Partitioning a graph into low-diameter clusters

Jack Zhang
Operations Research Center, Massachusetts Institute of Technology, MA, US
jackz111@mit.edu

Lucas Silveira
Computer Engineering Department, Military Institute of Engineering, RJ, Brazil
lucas.silveira@ime.eb.br

Hamidreza Validi
Department of Industrial, Manufacturing & Systems Engineering, Texas Tech University, TX, US
hvalidi@ttu.edu

Logan Smith
Computational Applied Mathematics & Operations Research, Rice University, TX, US
logan.smith@rice.edu

Austin Buchanan
Industrial Engineering & Management, Oklahoma State University, OK, US
buchanan@okstate.edu

Illya V. Hicks
Computational Applied Mathematics & Operations Research, Rice University, TX, US
ivhicks@rice.edu

Abstract. This paper studies the problems of partitioning the vertices of a graph $G = (V, E)$ into (or covering with) a minimum number of low-diameter clusters from the lenses of approximation algorithms and integer programming. Here, the low-diameter criterion is formalized by an s-club, which is a subset of vertices whose induced subgraph has diameter at most s. For these problems, we give $\tilde{O}(n^{1/2})$-approximation algorithms for any even integer s, generalizing a previous algorithm for the case $s = 2$. Complementing this, we show that for any $\varepsilon > 0$ the problem is NP-hard to approximate within $n^{1/2-\varepsilon}$ and $n^{1-\varepsilon}$, for each fixed even and odd integer s, respectively, suggesting an interesting contrast in approximability for even and odd values of s. Second, we develop new MIP-based heuristics (inspired by the approximation algorithms) that perform extremely well in practice, solving more than half of previous benchmark instances in less than one second. To handle the remaining instances, we propose a MIP formulation with an exponentially large class of cut-like inequalities that we solve with a branch-and-cut algorithm. With it, we tackle more benchmark instances than previous approaches and in less time.

Key words: Approximation algorithm, integer programming, low-diameter clusters, s-club, partitioning, covering, minimum k-clustering problem

1. Introduction

Clustering problems arise in diverse fields ranging from biology (Eisen et al. 1998), to dating apps (Rochat et al. 2019), to wireless sensor networks (Abbasi and Younis 2007). Ideally, there

should be few edges between clusters and many edges inside each cluster, perhaps forming a clique. This leads to problems where a graph's vertices should be partitioned into (or covered by) cliques.

The clique condition may be too restrictive, either due to data imperfections or the fact that "good" clusters need not be cliques. This has led to a plethora of clique relaxations (Pattillo et al. 2013b), such as γ-quasi-cliques (Abello et al. 2002, Pattillo et al. 2013a) which have edge density at least γ, s-defective-cliques (Yu et al. 2006) which have at most s missing edges, and s-plexes (Seidman and Foster 1978, Balasundaram et al. 2011) which permit each vertex to non-neighbor at most s other vertices. Each clique relaxation leads to an associated partitioning (or covering) problem.

In this paper, we focus on a diameter-based clique relaxation called an s-club (Mokken 1979). In a graph $G = (V, E)$, a subset of vertices $S \subseteq V$ is called an s-club if the diameter of its induced subgraph $G[S] := (S, E \cap \binom{S}{2})$ is at most s. When given a graph G and a positive integer s, the minimum s-club partitioning problem asks to partition the vertices of G into a minimum number of s-clubs. This problem is also known as the minimum s-clustering problem (Deogun et al. 1997), and its objective value, the s-club partition number, is sometimes denoted by $\mathrm{cl}_s(G)$. The covering problem is defined analogously, and we denote its objective value, the s-club cover number, by $\overline{\mathrm{cl}}_s(G)$. (This notation $\overline{\mathrm{cl}}_s(G)$ is chosen because cl is "covered" by $\overline{\ }$.) Because every partitioning is also a covering, we have the following observation.

LEMMA 1. *For any graph G and positive integer s, we have $\overline{\mathrm{cl}}_s(G) \leq \mathrm{cl}_s(G)$.*

Figure 1 shows solutions to these problems when $s = 2$ and $s = 3$ for the `karate` graph (Zachary 1977). Interestingly, when the real-life karate club split in two, each side formed a 3-club, thus giving a minimum 3-club partition. Observe that the depicted 2-clubs each have a hub vertex that dominates the others, and the 3-clubs each have a clique that dominates the others. These observations will be exploited in our approximation algorithms and MIP-based heuristics.

The minimum s-club partitioning problem has garnered interest from computer scientists and operations researchers, leading to various approximation algorithms, inapproximability results, and mixed-integer programming (MIP) models. The case $s = 1$, i.e., the minimum clique partitioning problem, is known to be NP-hard and hard to approximate. Specifically, Zuckerman (2006) shows that for every $\varepsilon > 0$ it is NP-hard to get an $n^{1-\varepsilon}$ approximation, where $n = |V|$ is the number of vertices. Meanwhile, for $\varepsilon = 0$ an $n^{1-\varepsilon}$ approximation is easy by putting each vertex in its own clique.

The situation changes for $s = 2$. Dondi et al. (2019) give an approximation algorithm for the minimum 2-club covering problem whose approximation factor is $2n^{1/2}\log^{3/2} n = \tilde{O}(n^{1/2})$. The

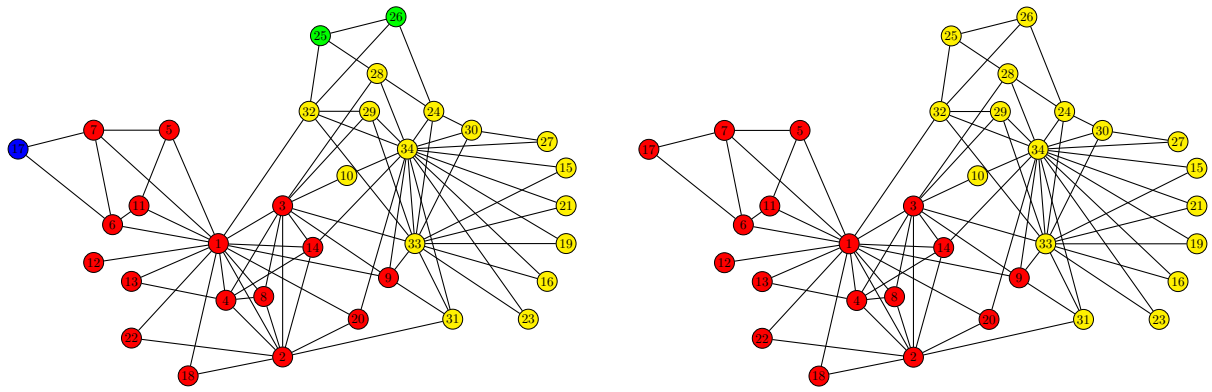

Figure 1 **Minimum 2-club and 3-club partitions for the `karate` graph G with $\mathrm{cl}_2(G) = 4$ and $\mathrm{cl}_3(G) = 2$ parts.**

idea behind this algorithm is to greedily find a small dominating set. They complement this by showing that for every $\varepsilon > 0$ it is NP-hard to get an $n^{1/2-\varepsilon}$ approximation. We extend these results to any even $s \geq 2$ (for both the covering and partitioning variants). That is, we give an approximation algorithm with factor $\tilde{O}(n^{1/2})$, and we prove that getting an $n^{1/2-\varepsilon}$ approximation is NP-hard, by carefully analyzing an earlier reduction proposed by Deogun et al. (1997) and using the newer hardness result of Zuckerman (2006). For the covering variant, we also propose a MIP-based heuristic that mimics the approximation algorithm, but uses a minimum dominating set (of an appropriately constructed graph) rather than a greedily found one. A simple post-processing procedure converts the covering to a partition of the same size.

For $s = 3$, the approximation hardness rears its head again. Dondi et al. (2019) show that it is NP-hard to get an $n^{1-\varepsilon}$ approximation. We extend this result to any odd s, again by carefully analyzing an earlier reduction of Deogun et al. (1997). Thus, the approximability of the minimum s-club partitioning (or covering) problem alternates $n, n^{1/2}, n, n^{1/2}, \ldots$ as the value $s = 1, 2, 3, 4, \ldots$ increases. For the covering variant, we also propose a MIP-based heuristic for odd s. In the $s = 3$ case, it amounts to dominating the vertices with a minimum number of cliques (which we can take to be maximal). Again, a simple post-processing procedure converts the covering to a partition of the same size.

