^{t+1}) and (x_{ij}^t) are optimal for their respective resolution parameters, and $\lambda_t < \lambda_{t+1}$, we have the following sequence of inequalities:

$$A_{t+1}(\lambda_t) \le A_{t+1}(\lambda_{t+1}) \le A_t(\lambda_{t+1}) < \frac{\lambda_{t+1}}{\lambda_t}\left(A_t(\lambda_t)\right) < \frac{\lambda_{t+1}}{\lambda_t}\left(A_{t+1}(\lambda_{t+1})\right).$$

Thus, both (x_{ij}^t) and (x_{ij}^{t+1}) are at worst a δ-approximation across the entire interval $[\lambda_t, \lambda_{t+1}]$, where $\delta = \lambda_{t+1}/\lambda_t$. ◀

We use Lemma 4 to construct a sequence of λ values and corresponding optimal LP solutions, to approximate the LambdaPrime objective in all parameter regimes.

Proof of theorem 3. For $\lambda < 4/n^2$, the optimal clustering will place all nodes in a single cluster, so we do not need to consider LP solutions below this threshold. Set $\lambda_1 = 4/n^2$ and let $q = \left\lfloor \log_{(1+\varepsilon)^2}(n^2/4) \right\rfloor + 1$. For $k = 2, 3, \ldots, q$, recursively define a sequence of λ values by setting $\lambda_k = (1+\varepsilon)^2\lambda_{k-1}$, and let $\lambda_{q+1} = 1/(1+\varepsilon)$. Evaluate the LambdaPrime LP relaxation at each of these λ values to obtain solutions $(x_{ij}^1), (x_{ij}^2), \ldots, (x_{ij}^{q+1})$. By Lemma 4, (x_{ij}^1) is a $(1+\varepsilon)$-approximate solution for all $\lambda \in [\lambda_1, (1+\varepsilon)\lambda_1]$, (x_{ij}^{q+1}) is a $(1+\varepsilon)$-approximation for $\lambda \in [(1+\varepsilon)\lambda_q, 1)$ and for any $k \in \{2, 3, \ldots, q\}$, (x_{ij}^k) is a $(1+\varepsilon)$-approximate solution for all $\lambda \in \left[(1+\varepsilon)\lambda_{k-1}, (1+\varepsilon)\lambda_k\right]$. Thus, using $q + 1 < \lfloor 2\log_{(1+\varepsilon)^2}(n) \rfloor + 2$ feasible solutions, we obtain a $(1+\varepsilon)$-approximate solution for every $\lambda \in [4/n^2, 1)$. ◀

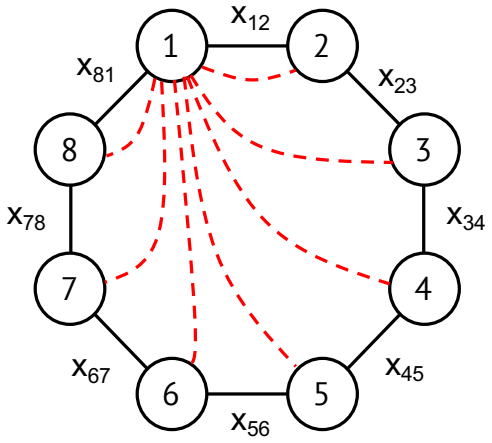


Figure 2 Ring graph G_3 with $n = 2^3$ nodes. Each pair of nodes shares a negative edge of weight λ. For simplicity, only the negative edges adjacent to Node 1 are shown, with red dashed lines; all other negative edges are omitted.

Theorem 3 shows that as ε tends to 0, the number of clusterings in our computed family is $O(\frac{1}{\varepsilon}\log n)$. However, given that our aim is simply to round LambdaPrime LP solutions to produce clusterings that are within $O(\log n)$ of optimal, it suffices to treat ε as a constant (e.g., $\varepsilon = 1$ or larger). Thus, in practice, the size of the approximating family is $O(\log n)$.

3.3 Asymptotic Tightness of the Set Family

One of the central contributions of our work is a proof that our logarithmic upper bound for approximating the LP relaxation in all parameter regimes is in fact asymptotically tight for the class of *ring graphs*, $G_k = (V, E)$ with $n = 2^k$ nodes ($k \in \mathbb{N}$ and $k \geq 3$). Without loss of generality, we assume that the nodes in a ring graph are always ordered so that node i is adjacent to node $i+1$ for $i = 1, 2, \ldots, n-1$, and nodes 1 and n are adjacent. Specifically for a ring graph, every edge $(i, j) \in E$ is viewed as a positive edge $(i, j) \in E^+$ with weight one, and for every $(i, j) \in V \times V$ there is a negative edge $(i, j) \in E^-$ with weight λ. Figure 2 displays a picture of G_3, the smallest graph in this class.

