On the Parameterized Complexity of the s-Club Cluster Edge Deletion Problem (Short Paper)

Fabrizio Montecchiani, Giacomo Ortali*, Tommaso Piselli and Alessandra Tappini

Università degli Studi di Perugia, Italy

Abstract

We study the parameterized complexity of the s-Club Cluster Edge Deletion problem: Given a graph G and two integers $s \geq 2$ and $k \geq 1$, is it possible to remove at most k edges from G such that each connected component of the resulting graph has diameter at most s? This problem is known to be NP-hard already when $s = 2$. We prove that it admits a fixed-parameter tractable algorithm when parameterized by s and the treewidth of the input graph.

Keywords

s-Club Cluster Edge Deletion, Parameterized Complexity, s-Club, Treewidth

1. Introduction

Graph clustering [1] is a classical task in data mining, with important applications in numerous fields including computational biology [2], image processing [3], and machine learning [4]. A prominent graph-theoretic formalization is Cluster Editing (also known as Correlation Clustering): Given a graph G and an integer k as input, the goal is to find a sequence of k operations, each of which can be an edge insertion or removal, such that the resulting graph is a so-called *cluster graph*, i.e., each of its connected components is a clique. If we restrict the editing operations to be edge removals only, then the problem is known as Cluster Edge Deletion. Equivalently, we seek for a partition of the vertices of G into cliques, such that the inter-cluster edges (whose end-vertices belong to different cliques) are at most k. Unfortunately, both Cluster Editing and Cluster Edge Deletion are well-known to be NP-complete [5, 6]. Indeed, their parameterized complexity with respect to the natural parameter k has been intensively investigated; in particular, both problems are in FPT [7, 8], but do not allow subexponential-time parameterized algorithms unless ETH fails [9, 6].

In many applications, modelling clusters with cliques might be a severe limitation, for instance, in presence of noise in the data collection process. Consequently, several notions of relaxed cliques have been introduced and investigated [10, 11]. We focus on the concept of s-club, in

Proceedings of the 23rd Italian Conference on Theoretical Computer Science, Rome, Italy, September 7-9, 2022

*Corresponding author.

fabrizio.montecchiani@unipg.it (F. Montecchiani); giacomo.ortali@unipg.it (G. Ortali); tommaso.piselli@studenti.unipg.it (T. Piselli); alessandra.tappini@unipg.it (A. Tappini)

0000-0002-0543-8912 (F. Montecchiani); 0000-0002-4481-698X (G. Ortali); 0000-0001-9192-2067 (A. Tappini)

which each pair of vertices is at distance at most $s \geq 2$ in the cluster. (Note that a 1-club is in fact a clique.) We remark that defining clusters as s-clubs proved to be effective in several application scenarios such as social network analysis and bioinformatics [12, 13, 14, 15, 16]. The s-Club Cluster Edge Deletion problem can be stated analogously as Cluster Edge Deletion by replacing cliques with s-clubs (formal definitions are given later). Unfortunately, s-Club Cluster Edge Deletion is NP-complete already for $s = 2$ [17]. Also, 2-Club Cluster Edge Deletion belongs to FPT parameterized by k [18, 17], and it admits no subexponential-time parameterized algorithm in k [19]. More in general, for any $s \geq 2$, s-Club Cluster Edge Deletion cannot be solved in time $2^{o(k)}n^{O(1)}$ unless ETH fails [19].

Based on the above discussion, we know that it is unlikely that s-Club Cluster Edge Deletion lies is FPT when parameterized by s, whereas the complexity of the problem parameterized by $s + k$ is open to the best of our knowledge. In this paper, we instead focus on those scenarios in which the solution size (measured by k) is large, and we still aim for tractable problems based on alternative parameterizations. In this respect, treewidth is a central parameter in the parameterized complexity analysis (see [20, 21]). We prove that s-Club Cluster Edge Deletion lies in FPT when parameterized by $s + \mathrm{tw}$, where tw is an upper bound for the treewidth of the input graph.

Theorem 1 *Let G be an n-vertex graph of treewidth at most* tw. *There is an algorithm that solves the s-Club Cluster Edge Deletion problem on G in $O(2^{2^{O(\mathrm{tw}^2 \log s)}} \cdot n)$ time.*

From the technical viewpoint, the main crux of our approach lies in the definition of sufficiently small records that allow to keep track of the distances between pairs of vertices in a (partial) s-club. With such records at hand, we then apply a standard DP algorithm over a tree decomposition of the input graph, which still requires a nontrivial amount of technicalities in order to update the records. Our records have similarities, but also several key differences, with those used in a technique presented by Dondi and Lafond in [22, Thm. 14], which solves a related problem for the restricted case $s = 2$.

For space constraints many technicalities are omitted and we only sketch the proof of Theorem 1. See [23] for the full version of the paper.