The proposed MIP-based heuristics often perform very well in practice, finding optimal solutions in less than one second for more than half of the benchmark instances used by Yezerska et al. (2019) and three-quarters of those used by Gschwind et al. (2021). We also use a lower bound from the literature (the independence number of the s-power graph) to certify optimality of the heuristic solutions. To handle the remaining benchmark instances, we propose a MIP formulation with an exponentially large class of cut-like inequalities that we solve with a branch-and-cut algorithm. Ultimately, we solve roughly twice as many benchmark instances as the previous approach

of Gschwind et al. (2021). For instances solved by multiple approaches, our implementation is fastest, sometimes by two orders of magnitude.

Outline. Section 2 provides the relevant background. Section 3 covers approximation algorithms and hardness. It also gives the MIP-based heuristics that are inspired by the approximation algorithms. Section 4 proposes the MIP formulation and branch-and-cut algorithm. It also discusses the separation routine and formulation strength. Section 5 applies the MIP-based heuristics and branch-and-cut algorithm to benchmark instances. We conclude in Section 6.

2. Background and Literature Review

We consider a simple undirected graph $G = (V, E)$ and often let $n := |V|$ and $m := |E|$. The distance $\mathrm{dist}_G(i, j)$ between vertices i and j in graph G is the length of a shortest path between them. In this paper, all distances are hop-based, meaning that each edge has a weight of one in these path lengths. The diameter of graph G is denoted by $\mathrm{diam}(G) = \max\{\mathrm{dist}_G(i, j) \mid i, j \in V\}$. The subgraph of G induced by a subset of vertices $S \subseteq V$ is denoted by $G[S] = (S, E \cap \binom{S}{2})$, where $\binom{S}{2} = \{e \mid e \subseteq S, |e| = 2\}$ is the collection of 2-element subsets of S.

A subset of vertices $S \subseteq V$ is called an s-club if $\mathrm{diam}(G[S]) \leq s$. Equivalently, $S \subseteq V$ is an s-club if $\mathrm{dist}_{G[S]}(i, j) \leq s$ for all vertices $i, j \in S$. Meanwhile, $S \subseteq V$ is called an s-clique if $\mathrm{dist}_G(i, j) \leq s$ for all vertices $i, j \in S$, see Luce (1950). The difference is that distances are measured in $G[S]$ for s-clubs and in G for s-cliques. Generally, if a subset of vertices is an s-club then it is also an s-clique, but not vice versa. However, for $s = 1$, both definitions amount to a clique.

Clearly, s-cliques are hereditary, meaning that every subset of an s-clique is also an s-clique. Consequently, the minimum s-clique partitioning problem is equivalent to the minimum s-clique covering problem. Further, in the covering problem, it suffices to consider only those s-cliques that are inclusion-wise maximal. However, these statements are not true for s-clubs due to lack of heredity (Gschwind et al. 2021). For example, in the five-vertex cycle graph the entire vertex set forms a 2-club, but if we remove any single vertex the result is no longer a 2-club.

The s-power of graph G is denoted by $G^s = (V, E^s)$. This graph G^s has the same vertex set as G, and two distinct vertices $i, j \in V$ are connected by an edge in G^s if their distance $\mathrm{dist}_G(i, j)$ in G is at most s. We have that $S \subseteq V$ is an s-clique in G if and only if S is a clique in the s-power graph G^s. Consequently, the minimum s-clique partitioning problem in G is equivalent to the minimum clique partitioning problem in G^s. In this sense, the covering and partitioning problems for cliques and s-cliques are all essentially the same.

2.1. Lower and upper bounds

Here, we review known lower and upper bounds on $\mathrm{cl}_s(G)$ and $\overline{\mathrm{cl}}_s(G)$.

The open neighborhood of vertex $i \in V$ is denoted by $N(i) := \{j \in V \mid \{i,j\} \in E\}$, and the closed neighborhood is $N[i] := N(i) \cup \{i\}$. The closed neighborhood forms a 2-club (but not vice versa). A subset of vertices $S \subseteq V$ is called a dominating set if every vertex of the graph either belongs to S or neighbors a vertex in S. Equivalently, S is a dominating set if the closed neighborhoods $\{N[i] \mid i \in S\}$ cover V. The minimum dominating set problem, which asks for a dominating set of minimum cardinality $\gamma(G)$, is related to the minimum 2-club partitioning problem via the inequality $\mathrm{cl}_2(G) \leq \gamma(G)$, see Deogun et al. (1997). We also have that

$$\mathrm{cl}_1(G) \geq \mathrm{cl}_2(G) \geq \mathrm{cl}_3(G) \geq \cdots$$
$$\overline{\mathrm{cl}}_1(G) \geq \overline{\mathrm{cl}}_2(G) \geq \overline{\mathrm{cl}}_3(G) \geq \cdots$$

implying that the s-club partition and s-club cover numbers are also at most $\gamma(G)$ when $s \geq 2$. Trivially, we have that $\overline{\mathrm{cl}}_s(G) = \mathrm{cl}_s(G) = 1$ when the diameter of graph G is at most s. So, in line with the idea of "six degrees of separation", these covering and partitioning problems can quickly become uninteresting as s grows, which is why most papers from the literature experiment only with small values like $s \in \{2,3,4,5\}$; these values also ensure a truly "low" diameter.

We can generate a *lower bound* using a familiar graph invariant (Yezerska et al. 2019). Suppose we have a vertex subset $I \subseteq V$ with $\mathrm{dist}_G(i,j) > s$ for all distinct $i,j \in S$. In other words, I is an independent set in the s-power graph. Then, no two vertices from I can belong to the same s-club. In particular, taking I to be a maximum independent set of G^s, i.e., of cardinality $|I| = \alpha(G^s)$, we have $\alpha(G^s) \leq \overline{\mathrm{cl}}_s(G)$. When $s = 1$, we have the familiar inequality $\alpha(G) \leq \overline{\mathrm{cl}}_1(G)$ relating the independence number and the clique cover number.

Figure 2 illustrates the lower bounding idea for the `karate` graph G, showing that $\overline{\mathrm{cl}}_2(G) \geq 4$ and $\overline{\mathrm{cl}}_3(G) \geq 2$. Recall that in Figure 1, we saw partitions showing $\mathrm{cl}_2(G) \leq 4$ and $\mathrm{cl}_3(G) \leq 2$. Together, these inequalities demonstrate that $\overline{\mathrm{cl}}_2(G) = \mathrm{cl}_2(G) = 4$ and $\overline{\mathrm{cl}}_3(G) = \mathrm{cl}_3(G) = 2$.

2.2. Approximation algorithms and hardness

Deogun et al. (1997) consider approximation algorithms for the minimum s-club partitioning problem, especially for graphs with a dominating diametral path, i.e., a shortest i,j-path P whose length equals $\mathrm{diam}(G) =: d$ and whose vertices $V(P)$ form a dominating set. For such graphs, they prove that $\mathrm{cl}_s(G)$ lies between $a := (d+1)/(s+1)$ and $b := (d+1)/(s-1)$, ultimately giving

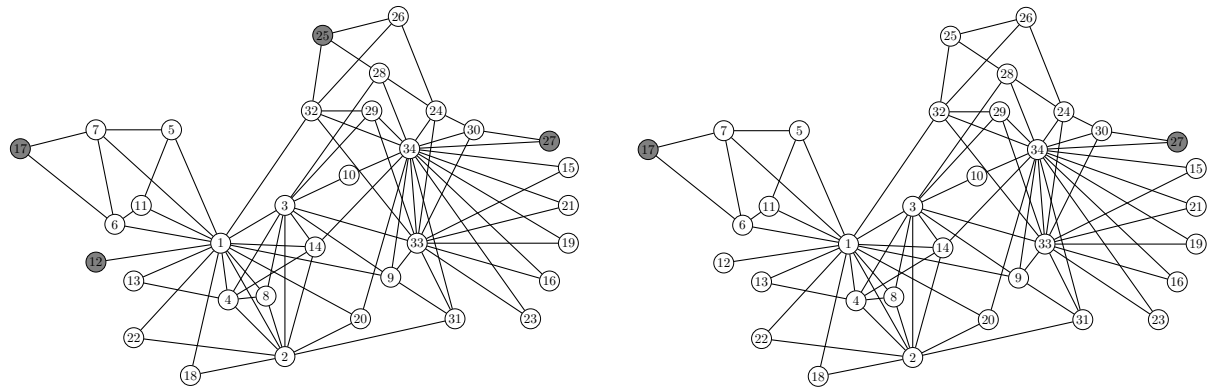

Figure 2 Maximum independent sets in power graph G^2 and G^3 with sizes 4 and 2 for the `karate` graph G.

an approximation algorithm with ratio $\lceil b \rceil / \lfloor a \rfloor$. They also show that $\mathrm{cl}_2(G) \leq \gamma(G)$, which holds at equality for strongly chordal graphs. They also provide a class of graphs $\{G_s\}_s$ on $n_s = 2s^2 + s$ vertices for which $\delta(G_s) = s$ and $\mathrm{cl}_s(G_s) = 2$, showing that the domination bound can be weak.