▶ **Theorem 5.** *For the class of ring graphs with $n = 2^k$ nodes ($k \in \mathbb{N}$ and $k \geq 3$), for every $\varepsilon > 0$, at least $\Omega(\log n)$ feasible LP solutions are needed in order to approximate the LP relaxation of LambdaPrime in all parameter regimes to within a factor $(1 + \varepsilon)$.*

The proof of this result relies on several connections between our parametric clustering problem and concave function approximation [12]. The optimal solution curve for the LambdaPrime LP relaxation corresponds to an increasing, concave, and piecewise-linear function of λ [3]. Approximating the LP relaxation in all parameter regimes with a small number of feasible solutions is therefore equivalent to finding another piecewise-linear curve with a small number of linear pieces. Previously, Magnanti and Stratila [13, 12] demonstrated that in order to approximate the square root function $sqrt(x) = \sqrt{x}$ via a piecewise-linear upper bound over an interval $[a, b]$, at least $\Omega(\log \frac{b}{a})$ linear pieces are needed. Although this bound does not immediately imply any result for parametric clustering, we use this as a step in proving a similar lower bound for the LambdaPrime LP relaxation. In particular, the proof of Theorem 5 follows from the following observations and theorems.

▶ **Observation 1.** *The sparsest cut of a ring graph is induced by a set S of $\frac{n}{2}$ nodes which are connected via a path: the cut ratio is $\lambda^* = \frac{\boldsymbol{cut}(S)}{|S||\bar{S}|} = \frac{2}{(n/2)(n/2)} = \frac{8}{n^2}$.*

▶ **Observation 2.** *For any $\lambda \geq 1/2$, pairing up the n nodes of a ring graph G_k into $n/2$ disjoint edges optimizes the LambdaPrime objective.*

By Observation 1 and according to the first bullet point in Theorem 1, for any $\lambda \leq \lambda^* = \frac{8}{n^2}$, placing all nodes in one cluster optimizes LambdaPrime objective on a ring graph. Combining with Observation 2, it suffices to consider $\lambda \in \left[\frac{8}{n^2}, \frac{1}{2}\right]$. Below is a crucial theorem for showing the lower bound, which is arguably one of the most interesting results in this paper.

▶ **Theorem 6.** *For any $\lambda \in \left[\frac{8}{n^2}, \frac{1}{2}\right]$ the optimal value of the LambdaPrime LP relaxation for the ring graph is*

$$\boldsymbol{LP}(\lambda) = \min_{t \in \mathbb{N}} \frac{n}{t}\left(1 + \lambda\binom{t}{2}\right). \tag{5}$$

Proof. To express the optimal value of the LP solutions, we deploy an alternative LP relaxation, originally considered by Wirth [24] for the unweighted version of Correlation Clustering. Each constraint in LP (6) corresponds to a *Negative Edge with Positive Path Cycle* (NEPPC), where $NEPPC(i_1, i_2, \ldots, i_m)$ represents a sequence (i.e., a *path*) of (positive) edges, $\{(i_1, i_2), (i_2, i_3), \ldots, (i_{m-1}, i_m)\} \subseteq E$, with a single non-edge (i.e., negative edge) completing the cycle: $(i_1, i_m) \in E^-$.

$$\begin{array}{lll} \text{minimize} & \sum_{(i,j)\in E}(1-\lambda)x_{ij} + & \sum_{(i,j)\notin E}\lambda(1-x_{ij}) \\ \text{subject to} & x_{i_1,i_m} \leq \sum_{j=1}^{m-1} x_{i_j,i_{j+1}} & \text{for all } NEPPC(i_1, i_2, \ldots, i_m) \\ & x_{ij} \leq 1 & \text{for all } (i,j) \notin E \\ & 0 \leq x_{ij} & \text{for all } (i,j)\,. \end{array} \tag{6}$$

Wirth [24] proved that the set of optimal solutions to the NEPPC LP (6) is exactly the same as the set of optimal solutions to the canonical LP. Via the NEPPC LP, we show that the optimal value of the LP solution on ring graphs for any $\lambda \in [8/n^2, 1/2]$ is expression (5). We start with two straightforward observations about the NEPPC objective (on ring graphs). In the following, we assume all subscripts are computed modulo n. For example, we express the positive-edge LP *distances* as $x_{i,i+1}$, for $i = 1, \ldots, n$, with the understanding that, $x_{n,n+1} \equiv x_{n,1} \equiv x_{1,n}$.