Preliminaries and notation. For any $d \in \mathbb{Z}^+$, we use $[d]$ as shorthand for the set $\{1, 2, \ldots, d\}$. Let $G = (V, E)$ be a graph. For any $W \subset V$, we denote by $G[W]$ the subgraph of G induced by the vertices of W. The *neighborhood* of a vertex v of G is defined as $N_G(v) = \{u : uv \in E\}$. Given two vertices $u, v \in V$, the *distance in* G between u and v, denoted by $d_G(u, v)$, is the number of edges in any shortest path between u and v in G. The *diameter* of G is the maximum distance in G between any two of its vertices. An *s-club* of G, with $s \geq 1$, is a subset $W \subseteq V$ such that the diameter of $G[W]$ is at most s. A *partition* of G is a collection of subsets $\mathcal{C} = \{C_i\}_{i \in [d]}$ such that: (a) $\bigcup_{i=1}^{d} C_i = V$, and (b) $C_i \cap C_j = \emptyset$ for each $i, j \in [d]$ with $i \neq j$. We denote by $E_{\mathcal{C}}$ the set of all edges uv of G such that $u, v \in C_i$, for some $i \in [d]$. We study the following problem.

s-Club Cluster Edge Deletion
Input: $G = (V, E)$, $k \geq 1$, $s \geq 2$.
Output: A partition $\mathcal{C} = \{C_i\}_{i \in [d]}$ of G such that C_i is an s-club for each $i \in [d]$, and $|E \setminus E_{\mathcal{C}}| \leq k$.

In what follows, for a graph $G = (V, E)$, the pair $(\mathcal{X}, T)$ denotes a *nice tree-decomposition*, such that $\mathcal{X} = \{X_i\}_{i \in [\ell]}$ is a collection of subsets of vertices of V, called *bags*, and T is a tree whose nodes are in one-to-one correspondence with the elements of $\mathcal{X}$. We point the reader to [24, 25] for the required background.

2. Sketch of the Proof of Theorem 1

The proof is based on a DP algorithm over a nice tree-decomposition. We first describe the records to be stored at each bag, and we then sketch the algorithm.

Definition of the records. Let $G = (V, E)$ be an n-vertex graph and let $(\mathcal{X}, T)$ be a nice tree-decomposition of G of width tw. For each $i \in [\ell]$, let T_i be the subtree of T rooted at the bag $X_i \in \mathcal{X}$ and let $G_i = (V_i, E_i)$ be the subgraph of G induced by the vertices that belong to at least one bag of T_i. A subset of vertices $C \subseteq V_i$ is a *potential s-club*, and we let $\partial C = C \cap X_i$ and int$(C) = C \setminus X_i$.

The first item of the record is a table that stores the pairwise distances of the vertices in ∂C. Namely, let $D(\partial C)$ be a table having one row and one column for each vertex in ∂C, and such that:

$$D(\partial C)[a, b] = \begin{cases} 0, & \text{if } a = b \\ d_{G[C]}(a, b), & \text{if } 1 \leq d_{G[C]}(a, b) \leq s \\ \infty, & \text{otherwise.} \end{cases}$$

The second item is a table that stores the distance between pairs of vertices such that one is in ∂C and the other is in int(C). Two vertices u, u' in int(C) are *equivalent* with respect to ∂C, if for each vertex $a \in \partial C$, then either $1 \leq d_{G[C]}(u, a) = d_{G[C]}(u', a) \leq s$, or $d_{G[C]}(u, a) > s$ and $d_{G[C]}(u', a) > s$. Namely, let $H(\partial C)$ be a table having one column for each vertex $a \in \partial C$, and one row for each equivalence class with respect to ∂C, denoted by $[u]_{\partial C}$. We have:

$$H(\partial C)[u, a] = \begin{cases} d_{G[C]}(u, a), & \text{if } 1 \leq d_{G[C]}(u, a) \leq s \\ \infty, & \text{otherwise.} \end{cases}$$

The third (and last) item of the record represents the key difference to extend the result in [22] to $s \geq 2$. Suppose that C is a subset of an s-club C' of G and that there exist two vertices $u, u' \in$ int(C) whose distance in $G[C]$ is larger than s. Then, any path between u and u' not containing two vertices in ∂C has length larger than s. Hence, since ∂C is a separator of G, we have to consider paths in G between u and u' going through some pair of vertices in ∂C. We formalize this observation. Let $w, z \in$ int(C) be two vertices such that $d_{G[C]}(w, z) > s$. A *request for ∂C*, denoted by R_{wz}, is a table having one row and one column for each vertex in ∂C. Namely, for each $a, b \in \partial C$, if there exists $2 \leq \delta \leq s - 2$ such that connecting a and b with a path π of length δ makes the distance between w and z to be at most s, then $R_{wz}[a,b] = \delta$, while $R_{wz}[a,b] = \star$ otherwise. Observe that if there exist two requests R_{wz} and $R_{w'z'}$ such that $R_{wz}[a,b] = R_{w'z'}[a,b]$ for each pair $a, b \in \partial C$, then w and w' are equivalent with respect to ∂C (i.e., $w, w' \in [w]_{\partial C}$), and the same holds for z and z'. Thus, we avoid storing duplicated requests and we denote by $Q(\partial C)$ the set containing all distinct requests for ∂C.