Deogun et al. (1997) prove that there exists an $\varepsilon > 0$ for which getting an n^ε approximation is NP-hard. For odd $s = 2r + 1$, their reduction takes a clique partitioning instance $G = (V, E)$ and attaches a pendant path of length r to each vertex. For even $s = 2r$, their reduction subdivides each original edge (thus creating $m = |E|$ new vertices), connects each pair of new vertices by a new edge, and then attaches a pendant path of length $r - 1$ to each original vertex. The hardness result that they use states that there exists an $\varepsilon > 0$ for which getting an n^ε approximation to the clique partitioning problem is NP-hard (Lund and Yannakakis 1994).

Fernandess and Malkhi (2002) consider the case of unit disk graphs and provide an approximation algorithm for the minimum s-club partitioning problem with factor $O(s)$.

We have already seen that Dondi et al. (2019) give an approximation algorithm for the covering case $s = 2$ with factor $2n^{1/2} \log^{3/2} n = \tilde{O}(n^{1/2})$. After initializing the set of uncovered vertices as $V' \leftarrow V$, this algorithm repeatedly finds a vertex $v \in V$ with maximum $|N[v] \cap V'|$, adds $N[v]$ to the collection of 2-clubs, and then updates $V' \leftarrow V' \setminus N[v]$. It terminates when all vertices have been covered. This is essentially the greedy algorithm for the dominating set problem, and the same vertex may be selected in multiple 2-clubs. However, one can easily find a 2-club partitioning of the same size by taking D as the associated dominating set (the vertices v selected by the algorithm) and re-assigning each vertex from $V \setminus D$ to an arbitrary neighbor from D. This immediately gives an approximation algorithm for the partitioning case $s = 2$ with the same factor.

Dondi et al. (2019) provide a (near) matching inapproximability result, showing that finding an $O(n^{1/2-\varepsilon})$ approximation to the covering problem is NP-hard for $s = 2$. The reduction is similar to

the $s = 2$ reduction of Deogun et al. (1997). Specifically, each edge of a clique partitioning instance is subdivided, and two new "edge" vertices are connected by an edge if the edges are incident in the input graph (whereas Deogun et al. (1997) made these "edge" vertices a clique). Additionally, Dondi et al. (2019) invoke the hardness result of Zuckerman (2006), while Deogun et al. (1997) invoke the earlier result of Lund and Yannakakis (1994).

Dondi et al. (2019) also show that finding an $O(n^{1-\varepsilon})$ approximation to the covering problem is NP-hard for $s = 3$. The reduction is exactly the same as in Deogun et al. (1997), i.e., to take a clique partitioning instance and add a pendant vertex to each original vertex. Further, Dondi et al. show that determining whether a graph can be covered by two 3-clubs (or three 2-clubs) is NP-complete.

2.3. MIPs for s-clubs

There are many existing formulations for identifying a single s-club in a graph. Each uses a binary variable x_i that equals one when vertex $i \in V$ is selected. The case $s = 2$ admits the common neighbor formulation in which selecting nonadjacent vertices $i, j \in V$ requires selecting at least one common neighbor as well, i.e., $x_i + x_j \leq 1 + \sum_{v \in N(i) \cap N(j)} x_v$. The cases where $s \geq 3$ are less straightforward to model.

Bourjolly et al. (2002) propose the chain formulation (cf. Balasundaram et al. (2005)), which uses an additional binary variable y_P for each chain (or path) P of length at most s. This leads to a model with $O(n^{s+1})$ variables and nonzeros. Wotzlaw (2014) observes that it suffices to define the y variables for the *interior* nodes of each path, which leads to a model with $O(n^{s-1})$ variables. Salemi and Buchanan (2020) further reduce the size by observing that the y variables are only needed for length-bounded connectors (i.e., without specifying the node sequence inside each path).

Veremyev and Boginski (2012), Veremyev et al. (2015) propose compact (i.e., polynomial-size) formulations with $O(sn^2)$ variables. The idea here is to define a binary variable y_{ij}^t for every pair of vertices $i, j \in V$ and each distance $t \leq s$, and to relate these variables to each other recursively.

Almeida and Carvalho (2012) consider the case $s = 3$ and compare the chain formulation, a so-called neighborhood formulation, and a node cut set formulation. The neighborhood formulation is essentially the $s = 3$ case of the length-bounded connector formulation of Salemi and Buchanan (2020). Meanwhile, the node cut set formulation is the $s = 3$ case of the length-bounded separator formulation of Salemi and Buchanan (2020).

The length-bounded separator formulation has constraints of the form $x_i + x_j \leq 1 + \sum_{v \in S} x_v$, where $S \subseteq V \setminus \{i, j\}$ is a length-s i, j-separator, i.e., $\mathrm{dist}_{G[V \setminus S]}(i, j) > s$. It suffices to impose these constraints only for inclusion-wise minimal separators. The case $s = 2$ of this formulation is

precisely the common neighbor formulation. Salemi and Buchanan (2020) show that the separation problem for the length-bounded separator constraints is polynomial-time solvable for $s \in \{2, 3, 4\}$ and NP-hard for $s \geq 5$. Salemi and Buchanan find that a branch-and-cut implementation of this formulation outperforms other formulations, often by orders of magnitude. They use a separation routine that is invoked only when LP relaxation solution is integer. It runs in time $O(nm)$ for any s and returns a violated minimal length-s i, j-separator inequality (if one exists).

2.4. MIPs for s-club partitioning and covering

Only a couple papers from the literature propose MIPs for s-club partitioning and covering. Let k be an upper bound on the s-club partition number. Yezerska et al. (2019) propose to use a binary variable x_{ij} for each vertex $i \in V$ and each part $j \in [k]$ and write:

$$\min z \tag{1a}$$

$$\sum_{j=1}^{k} x_{ij} = 1 \qquad \forall i \in V \tag{1b}$$

$$jx_{ij} \leq z \qquad \forall i \in V,\ \forall j \in [k] \tag{1c}$$

$$\{i \in V \mid x_{ij} = 1\} \text{ is an } s\text{-club} \qquad \forall j \in [k] \tag{1d}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in V,\ \forall j \in [k]. \tag{1e}$$

They impose the s-club constraints (1d) in a recursive manner, similar to Veremyev and Boginski (2012), Veremyev et al. (2015), ultimately giving a model with $O(sn^2)$ variables. With this formulation, they solve 6 of 14 real-life instances when $s = 3$, with the largest one being `adjnoun` with $n = 112$ vertices. They also propose a combinatorial branch-and-bound algorithm, which also solves 6 of 14 instances (although not the same 6), with the largest one being `jazz` with $n = 198$ vertices. Later, we will see that our MIP-based heuristic finds optimal solutions to `adjnoun` and `jazz` in a fraction of a second, and these solutions are also proven optimal in a fraction of a second by the independence number lower bound $\alpha(G^s)$.

Gschwind et al. (2021) propose a set partitioning formulation with exponentially many binary variables λ_S, one for each s-club $S \subseteq V$ from the collection of all s-clubs $\mathcal{S}$.

$$\min \sum_{S \in \mathcal{S}} \lambda_S \tag{2a}$$

$$\sum_{S \in \mathcal{S}: i \in S} \lambda_S = 1 \qquad \forall i \in V \tag{2b}$$

$$\lambda_S \in \{0, 1\} \qquad \forall S \in \mathcal{S}. \tag{2c}$$

Rather than solve this MIP directly, Gschwind et al. (2021) propose to use branch-and-price. Of their nine benchmark instances (see their Table 6), the smallest one `karate` has $n = 34$ vertices, and the largest one `celegansneural` has $n = 297$. Their Table 11 reports that 3/9 were solved when $s = 3$ (in an average of 376.9 seconds), but does not report which three. Later we will see that our MIP-based heuristic (and independence number lower bound) solve 5/9 instances, each in a fraction of a second. Our branch-and-cut implementation solves all 9 instances, each in under 75 seconds. We also solve the much larger instance `netscience` ($n = 1589$) in two seconds.

3. Approximation Algorithms, Hardness, and MIP Heuristics

This section proposes approximation algorithms for the s-club partitioning and covering problems. We complement them with approximation hardness results. We also propose MIP-based heuristics that are inspired by the approximation algorithms.