▶ **Observation 3** (Wirth [24])**.** *Every negative edge $(i,j) \in E^-$ is either involved in a tight NEPPC constraint, or $x_{ij} = 1$.*

▶ **Observation 4.** *If we assign $x_{i,i+1} = c$ for all $i = 1, 2, \ldots, n$ for some constant $0 \leq c \leq 1$, then for this fixed assignment, the LP will be minimized if $x_{ij} = \min\{1, c \cdot dist(i,j)\}$ for each $(i,j) \in E^-$, where $dist(i,j)$ is the shortest-path distance (in terms of the number of positive edges) between nodes i and j in the ring graph.*

Step 1: All positive-edge LP distances are equal

Let $\mathbf{x}^1 = (x^1_{ij})$ be an arbitrary solution to the NEPPC LP relaxation (6). Construct $n-1$ other optimal solutions, $\mathbf{x}^2, \mathbf{x}^3, \ldots, \mathbf{x}^n$, by setting $x^t_{i,j} = x^1_{i+t,j+t}$, for all $i < j$ and $t = 2, 3, \ldots, n$. In other words, we exploit the symmetry in the ring graph and *rotate* LP distances around the ring, one node at a time, to produce each new optimal solution. Then, let

$$\mathbf{x}^* = \frac{1}{n}\sum_{j=1}^{n} \mathbf{x}^j\,.$$

Since $\mathbf{x}^*$ is a convex combination of optimal LP solutions, it is itself an optimal LP solution. Furthermore, for every $i = 1, 2, \ldots, n$,

$$x^*_{i,i+1} = \frac{1}{n}\sum_{j=1}^{n} x^j_{i,i+1} = \frac{1}{n}\sum_{j=1}^{n} x^1_{i+j-1,i+j}\,.$$

The right-hand side is the sum of all positive-edge distances in the LP solution $\mathbf{x}^1$. Therefore, all $x^*_{i,i+1}$ have the same value, denoted by $c^* \geq 0$. By Observation 4, for each $(i,j) \notin E$, $x^*_{ij} = \min\{1, c^* \cdot \mathit{dist}(i,j)\}$. Let $\mathbf{LP}(\lambda, c)$ denote the value of the LP relaxation for the given value of λ, with all positive edges having distance c. As a result, we have:

$$\mathbf{LP}(\lambda) = \mathbf{LP}(\lambda, c^*) = \min_{c \geq 0} \mathbf{LP}(\lambda, c)\,.$$

Step 2: Bounding c^* from below

Next, we show that when $\lambda \geq 8/n^2$, the constant c^* satisfies $\frac{2}{n} \leq c^* \leq 1$. First, consider the value of $\mathbf{LP}(\lambda, c)$. By symmetry, it suffices to focus on the LP cost associated with node 1, and then multiply by n. In particular, we charge three types of costs to node 1:

- the LP cost of the *clockwise* positive edge, $x_{1,2} = c$ (the cost of the counter-clockwise positive edge is charged to node n);
- the LP cost of the *clockwise* negative edges, i.e., $\lambda \cdot (1 - x_{1,(i+1)})$, for $i = 1, \ldots, \frac{n}{2} - 1$: since $x_{1,(i+1)} = \min\{1, c \cdot \mathit{dist}(1,(i+1))\}$, this is, $\lambda \cdot \max\{0, 1 - c \cdot i\}$;
- *half* of the LP cost of the negative edge $(1, n/2+1)$, i.e., $\frac{1}{2}\lambda \cdot (1 - x_{1,(n/2+1)}) = \frac{1}{2}\lambda \cdot \max\{0, (1 - \frac{n \cdot c}{2})\}$, sharing the cost of this *diameter* between nodes 1 and $n/2+1$.