3.1. Approximation algorithm for even s

Here, we generalize the approximation algorithm of Dondi et al. (2019) for the 2-club covering problem so that it works for all even values of s and for both the partitioning and covering variants. The approximation factor $\tilde{O}(n^{1/2})$ remains the same. Pseudocode is given in Algorithm 1. We will assume that $n := |V| \geq 3$, since otherwise both problems are easy.

Algorithm 1 Approximate-Club-Partitioning-Even(G, s)

create $(s/2)$-power graph $H := G^{s/2}$
initialize $U \leftarrow V$ and $D \leftarrow \emptyset$
while $U \neq \emptyset$ **do**
    pick a vertex $v \in V$ with maximum $|N_H[v] \cap U|$
    $D \leftarrow D \cup \{v\}$
    $U \leftarrow U \setminus N_H[v]$
create a new auxiliary vertex r
create $G' = (V', E')$ with $V' = V \cup \{r\}$ and $E' = E \cup \{\{r, i\} \mid i \in D\}$
find a BFS tree of G' rooted at r and call it T
let $T_1, T_2, \ldots, T_{|D|}$ be the components of $T - \{r\}$
return $V(T_1), V(T_2), \ldots, V(T_{|D|})$

The idea behind the algorithm is to greedily find a dominating set $D \subseteq V$ in the $(s/2)$-power graph and assign the other vertices $V \setminus D$ to them. Thus, when $s = 2$, the $(s/2)$-power graph is the same as G, and the algorithm essentially reduces to that of Dondi et al. (2019).

Lines 2–6 are the greedy algorithm for dominating set, applied to the power graph $H := G^{s/2}$. Let D^* be a minimum dominating set of graph H, and let D be the set of vertices picked by Algorithm 1. We have the following inequalities, where the first inequality holds by the greedy dominating set approximation (Johnson 1974) and the second inequality holds when $n \geq 3$.

$$|D| \leq |D^*|(1 + \ln n) \leq 2|D^*| \ln n. \tag{3}$$

Lines 7–11 find s-clubs that are "centered" at the dominating set vertices. The diameter of each induced subgraph $G[V(T_j)]$ is at most s because it contains a rooted tree T_j whose depth is at most $s/2$. The algorithm's running time is $O(nm)$, with the most time-consuming step being the creation of the power graph H. Like Dondi et al. (2019), we require the following lemma.

LEMMA 2 **(Desormeaux et al. (2014))**. *If an n-vertex graph $G = (V, E)$ has diameter two, then it admits a dominating set of size at most $1 + \sqrt{n \ln n}$.*

LEMMA 3. *Let s be even. If S is an s-club in G, then S is a 2-club in $H := G^{s/2}$.*

Proof of Lemma 3. Let u and v be distinct vertices from the s-club S. By definition of s-club, we have $\mathrm{dist}_{G[S]}(u, v) \leq s$. We are to show that $\mathrm{dist}_{H[S]}(u, v) \leq 2$. In the first case, suppose that $\mathrm{dist}_{G[S]}(u, v) \leq s/2$. This implies that $\mathrm{dist}_G(u, v) \leq s/2$ as well, so $\{u, v\}$ is an edge in H and thus $\mathrm{dist}_{H[S]}(u, v) = 1$. In the other case, where $s/2 < \mathrm{dist}_{G[S]}(u, v) \leq s$, there is an intermediate vertex $w \in S$ for which $\mathrm{dist}_{G[S]}(u, w) = s/2$ and $\mathrm{dist}_{G[S]}(w, v) \leq s/2$. This implies that $\mathrm{dist}_G(u, w) \leq s/2$ and $\mathrm{dist}_G(w, v) \leq s/2$, so $\{u, w\}$ and $\{w, v\}$ are edges of H and $\mathrm{dist}_{H[S]}(u, v) \leq 1 + 1 = 2$. □

THEOREM 1. *Algorithm 1 provides an $\tilde{O}(n^{1/2})$-approximation for the minimum s-club partitioning and covering problems when s is even.*

Proof of Theorem 1. Let $\mathcal{S}^*$ be a minimum s-club *covering* of G and define $\mathrm{OPT} := |\mathcal{S}^*|$. Meanwhile, the algorithm returns a *partition* with $|D|$ different s-clubs. To prove the approximation factor for both the partitioning and covering problems, it suffices to show that $|D| \leq \tilde{O}(\sqrt{n})\,\mathrm{OPT}$.

Let D^* be a minimum dominating set of power graph $H = G^{s/2}$. By inequality (3), we have

$$|D| \leq 2|D^*| \ln n. \tag{4}$$

By Lemma 3, each s-club S in the covering $\mathcal{S}^*$ is a 2-club in H. By Lemma 2, $H[S]$ has a dominating set D_S of size at most $1+\sqrt{|S|\ln(|S|)} \le 2\sqrt{n\ln n}$. Taking their union $D' := \cup_{S\in\mathcal{S}^*} D_S$ gives a dominating set for H whose size $|D'|$ satisfies

$$|D^*| \le |D'| \le \sum_{S\in\mathcal{S}*} |D_S| \le \mathrm{OPT}(2\sqrt{n\ln n}), \tag{5}$$

with the first inequality holding because D^* is minimum. Putting inequalities (4) and (5) together,

$$|D| \le 2|D^*|\ln n \le 4(\ln n)\sqrt{n\ln n}\,\mathrm{OPT} = \tilde{O}(\sqrt{n})\,\mathrm{OPT}\,. \quad \square$$

3.2. MIP-based heuristic for even s

To get better practical performance, we find a *minimum* dominating set of $H = G^{s/2}$ in lines 2–6 rather than a greedy one. To accomplish this, we solve the integer program (IP)

$$\min_{z \text{ binary}} \left\{ \sum_{i\in V} z_i \;\middle|\; \sum_{j\in N_H[i]} z_j \ge 1 \;\forall i \in V \right\}. \tag{6}$$

While the minimum dominating set problem is NP-hard, it can often be solved quickly in practice, including for the benchmark instances used by Yezerska et al. (2019) and Gschwind et al. (2021). This gives a better approximation factor, although the procedure is no longer polynomial.

REMARK 1. The approximation factor of Algorithm 1 improves to $2\sqrt{n\ln n}$ if the set D in lines 2-6 is taken as a minimum dominating set, as inequality (5) shows.

3.3. Approximation hardness for even s

We complement the approximation algorithm with an inapproximability result, extending a result of Dondi et al. (2019) for 2-club covering to all even values of s for both covering and partitioning.

THEOREM 2. *Let $s \ge 2$ be an even integer and $\varepsilon > 0$. It is NP-hard to get an $n^{1/2-\varepsilon}$ approximation to the s-club partitioning (or covering) problem.*

Proof of Theorem 2. Zuckerman (2006) shows that, for any $\varepsilon > 0$, it is NP-hard to get an $n^{1-\varepsilon}$ approximation for the clique partitioning problem. So, consider an instance $G' = (V', E')$ of the clique partitioning problem with $n' = |V'|$ vertices and $m' = |E'|$ edges. We may assume that G' has at least one edge and is connected, in which case $m' \ge 1$ and $m' \ge n' - 1$ so $2m' \ge n'$. Also, if the number of vertices n' is smaller than $n'_0 := (s+1)^{-1+1/2\varepsilon}$, then the instance is easy as n'_0 is a constant. So, we may also assume that $n' \ge n'_0$.

We proceed by using the construction of Deogun et al. (1997). That is, letting $r = s/2$, we subdivide each edge of G' (thus creating $m' = |E'|$ new vertices), connect each pair of these new

"edge" vertices by a new edge (i.e., making them a clique), and then attach a pendant path of length $r-1$ to each original vertex. The number of vertices in this new graph $G=(V,E)$ is $n=n'+m'+(r-1)n'$ or equivalently $n=(s/2)n'+m'$.

Suppose that we have an $n^{1/2-\varepsilon}$ approximation algorithm for s-club partitioning in G. That is, it returns a feasible solution with objective $\mathrm{ALG} \le n^{1/2-\varepsilon}\,\mathrm{OPT}$. We show that it can be used to give an $(n')^{1-\varepsilon}$ approximation algorithm for clique partitioning in G', i.e., returning a feasible solution with objective $\mathrm{ALG}' \le (n')^{1-\varepsilon}\,\mathrm{OPT}'$.

We observe that $\mathrm{OPT} \le \mathrm{OPT}'$ because a clique partitioning $\mathcal{Q}'$ of G' can be converted into an s-club partitioning of G of the same size. Specifically, for each clique $Q' \in \mathcal{Q}'$, we construct an s-club Q by taking the original vertices Q', the pendant paths that are connected to them, and any of the new "edge" vertices that lie between the original clique vertices. After this process, there will be some remaining "edge" vertices that neighbor two different cliques; arbitrarily assign them to either side. This gives the s-club partition $\mathcal{Q}$ of G.

After applying the supposed $n^{1/2-\varepsilon}$ approximation algorithm to get an s-club partitioning $\mathcal{S}$ of G, we can recover a clique partitioning $\mathcal{S}'$ of G' of the same size. Specifically, for each s-club $S \in \mathcal{S}$, construct a clique $S' = S \cap V'$ by taking just its original vertices (i.e., ignoring the "edge" vertices and pendant path vertices). Some S' may be empty. This shows $\mathrm{ALG}' = \mathrm{ALG}$. Then,

$$\begin{aligned}
\mathrm{ALG}' = \mathrm{ALG} \le n^{1/2-\varepsilon}\,\mathrm{OPT} &= ((s/2)n'+m')^{1/2-\varepsilon}\,\mathrm{OPT} && \text{(7a)}\\
&\le ((s/2)n'+m')^{1/2-\varepsilon}\,\mathrm{OPT}' && \text{(7b)}\\
&\le (sm'+m')^{1/2-\varepsilon}\,\mathrm{OPT}' && \text{(7c)}\\
&= (s+1)^{1/2-\varepsilon} \times (m')^{1/2-\varepsilon}\,\mathrm{OPT}' && \text{(7d)}\\
&\le (s+1)^{1/2-\varepsilon} \times (n')^{1-2\varepsilon}\,\mathrm{OPT}' && \text{(7e)}\\
&= (n'_0)^{\varepsilon} \times (n')^{1-2\varepsilon}\,\mathrm{OPT}' && \text{(7f)}\\
&\le (n')^{\varepsilon} \times (n')^{1-2\varepsilon}\,\mathrm{OPT}' = (n')^{1-\varepsilon}\,\mathrm{OPT}'. && \text{(7g)}
\end{aligned}$$

The proof is essentially the same for the covering case. □

3.4. Approximation hardness for odd s

We extend a result of Dondi et al. (2019) for 3-club covering to all odd values of s for both covering and partitioning. For this reason, it seems hopeless to search for polynomial-time approximation algorithms with nontrivial factors when s is odd.

THEOREM 3. *Let $s \geq 3$ be an odd integer and $\varepsilon > 0$. It is NP-hard to get an $n^{1-\varepsilon}$ approximation to the s-club partitioning (or covering) problem.*

Proof of Theorem 3. Zuckerman (2006) shows that, for any $\varepsilon' > 0$, it is NP-hard to get an $n^{1-\varepsilon'}$ approximation for the clique partitioning problem. In particular, it is hard for $\varepsilon' = \varepsilon/2 > 0$. So, consider an instance $G' = (V', E')$ of the clique partitioning problem with $n' = |V'|$ vertices and $m' = |E'|$ edges. We may assume that G' has at least one edge and is connected, in which case $m' \geq 1$ and $m' \geq n' - 1$ so $2m' \geq n'$. Also, if the number of vertices n' is smaller than $n'_0 := (\frac{s+1}{2})^{-2+1/\varepsilon'}$, then the instance is easy as n'_0 is a constant. So, we may also assume that $n' \geq n'_0$.

We proceed by using the (other) construction of Deogun et al. (1997). That is, let $r = (s-1)/2$ and attach a pendant path of length r to each original vertex. The number of vertices in this new graph $G = (V, E)$ is $n = n' + rn'$ or equivalently $n = n'(s+1)/2$.

Suppose that we have an $n^{1-\varepsilon}$ approximation algorithm for s-club partitioning in G. That is, it returns a feasible solution with objective $\mathrm{ALG} \leq n^{1-\varepsilon}\,\mathrm{OPT}$. We show that it can be used to give an $(n')^{1-\varepsilon'}$ approximation algorithm for clique partitioning in G', i.e., returning a feasible solution with objective $\mathrm{ALG}' \leq (n')^{1-\varepsilon'}\,\mathrm{OPT}'$.

We observe that $\mathrm{OPT} \leq \mathrm{OPT}'$ because a clique partitioning $\mathcal{Q}'$ of G' can be converted into an s-club partitioning of G of the same size. Specifically, for each clique $Q' \in \mathcal{Q}'$, we construct an s-club Q by taking the original vertices Q' and adding the pendant paths that are connected to them. This gives the s-club partition $\mathcal{Q}$ of G.

After applying the supposed $n^{1-\varepsilon}$ approximation algorithm to get an s-club partitioning $\mathcal{S}$ of G, we can recover a clique partitioning $\mathcal{S}'$ of G' of the same size. Specifically, for each s-club $S \in \mathcal{S}$, construct a clique $S' = S \cap V'$ by taking just its original vertices (i.e., ignoring the pendant path vertices). Some S' may be empty. This shows $\mathrm{ALG}' = \mathrm{ALG}$. Then,

$$\mathrm{ALG}' = \mathrm{ALG} \leq n^{1-\varepsilon}\,\mathrm{OPT} = \left(\frac{s+1}{2} \times n'\right)^{1-2\varepsilon'} \mathrm{OPT} \tag{8a}$$

$$\leq \left(\frac{s+1}{2} \times n'\right)^{1-2\varepsilon'} \mathrm{OPT}' \tag{8b}$$

$$= \left(\frac{s+1}{2}\right)^{1-2\varepsilon'} \times (n')^{1-2\varepsilon'}\,\mathrm{OPT}' \tag{8c}$$

$$= (n'_0)^{\varepsilon'} \times (n')^{1-2\varepsilon'}\,\mathrm{OPT}' \tag{8d}$$

$$\leq (n')^{\varepsilon'} \times (n')^{1-2\varepsilon'}\,\mathrm{OPT}' = (n')^{1-\varepsilon'}\,\mathrm{OPT}'\,. \tag{8e}$$

The proof is essentially the same for the covering case. □

3.5. MIP-based heuristic for odd s

The approximation algorithm of Dondi et al. (2019) seeks a small dominating set of G, with the idea that these vertices can be used as centers of the 2-clubs. Algorithm 1 generalized this to all even s by seeking a dominating set of the power graph $G^{s/2}$.

In this section, we extend the idea to odd s. Here, each s-club will be centered at a *clique*, and these cliques should dominate the other vertices from a distance—in power graph $H = G^{(s-1)/2}$.

First, we propose a greedy version that repeatedly identifies a (maximal) clique $Q \subseteq V$ whose closed neighborhood $N[Q] = \cup_{i \in Q} N[i]$ covers the most uncovered vertices in H. This repeats until all vertices have been covered. The resulting collection of maximal cliques $\mathcal{Q}$ may include the same vertex multiple times, so we remove any such duplicates. For example, if the selected maximal cliques are $Q_1, Q_2, Q_3, \ldots, Q_q$, we may pick $Q'_1 = Q_1, Q'_2 = Q_2 \setminus Q_1, Q'_3 = Q_3 \setminus (Q_1 \cup Q_2), \ldots, Q'_q = Q_q \setminus (Q_1 \cup Q_2 \cup \cdots \cup Q_{q-1})$ and then update $\mathcal{Q}$ to be this new collection of cliques $\{Q'_1, Q'_2, \ldots, Q'_q\}$. The other vertices—that do not belong to any chosen clique—are assigned to the cliques to get an s-club partitioning. Pseudocode is provided in Algorithm 2.