Therefore, we have:

$$\mathbf{LP}(\lambda, c) = n\left(c + \lambda \sum_{i=1}^{n/2-1} \max\{0, (1 - ci)\} + \frac{1}{2}\lambda \cdot \max\{0, \left(1 - \frac{nc}{2}\right)\}\right). \tag{7}$$

If $0 \leq c \leq \frac{2}{n}$, then $1 - \frac{n \cdot c}{2} \geq 0$, and Equation (7) equals

$$\mathbf{LP}(\lambda, c) = n \cdot c \cdot \left(1 - \frac{\lambda n^2}{8}\right) + \lambda\binom{n}{2}. \tag{8}$$

Because $\lambda \geq \frac{8}{n^2}$, expression (8) is a strictly decreasing function of c on $[0, 2/n]$. Since we are seeking the value of c^*, we henceforth assume $c \geq 2/n$.

Step 3: Proving existence of integral $1/c^*$

If $\frac{2}{n} \leq c \leq 1$, then $\lfloor \frac{1}{c} \rfloor + 1 \leq i \leq \frac{n}{2}$, implies $1 - ci \leq 0$ and Equation (7) equals

$$\mathbf{LP}(\lambda, c) = n\left(c + \lambda\left\lfloor\frac{1}{c}\right\rfloor - \frac{\lambda c}{2}\left\lfloor\frac{1}{c}\right\rfloor\left(\left\lfloor\frac{1}{c}\right\rfloor + 1\right)\right). \tag{9}$$

We prove the existence of an integral $1/c^*$ value by contradiction: suppose that $1/c^*$ is *not* an integer. Put $t = \lfloor \frac{1}{c^*} \rfloor$. Since $n/2$ is an integer, $1/c^* < n/2$, hence $t \leq n/2 - 1$. Therefore, for all c such that $\lfloor \frac{1}{c} \rfloor = t$, i.e., $c \in (1/(t+1), 1/t]$, expression (9) becomes

$$\mathbf{LP}(\lambda, c) = n\left(c + \lambda t - \frac{\lambda c}{2} t\,(t+1)\right) = nc\left(1 - \lambda\frac{t(t+1)}{2}\right) + \lambda n t\,. \tag{10}$$

We select two new values, c_ℓ and c_r, satisfying:

$$\frac{2}{n} \leq \frac{1}{t+1} < c_\ell < c^* < c_r \leq \frac{1}{t}\,.$$

Not only do values c_ℓ and c_r bound c^* below and above, but also applying the floor function to all three reciprocals yields the same integer, t.

If $1 - \lambda t(t+1)/2 < 0$, then expression (10) shows that $\mathbf{LP}(\lambda, c_r) < \mathbf{LP}(\lambda, c^*)$. Likewise, if $1 - \lambda t(t+1)/2 > 0$, then $\mathbf{LP}(\lambda, c_\ell) < \mathbf{LP}(\lambda, c^*)$. In each of these cases, c^* would not be the optimum setting of c.

However, if $1-\lambda t(t+1)/2 = 0$, then $\mathbf{LP}(\lambda, c)$ is constant for $c \in (1/(t+1), 1/t]$. Hence $1/c^*$ can be assumed to be the integer t. Substituting into the middle part of Equation (10), the LambdaPrime LP relaxation optimum is

$$\mathbf{LP}(\lambda) = \min_{t\in\mathbb{N}} n\left(\frac{1}{t} + \lambda t - \lambda\frac{(t+1)}{2}\right) = \min_{t\in\mathbb{N}} \frac{n}{t}\left(1 + \lambda\binom{t}{2}\right).$$

Theorem 6 follows. ◀

Based on Theorem 6, Lemma 7, proved in the full version [8], shows that $\mathbf{LP}(\lambda)$ and $\mathbf{OPT}(\lambda)$ are bounded below and above by a square root function of λ.

▶ **Lemma 7.** *Let* $q(\lambda) = \frac{3n}{4}\sqrt{2\lambda}$. *For all* $\lambda \in \left[\frac{8}{n^2}, \frac{1}{2}\right]$,

$$q(\lambda) \leq \boldsymbol{LP}(\lambda) \leq \boldsymbol{OPT}(\lambda) \leq \frac{4\sqrt{2}}{3} q(\lambda). \tag{11}$$

This lemma shows that the LambdaPrime LP relaxation on the ring graph behaves very similarly to the square root function. In the full version [8], we prove that Theorem 5 follows by combining this bound with the lower bound results of Mangnanti and Stratila [13] on approximating the square root function.