Algorithm 2 Heuristic-Club-Partitioning-Odd(G, s)

create $(s-1)/2$-power graph $H := G^{(s-1)/2}$
initialize $U \leftarrow V$ and $\mathcal{Q} \leftarrow \{\}$
while $U \neq \emptyset$ **do**
    pick a maximal clique $Q \subseteq V$ in G with maximum $|N_H[Q] \cap U|$
    $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{Q\}$
    $U \leftarrow U \setminus N_H[Q]$
remove duplicates from $\mathcal{Q}$ (e.g., keep a vertex only in its earliest clique)
create a new auxiliary vertex r and a new vertex v_Q for each clique $Q \in \mathcal{Q}$
let $V_Q = \{v_Q \mid Q \in \mathcal{Q}\}$ and $E_Q = \{\{v_Q, v\} \mid Q \in \mathcal{Q},\ v \in Q\}$
create $G' = (V', E')$ with $V' = V \cup V_Q \cup \{r\}$ and $E' = E \cup E_Q \cup \{\{r, v_Q\} \mid Q \in \mathcal{Q}\}$
find a BFS tree of G' rooted at r and call it T
let $T_1, T_2, \ldots, T_{|\mathcal{Q}|}$ be the components of $T - \{r\}$
return $V(T_1), V(T_2), \ldots, V(T_{|\mathcal{Q}|})$ (omitting the vertices from V_Q)

To get better practical performance, we can take an exact approach to select cliques rather than being greedy. Namely, for each maximal clique Q, we introduce a binary variable z_Q and write:

$$\min \sum_{Q} z_Q \tag{9a}$$

$$\sum_{Q \,:\, i \in N_H[Q]} z_Q \geq 1 \qquad \forall i \in V \tag{9b}$$

$$z_Q \in \{0,1\} \qquad \forall Q. \tag{9c}$$

Generally, a graph may have up to $3^{n/3}$ maximal cliques (Miller and Muller 1960, Moon and Moser 1965), possibly making this model quite large. However, many real-life graphs are very sparse with relatively few maximal cliques. They can be enumerated with the well-known algorithm of Bron and Kerbosch (1973), which is implemented in the Python package `NetworkX` (Hagberg et al. 2008), or with the more recent algorithm of Eppstein et al. (2013) that is tailored for graphs that have low degeneracy (a measure of density). We will see that this approach works quite well for the benchmark instances used by Yezerska et al. (2019) and Gschwind et al. (2021).

4. Integer Programming Formulations

This section provides an IP formulation for the minimum s-club partitioning problem. Like the formulation of Yezerska et al. (2019), it relies on an upper bound k on $\mathrm{cl}_s(G)$ and uses a binary variable x_{ij} for each vertex $i \in V$ and each part $j \in [k]$. However, rather than simply using a single variable z to count the number of parts that are being used (and writing $jx_{ij} \leq z$), we introduce a variable y_j for each part $j \in [k]$ and minimize their sum. This leads to the following formulation, which turns out to be stronger. We also break some symmetry with constraints (10d). To model the covering problem, change the sense of the assignment constraints (10b) from $=$ to $\geq$.

$$\min \sum_{j=1}^{k} y_j \tag{10a}$$

$$\sum_{j=1}^{k} x_{ij} = 1 \qquad \forall i \in V \tag{10b}$$

$$x_{ij} \leq y_j \qquad \forall i \in V,\ \forall j \in [k] \tag{10c}$$

$$y_k \leq y_{k-1} \leq \cdots \leq y_1 \leq 1 \tag{10d}$$

diameter-bounding constraints (10e)

$$x_{ij} \in \{0,1\} \qquad \forall i \in V,\ \forall j \in [k]. \tag{10f}$$

For now, we write the diameter-bounding constraints in a generic form (10e) so that we can compare the "cores" of the models. Specifically, consider a fractional point $(\hat{x}, \hat{y})$ that satisfies constraints (10c). Then, the similar point $(\hat{x}, \hat{z})$ with $\hat{z} := \sum_{j\in[k]} \hat{y}_j$ satisfies the following big-M-style constraints from Yezerska et al. (2019) and has the same LP objective value.

$$jx_{ij} \le z \qquad \forall i \in V,\ \forall j \in [k].$$

Moreover, the reverse is not always true. In this sense, the constraints (10c) are stronger (and add little size to the formulation), and we will see that they perform better in practice.

To ensure that each part has diameter at most s, we use length-bounded separator constraints. This choice is informed by previous success with these constraints for other diameter-constrained problems (Salemi and Buchanan 2020, Validi and Buchanan 2020, Lu et al. 2022, Validi et al. 2022). Recall that if $a, b \in V$ are distinct, nonadjacent vertices, then $S \subseteq V \setminus \{a, b\}$ is called a length-s a, b-separator if $\mathrm{dist}_{G[V\setminus S]}(a, b) > s$. For any such a, b, S, we can write the constraints

$$x_{aj} + x_{bj} \le 1 + \sum_{i\in S} x_{ij} \qquad \forall j \in [k].$$

It follows from Salemi and Buchanan (2020) that the collection of all such inequalities will ensure that each part is an s-club. Generally, there are exponentially many such inequalities, although relatively few are needed to prove optimality in practice. Like Salemi and Buchanan (2020), we add them on-the-fly in a branch-and-cut implementation, separating only integer infeasible points x^*. In this case, separation can be performed in time $O(nm)$, as follows.

- for $j \in [k]$ do
  - —construct graph $G[V_j]$ where $V_j = \{i \in V \mid x^*_{ij} = 1\}$
  - —if $\mathrm{diam}(G[V_j]) \le s$ then continue
  - —let $a, b \in V_j$ be vertices with $\mathrm{dist}_{G[V_j]}(a, b) > s$
  - —observe that $S' = V \setminus V_j$ is a length-s a, b-separator
  - —$S \leftarrow \texttt{minimalize}(S')$
  - —add cut $x_{aj} + x_{bj} \le 1 + \sum_{i\in S} x_{ij}$ and exit

Here, `minimalize` is a subroutine proposed by Salemi and Buchanan (2020) that takes a length-s a, b-separator S' and returns a minimal length-s a, b-separator S in time $O(nm)$. For us to achieve a total running time of $O(nm)$, it should be observed that the diameters of *all* graphs $G[V_j] = (V_j, E_j)$ can be computed in a total of time $\sum_{j=1}^{k} O(|V_j||E_j|) = \sum_{j=1}^{k} O(|V_j|m) = O(nm)$. In practice, we

may choose to add multiple cuts, one for each part $j \in [k]$ for which V_j is not an s-club. This can be accomplished by removing the "exit" command, in which case the total running time is $O(knm)$.

To speed up the computation, we fix some variables to zero or one. Specifically, let $I = \{i_1, i_2, \ldots, i_{\alpha(H)}\}$ be maximum independent set of the s-power graph $H = G^s$. Since no two vertices from I can belong to the same s-club, it is safe to fix them into different parts, e.g., to fix i_j into part j. This breaks a considerable amount of formulation symmetry and also allows us to fix $x_{vj} = 0$ for all vertices v that are "far" from i_j (i.e., with $\mathrm{dist}_G(i_j, v) > s$). These techniques apply to both partitioning and covering.

5. Computational Experiments

To evaluate our proposed methods (and to compare them with previous ones), we experiment on the benchmark instances used by Yezerska et al. (2019) and Gschwind et al. (2021). Table 1 summarizes these 14 instances, showing that Gschwind et al. (2021) considered the 9 smallest instances from Yezerska et al. (2019). Like Gschwind et al. (2021), we consider $s \in \{2, 3, 4, 5\}$.

Our experiments are conducted on a machine running Windows 11 Enterprise with an Intel Core i9-13900 processor (2.00 GHz base, 5.2 GHz turbo) and 32 GB RAM. The code is written in Python, handles graphs with NetworkX (Hagberg et al. 2008) (including to enumerate maximal cliques), and solves MIPs using Gurobi v11.0.3. Our code and data are available at https://github.com/JackDaihanZhang/Partitioning-a-graph-into-low-diameter-clusters.

5.1. Evaluating the lower and upper bounds

In this section, we evaluate the performance of the lower and upper bounds for both the covering and partitioning problems. In particular, the lower bound (for both problem variants) is the independence number of the s-power graph $\alpha(G^s)$.

For even s, we consider two upper bounds: the approximation algorithm (Algorithm 1) and the associated MIP-based heuristic that solves the dominating set problem (6) exactly. Results for $s \in \{2, 4\}$ are provided in Table 2. We exclude the trivial instances that have $\mathrm{diam}(G) \leq s$ and thus $\mathrm{cl}(G) = \overline{\mathrm{cl}}(G) = 1$. For odd s, we also consider two upper bounds: the greedy heuristic (Algorithm 2) and the associated MIP-based heuristic that solves the dominating cliques problem (9) exactly. Results for $s \in \{3, 5\}$ are provided in Table 3.

Inspecting the tables, we see that both the lower and upper bounds can be computed quite quickly, usually in a small fraction of a second. Moreover, the lower bound equals the upper bound coming from the MIP-based heuristic for more than 55% of the instances (26/47). As expected, the

Table 1 The benchmark instances that we use. We report the number of vertices n, edges m, maximal cliques c, and diameter diam. For the disconnected instances (shown by *), the largest diameter of any component is reported. The names `celegansneural` and `celegans_metabolic` are shortened to CN and CM.

					Used by	
Graph	n	m	c	diam	Yezerska et al.	Gschwind et al.
karate	34	78	36	5	✓	✓
chesapeake	39	170	139	3	✓	✓
dolphins	62	159	84	8	✓	✓
lesmis	77	254	59	5	✓	✓
polbooks	105	441	199	7	✓	✓
adjnoun	112	425	303	5	✓	✓
football	115	613	281	4	✓	✓
jazz	198	2,742	746	6	✓	✓
CN	297	2,148	1,386	5	✓	✓
CM	453	2,025	668	7	✓	✗
netscience	1,589	2,742	741	17*	✓	✗
polblogs	1,490	16,715	49,884	8*	✓	✗
email	1,133	5,451	3,267	8	✓	✗
data	2,851	15,093	11,928	79	✓	✗

approximation algorithm and greedy heuristic give worse solutions than the MIP-based heuristic, only matching the lower bound for 21% (10/47) of the instances. For example, for `netscience` and $s = 4$, the approximation algorithm gives a solution with objective 421, while the MIP-based heuristic gives an optimal solution with objective 418. The difference is even more noticeable for `data` and $s = 2$, where the approximation algorithm gives 358, while the MIP-based heuristic gives 286. Similarly, for `data` and $s = 5$, the greedy heuristic gives a solution with objective 89, while the MIP-based heuristic gives 68, nearly matching the lower bound of 67.

We conclude that the MIP-based heuristic is a solid choice, given that it is fast and gives optimal or near-optimal solutions for most benchmark instances used by Yezerska et al. (2019) and Gschwind et al. (2021).

5.2. Evaluating the branch-and-cut implementation

In this section, we evaluate the performance of the branch-and-cut implementation as a whole. If the graph's diameter is at most s, then an optimal solution has just one s-club consisting of all

Table 2 Lower and upper bounds for the minimum s-club partitioning and covering problems for even s. The lower bounds LB are $\alpha(G^s)$. The upper bounds UB come from the approximation algorithm (APX) and the MIP-based heuristic (MIP) under a 60-second time limit (TL). Dashes indicate $\mathrm{diam}(G) \le s$, i.e., $\mathrm{cl}(G) = \overline{\mathrm{cl}}(G) = 1$.

	$s=2$						$s=4$					
Graph	LB	time	APX	time	MIP	time	LB	time	APX	time	MIP	time
karate	4	0.01	4	0.00	4	0.00	2	0.00	2	0.00	2	0.00
chesapeake	3	0.02	3	0.00	3	0.00	-	-	-	-	-	-
dolphins	13	0.02	16	0.00	14	0.00	4	0.02	5	0.00	4	0.02
lesmis	10	0.02	10	0.00	10	0.00	2	0.02	2	0.02	2	0.01
polbooks	12	0.02	14	0.00	13	0.02	2	0.03	2	0.03	2	0.03
adjnoun	18	0.02	18	0.02	18	0.00	3	0.05	4	0.05	3	0.04
football	7	0.13	14	0.02	12	0.20	-	-	-	-	-	-
jazz	13	0.11	14	0.08	13	0.06	4	0.14	4	0.42	4	0.31
CN	14	0.36	19	0.05	16	0.05	2	0.32	3	0.42	3	0.41
CM	29	0.36	30	0.11	29	0.08	7	0.74	8	0.80	7	0.58
netscience	477	0.24	477	0.11	477	0.34	418	0.44	421	0.23	418	0.33
polblogs	395	4.12	396	1.09	395	0.55	280	7.02	284	8.42	281	6.00
email	209	1.43	222	0.58	210	0.13	37	15.20	51	1.04	40	0.69
data	250	TL	358	2.23	286	TL	91	13.94	123	1.80	96	18.09

vertices. Otherwise, it computes a maximum independent set $I \subseteq V$ of G^s and uses this as a lower bound. Then, it applies the MIP-based heuristic to get an upper bound. If these bounds match, then it terminates. Otherwise, it fixes the vertices of I to different parts and solves the integer programming formulation (10) by separating the length-s i,j-separator inequalities on-the-fly. The formulation is warm-started with the MIP-based heuristic solution.

Tables 4, 5, 6, 7 summarize the results for $s \in \{2,3,4,5\}$ for both the partitioning and covering variants. To isolate the effects of our proposed formulation, we compare our implementation with an alternative one that instead uses the integer programming formulation of Yezerska et al. (2019). Both use a one-hour time limit. We see that our implementation solves most instances in a short amount of time, with only a few exceptions for each value of s. These same challenging instances also cause troubles for the formulation of Yezerska et al. (2019). Our formulation solves several more instances, namely `football`, `CN`, and `email` for $s=2$; `CN` for $s=3$; and `data` for $s=5$.

Table 3 Lower and upper bounds for the minimum s-club partitioning and covering problems for odd s. The lower bounds LB are $\alpha(G^s)$. The upper bounds UB come from the greedy heuristic (GRE) and the MIP-based heuristic (MIP) under a 60-second time limit (TL). Dashes indicate $\mathrm{diam}(G) \leq s$, i.e., $\mathrm{cl}(G) = \overline{\mathrm{cl}}(G) = 1$.

	$s=3$						$s=5$					
Graph	LB	time	GRE	time	MIP	time	LB	time	GRE	time	MIP	time
karate	2	0.01	3	0.00	2	0.00	-	-	-	-	-	-
chesapeake	-	-	-	-	-	-	-	-	-	-	-	-
dolphins	6	0.01	8	0.01	7	0.01	2	0.02	3	0.01	2	0.01
lesmis	6	0.02	7	0.01	6	0.01	-	-	-	-	-	-
polbooks	5	0.03	7	0.02	6	0.02	2	0.04	3	0.03	2	0.04
adjnoun	8	0.05	11	0.03	8	0.02	-	-	-	-	-	-
football	3	0.07	7	0.03	6	0.17	-	-	-	-	-	-
jazz	6	0.14	6	0.24	6	0.17	2	0.13	3	0.42	2	0.50
CN	5	0.36	9	0.25	7	0.25	-	-	-	-	-	-
CM	15	0.70	17	0.45	15	0.23	5	0.72	6	1.12	5	0.86
netscience	444	0.08	449	0.14	445	0.22	410	0.27	413	0.19	410	0.35
polblogs	311	13.99	326	TL	314	TL	272	6.39	273	42.90	273	TL
email	93	29.80	120	2.62	97	0.36	16	8.41	25	2.89	20	3.78
data	127	12.79	183	10.26	139	TL	67	28.03	89	7.71	68	3.72

Table 4 Computational results for the minimum 2-club partitioning (or covering) problem using our MIP and that of Yezerska et al. (2019) under a 3,600-second time limit (TL). MEM stands for memory crash.

	Our MIP				Yezerska et al. (2019)			
Graph	partition	time	cover	time	partition	time	cover	time
dolphins	13	0.11	13	0.09	13	1.34	13	1.37
polbooks	12	0.62	12	0.84	12	4.19	12	3.84
football	10	610.72	10	2,426.76	[7, 12]	TL	[7, 12]	TL
CN	15	52.84	15	50.12	[14, 16]	TL	[14, 16]	TL
email	209	56.30	209	67.88	MEM	MEM	MEM	MEM
data	[251, 286]	TL	[251, 286]	TL	MEM	MEM	MEM	MEM

We would have liked to compare our results with the branch-and-price algorithm of Gschwind et al. (2021), but the associated codes have not been shared publicly, and the authors declined to

Table 5 Computational results for the minimum 3-club partitioning (or covering) problem using our MIP and that of Yezerska et al. (2019) under a 3,600-second time limit (TL). MEM stands for memory crash.

	Our MIP				Yezerska et al. (2019)			
Graph	partition	time	cover	time	partition	time	cover	time
dolphins	6	0.14	6	0.17	6	1.69	6	1.55
polbooks	5	1.03	5	1.16	5	9.91	5	11.36
football	3	7.03	3	6.27	3	558.28	3	388.17
CN	5	74.19	5	36.63	[5, 7]	TL	[5, 7]	TL
netscience	444	2.14	444	2.17	444	71.29	444	81.89
polblogs	[311, 314]	TL	[312, 314]	TL	MEM	MEM	MEM	MEM
email	[94, 97]	TL	[94, 97]	TL	MEM	MEM	MEM	MEM
data	[128, 139]	TL	[128, 139]	TL	MEM	MEM	MEM	MEM

Table 6 Computational results for the minimum 4-club partitioning (or covering) problem using our MIP and that of Yezerska et al. (2019) under a 3,600-second time limit (TL). MEM stands for memory crash.

	Our MIP				Yezerska et al. (2019)			
Graph	partition	time	cover	time	partition	time	cover	time
CN	2	18.45	2	6.80	2	1,580.87	2	2,365.68
polblogs	[280, 281]	TL	[280, 281]	TL	MEM	MEM	MEM	MEM
email	[37, 40]	TL	[38, 40]	TL	MEM	MEM	MEM	MEM
data	[92, 96]	TL	[91, 96]	TL	MEM	MEM	MEM	MEM

Table 7 Computational results for the minimum 5-club partitioning (or covering) problem using our MIP and that of Yezerska et al. (2019) under a 3,600-second time limit (TL). MEM stands for memory crash.