4 Towards Overcoming the Logarithmic Barrier

While in the worst case, $\Omega(\log n)$ feasible solutions are required to approximate the LP relaxation in all parameter regimes, this is not necessarily the case for every graph. In the full version [8], we show that on star graphs, a single LP solution suffices to optimize the LP relaxation in all non-trivial parameter regimes. Thus, an algorithm that can obtain a family of solutions with an instance-specific size bound would be interesting and more practical.

Motivated by this observation, we propose an algorithm which can possibly find a smaller family than that returned by the method in Theorem 3. The main idea of our algorithm is to carefully select the λ values for which we solve LP relaxation, and then actually compute, rather than just bound, the range of λ values for which the resulting LP solution is approximately optimal. In this way, we will need to evaluate the LP relaxation at fewer λ values. As we show shortly, the family returned by our new algorithm is *nearly optimal* (i.e., at most twice) in terms of size, compared to the *minimum* family of solutions that contains an $(1+\varepsilon)$-approximation for every λ.

4.1 The Frontier Extension Algorithm

Next we introduce the Frontier Extension (FE) algorithm that can return a valid family to $(1+\varepsilon)$-approximate all λ with size at most twice of the minimum valid family.

The key observation in the design of the FE algorithm is that, given an optimal solution to the LP relaxation at a value λ_0, we can in fact exactly compute values (θ^-, θ^+) such that the LP solution is a $(1+\varepsilon)$ approximation exactly on the interval $[\lambda_0+\theta^-, \lambda_0+\theta^+]$. We refer to this as the $(1+\varepsilon)$-*Approximate Covering Range* (for short, $(1+\varepsilon)$-ACR) of λ_0. According to the framework of Nowozin and Jegelka [16], given an optimal LambdaPrime LP relaxation solution at λ_0, the $(1+\varepsilon)$-ACR of λ_0 can be computed by solving *two* "dual-like" LP's: one for computing θ^- and the other for θ^+. We discuss the details in the full version. Based on this, starting from $\lambda_0 = 4/n^2$, the FE algorithm solves the LP relaxation at λ_0, and then computes the right endpoint of the $(1+\varepsilon)$-ACR of λ_0. It then updates λ_0 to be this right endpoint, and repeats this process until a point where the right endpoint of the $(1+\varepsilon)$-ACR of the latest λ_0 is greater than or equal to 1. This procedure is shown in Algorithm 1.

Algorithm 1 Frontier Extension (FE) Algorithm (High Level).

```
1: function FE(G, ε)
2:     λ_0 ← 4/n^2
3:     while λ_0 < 1 do
4:         Compute the optimal LP solution x* ← LP(λ_0)
5:         Update λ_0 to the right end of the (1+ε)-ACR of λ_0
6:     return 𝒞, set of all the computed x* solutions.
```

After running Algorithm 1, it is possible for some λ values to be covered by more than one $(1+\varepsilon)$-ACR of the λ values at which we evaluated the LP. The size of the returned LP solution family could be further reduced by a post-processing step that removes any *redundant* solution. Done carefully, the size of the resulting family will be at most twice the size of the minimum LP solution family. We formalize this result in the following theorem.

▶ **Theorem 8.** *For $\varepsilon > 0$, let M_ε be the minimum number of LP solutions needed to approximate the LambdaPrime LP relaxation in all parameter regimes to within a factor $(1+\varepsilon)$. We obtain a family of $2M_\varepsilon$ or fewer LP solutions that contains a $(1+\varepsilon)$-approximate solution to the LP in every parameter regime.*

It should be noted that the above bound applies to the *size* of the family of LP solutions. In practice, however, we may need to evaluate the LP relaxation more than $2M_\varepsilon$ times to actually obtain this family. Furthermore, in the worst case, we may still need to evaluate the LP up to $O(\log n)$ times. Nevertheless, this result shows that, without changing our worst-case asymptotic runtime, we can find a family of LP solutions that is nearly optimal in terms of output size.

5 Discussion and Future Work

In this work we demonstrate how to find a family of clusterings that contains, for every resolution parameter value, an approximate solution to the LambdaPrime (ILP) objective. Our results come with rigorous guarantees both in terms of the approximation factor and in terms of the number of clusterings needed to approximate the objective in all parameter regimes. We provide a means to obtain a family of clusterings of size $O(\log n)$, by solving $O(\log n)$ LPs. The size of the family is asymptotically tight as shown by a lower bound established for the ring graphs.