	Our MIP				Yezerska et al. (2019)			
Graph	partition	time	cover	time	partition	time	cover	time
polblogs	[272, 273]	TL	272	566.08	MEM	MEM	MEM	MEM
email	[16, 20]	TL	[16, 20]	TL	MEM	MEM	MEM	MEM
data	68	2,676.61	[67, 68]	TL	MEM	MEM	MEM	MEM

send them to us when we asked. So, Table 8 summarizes the performance of our approach and that reported by Gschwind et al. (2021), both using the same 10-minute time limit. We see that our implementation solves all 36 instances of the partitioning variant, while Gschwind et al. (2021)

solve 19/36. Further, by referring back to Tables 2 and 3, we see that our initial lower and upper bounds (along with the "is $\mathrm{diam}(G) \leq s$?" check) suffice to solve 27/36 instances, without even invoking the branch-and-cut algorithm. Meanwhile, for the covering variant, both approaches work well; we solve all 36 instances, while Gschwind et al. (2021) solve 35.

Table 8 Number of instances solved across the 9 benchmark instances of Gschwind et al. (2021) of the minimum s-club partitioning (or covering) problem within a 600-second time limit.

	Our approach		Gschwind et al. (2021)	
s	partition	cover	partition	cover
2	9	9	4	8
3	9	9	3	9
4	9	9	4	9
5	9	9	8	9

6. Conclusion and Future Work

This paper considers the problem of partitioning a graph into (or covering with) a minimum number of s-clubs. For even values of s, we give an $\tilde{O}(n^{1/2})$ approximation algorithm, and then show that getting an $n^{1/2-\varepsilon}$ approximation is NP-hard. For odd values of s, we show that getting an $n^{1-\varepsilon}$ approximation is NP-hard. These results generalize Dondi et al. (2019), who considered the covering problem for $s = 2$ and $s = 3$, to all values of s for both the covering and partitioning variants. We also propose MIP-based heuristics, based on the approximation algorithms, that perform extremely well in practice, solving roughly half of the instances of Yezerska et al. (2019) and three-quarters of the instances of Gschwind et al. (2021) in less than one second. With an improved integer programming formulation and an associated branch-and-cut algorithm, we solve nearly all remaining instances and do so in significantly less time than previous approaches.

Future work may consider the partitioning and covering problems for other clique relaxations. Do they admit approximation algorithms with a nontrivial approximation factor? If so, can better practical performance be achieved by embedding a MIP inside them (instead of a greedy step)? How well do they perform in practice? Do the problems admit improved integer programming formulations, and what tricks can be used to speed them up? Do the resulting implementations outperform the general-purpose branch-and-price algorithms of Gschwind et al. (2021)?

Acknowledgments

We thank Hosseinali Salemi for initial discussions on this problem. This material is based upon work supported by the National Science Foundation under Grant Nos. 1942065 and 2318790.

References

Abbasi AA, Younis M (2007) A survey on clustering algorithms for wireless sensor networks. *Computer Communications* 30(14-15):2826–2841.

Abello J, Resende MG, Sudarsky S (2002) Massive quasi-clique detection. *LATIN 2002: Theoretical Informatics: 5th Latin American Symposium Cancun, Mexico, April 3–6, 2002 Proceedings 5*, 598–612 (Springer).

Almeida MT, Carvalho FD (2012) Integer models and upper bounds for the 3-club problem. *Networks* 60(3):155–166.

Balasundaram B, Butenko S, Hicks IV (2011) Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research* 59(1):133–142.

Balasundaram B, Butenko S, Trukhanov S (2005) Novel approaches for analyzing biological networks. *Journal of Combinatorial Optimization* 10:23–39.

Bourjolly JM, Laporte G, Pesant G (2002) An exact algorithm for the maximum k-club problem in an undirected graph. *European Journal of Operational Research* 138(1):21–28.

Bron C, Kerbosch J (1973) Algorithm 457: finding all cliques of an undirected graph. *Communications of the ACM* 16(9):575–577.

Deogun JS, Kratsch D, Steiner G (1997) An approximation algorithm for clustering graphs with dominating diametral path. *Information Processing Letters* 61(3):121–127.

Desormeaux WJ, Haynes TW, Henning MA, Yeo A (2014) Total domination in graphs with diameter 2. *Journal of Graph Theory* 75(1):91–103.

Dondi R, Mauri G, Sikora F, Italo Z (2019) Covering a graph with clubs. *Journal of Graph Algorithms and Applications* 23(2).

Eisen MB, Spellman PT, Brown PO, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences* 95(25):14863–14868.

Eppstein D, Löffler M, Strash D (2013) Listing all maximal cliques in large sparse real-world graphs. *ACM Journal of Experimental Algorithmics* 18(3):3.1.

Fernandess Y, Malkhi D (2002) k-clustering in wireless ad hoc networks. *Proceedings of the second ACM international workshop on Principles of mobile computing*, 31–37.

Gschwind T, Irnich S, Furini F, Calvo RW (2021) A branch-and-price framework for decomposing graphs into relaxed cliques. *INFORMS Journal on Computing* 33(3):1070–1090.

Hagberg A, Swart PJ, Schult DA (2008) Exploring network structure, dynamics, and function using NetworkX. Technical report, Los Alamos National Laboratory (LANL), Los Alamos, NM (United States).

Johnson DS (1974) Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences* 9(3):256–278.

Lu Y, Salemi H, Balasundaram B, Buchanan A (2022) On fault-tolerant low-diameter clusters in graphs. *INFORMS Journal on Computing* 34(6):3181–3199.

Luce RD (1950) Connectivity and generalized cliques in sociometric group structure. *Psychometrika* 15(2):169–190.

Lund C, Yannakakis M (1994) On the hardness of approximating minimization problems. *Journal of the ACM (JACM)* 41(5):960–981.

Miller RE, Muller DE (1960) A problem of maximum consistent subsets. Technical report, IBM Research Report RC-240, JT Watson Research Center, Yorktown Heights, NY.

Mokken RJ (1979) Cliques, clubs and clans. *Quality & Quantity* 13(2):161–173.

Moon JW, Moser L (1965) On cliques in graphs. *Israel Journal of Mathematics* 3:23–28.

Pattillo J, Veremyev A, Butenko S, Boginski V (2013a) On the maximum quasi-clique problem. *Discrete Applied Mathematics* 161(1-2):244–257.

Pattillo J, Youssef N, Butenko S (2013b) On clique relaxation models in network analysis. *European Journal of Operational Research* 226(1):9–18.

Rochat L, Bianchi-Demicheli F, Aboujaoude E, Khazaal Y (2019) The psychology of "swiping": A cluster analysis of the mobile dating app Tinder. *Journal of Behavioral Addictions* 8(4):804–813.

Salemi H, Buchanan A (2020) Parsimonious formulations for low-diameter clusters. *Mathematical Programming Computation* 12(3):493–528.

Seidman SB, Foster BL (1978) A graph-theoretic generalization of the clique concept. *Journal of Mathematical sociology* 6(1):139–154.

Validi H, Buchanan A (2020) The optimal design of low-latency virtual backbones. *INFORMS Journal on Computing* 32(4):952–967.

Validi H, Buchanan A, Lykhovyd E (2022) Imposing contiguity constraints in political districting models. *Operations Research* 70(2):867–892.

Veremyev A, Boginski V (2012) Identifying large robust network clusters via new compact formulations of maximum k-club problems. *European Journal of Operational Research* 218(2):316–326.

Veremyev A, Prokopyev OA, Pasiliao EL (2015) Critical nodes for distance-based connectivity and related problems in graphs. *Networks* 66(3):170–195.

Wotzlaw A (2014) On solving the maximum k-club problem. *arXiv preprint arXiv:1403.5111* .

Yezerska O, Pajouh FM, Veremyev A, Butenko S (2019) Exact algorithms for the minimum s-club partitioning problem. *Annals of Operations Research* 276(1-2):267–291.

Yu H, Paccanaro A, Trifonov V, Gerstein M (2006) Predicting interactions in protein networks by completing defective cliques. *Bioinformatics* 22(7):823–829.

Zachary WW (1977) An information flow model for conflict and fission in small groups. *Journal of Anthropological Research* 33(4):452–473.

Zuckerman D (2006) Linear degree extractors and the inapproximability of max clique and chromatic number. *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, 681–690.