In the future, building on the FE algorithm, we seek techniques for finding provably small approximating families without having to evaluate the LP relaxation a logarithmic number of times. We also seek to better understand upper and lower bounds on the number of clusterings needed to approximate or exactly solve the LambdaPrime objective on other special classes of graphs.

References

1 A Arenas, A Fernández, and S Gómez. Analysis of the structure of complex networks at different resolution levels. *New Journal of Physics*, 10(5):053039, May 2008. doi:10.1088/1367-2630/10/5/053039.

2 Nikhil Bansal, Avrim Blum, and Shuchi Shawla. Correlation clustering. *Machine Learning*, 56:89–113, 2004.

3 Patricia J. Carstensen. Complexity of some parametric integer and network programming problems. *Math. Program.*, 26(1):64–75, May 1983. doi:10.1007/BF02591893.

4 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383, 2005. doi:10.1016/j.jcss.2004.10.012.

5 J.-C. Delvenne, S. N. Yaliraki, and M. Barahona. Stability of graph communities across time scales. *Proceedings of the National Academy of Sciences*, 107(29):12755–12760, 2010. doi:10.1073/pnas.0903215107.

6 Erik D. Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2):172–187, 2006. Approximation and Online Algorithms. doi:10.1016/j.tcs.2006.05.008.

7 Thang N. Dinh, Xiang Li, and My T. Thai. Network clustering via maximizing modularity: Approximation algorithms and theoretical limits. In *ICDM*, pages 101–110, 2015. doi:10.1109/ICDM.2015.139.

8 Junhao Gan, David F. Gleich, Nate Veldt, Anthony Wirth, and Xin Zhang. Graph clustering in all parameter regimes. *arXiv*, cs.CC, 2019.

9 David F. Gleich, Nate Veldt, and Anthony Wirth. Correlation Clustering Generalized. In *ISAAC*, pages 44:1–44:13, 2018. doi:10.4230/LIPIcs.ISAAC.2018.44.

10 Lucas G. S. Jeub, Marya Bazzi, Inderjit S. Jutla, and Peter J. Mucha. A generalized Louvain method for community detection implemented in MATLAB, 2011–2019. https://github.com/GenLouvain/GenLouvain.

11 Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM (JACM)*, 46(6):787–832, November 1999.

12 Thomas L. Magnanti and Dan Stratila. Separable concave optimization approximately equals piecewise linear optimization. In *IPCO*, pages 234–243, 2004.

13 Thomas L Magnanti and Dan Stratila. Separable concave optimization approximately equals piecewise-linear optimization. *arXiv preprint arXiv:1201.3148*, 2012.

14 Katta G. Murty. Computational complexity of parametric linear programming. *Mathematical Programming*, 19(1):213–219, December 1980. doi:10.1007/BF01581642.

15 Mark EJ Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(026113), 2004.

16 Sebastian Nowozin and Stefanie Jegelka. Solution stability in linear programming relaxations: Graph partitioning and unsupervised learning. In *ICML*, pages 769–776. ACM, 2009.

17 Jörg Reichardt and Stefan Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Physical Review Letters*, 93:218701, November 2004. doi:10.1103/PhysRevLett.93.218701.

18 Jörg Reichardt and Stefan Bornholdt. Statistical mechanics of community detection. *Physical Review E*, 74(016110), 2006.

19 Ron Shamir, Roded Sharan, and Dekel Tsur. Cluster graph modification problems. *Discrete Applied Mathematics*, 144:173–182, 2004.

20 V. A. Traag, P. Van Dooren, and Y. Nesterov. Narrow scope for resolution-limit-free community detection. *Physical Review E*, 84:016114, July 2011. doi:10.1103/PhysRevE.84.016114.

21 V. A. Traag, L. Waltman, and N. J. van Eck. From louvain to leiden: guaranteeing well-connected communities. *Scientific Reports*, 9(1):5233, 2019. doi:10.1038/s41598-019-41695-z.

22 Nate Veldt, David Gleich, Anthony Wirth, and James Saunderson. Metric-constrained optimization for graph clustering algorithms. *SIAM Journal on Mathematics of Data Science*, 1(2):333–355, 2019.

23 Nate Veldt, David F. Gleich, and Anthony Wirth. A correlation clustering framework for community detection. In *WWW*, pages 439–448, 2018. doi:10.1145/3178876.3186110.

24 Anthony Wirth. *Approximation algorithms for clustering*. PhD thesis, Princeton University, 2004.