If a potential s-club C is such that $\partial C = \emptyset$ (recall that $C \subseteq V_i$), then we call it *complete*. Consider a partitioning $\mathcal{P}_i^l$ of G_i into potential s-clubs and let $\mathcal{C}_i^l = \{C_{j,i}^l \mid j \in [d_l]\}$ be the potential s-clubs in $\mathcal{P}_i^l$ that are not complete. Let $\partial\mathcal{C}_i^l = \{\partial C_{j,i}^l \mid j \in [d_l]\}$, $\mathcal{D}_i^l = \{D(\partial C_{j,i}^l) \mid j \in [d_l]\}$, $\mathcal{H}_i^l = \{H(\partial C_{j,i}^l) \mid j \in [d_l]\}$, and $\mathcal{Q}_i^l = \{Q(\partial C_{j,i}^l) \mid j \in [d_l]\}$. A *solution* of X_i is a tuple $S_i^l = \langle \partial\mathcal{C}_i^l, \mathcal{D}_i^l, \mathcal{H}_i^l, \mathcal{Q}_i^l, k_i^l \rangle$. Here k_i^l is an integer, called *edge-counter*, equal to $|E_i \setminus \mathcal{P}_i^l(E_i)|$, hence $k_i^l \leq k$. Two solutions $S_i^l = \langle \partial\mathcal{C}_i^l, \mathcal{D}_i^l, \mathcal{H}_i^l, \mathcal{Q}_i^l, k_i^l \rangle$ and $S_i^g = \langle \partial\mathcal{C}_i^g, \mathcal{D}_i^g, \mathcal{H}_i^g, \mathcal{Q}_i^g, k_i^g \rangle$ are *distinct* if $\partial\mathcal{C}_i^l \neq \partial\mathcal{C}_i^g$, or $\mathcal{D}_i^l \neq \mathcal{D}_i^g$, or $\mathcal{H}_i^l \neq \mathcal{H}_i^g$, or $\mathcal{Q}_i^l \neq \mathcal{Q}_i^g$. Observe that if S_i^l and S_i^g are not distinct but $k_i^l < k_i^g$, then it suffices to consider only S_i^l.

Lemma 1 *For a bag X_i, there exist $O(2^{2^{O(\mathrm{tw}^2 \log s)}})$ distinct solutions.*

Sketch of the algorithm. Let X_i be the current bag visited by the algorithm. We compute the set of solutions for X_i based on the solutions computed for its child or children. If the resulting set of solutions is empty, the algorithm halts and returns a negative answer. The running time of the algorithm follows from Lemma 1. We only describe the case in which X_i is an introduce bag. The cases in which X_i is a leaf, a forget, or a join bag are omitted in this extended abstract.

X_i **is an introduce bag.** Let $X_j = X_i \setminus \{v\}$ be the child of X_i. The algorithm exhaustively extends each solution S_j^l of X_j as follows. It first generates at most d_l new partitions by placing v in each $\partial C' \in \partial\mathcal{C}_j^l$. Also, it generates a partition in which v forms a new potential s-club $C = \partial C = \{v\}$. Consider one of the new partitions generated by the algorithm. In order to build the corresponding new solution for X_i, we distinguish the following two cases.

Case A ($\partial C = \{v\}$). $D(\partial C)$ is trivially defined, $H(\partial C)$ and $Q(\partial C)$ are empty.

Case B ($\partial C = \partial C' \cup \{v\}$). The next observation immediately follows from the fact that $\partial C = \partial C' \cup \{v\}$ and int(C) = int(C').

Observation 1 *Suppose that there exist $a, b \in \partial C'$ such that $d_{G[C']}(a,b) > d_{G[C]}(a,b)$, then any shortest path between a and b in $G[C]$ contains vertex v.*

– Computing $D(\partial C)$ from $D(\partial C')$.

1. We add a new row and a new column for vertex v.
2. For each vertex $a \in \partial C'$, let $\delta_{av} = \min_{b \in N_{G[X_i]}(v)} D(\partial C')[a,b]$, and note that $\delta_{av} = 0$ if edge av belongs to $G[C]$. Clearly, it holds that

$$D(\partial C)[a,v] = \begin{cases} \infty, & \text{if } \delta_{av} \in \{s, \infty\} \\ 1 + \delta_{av}, & \text{otherwise.} \end{cases}$$

3. By Observation 1, for each pair $a, b \in \partial C'$, the corresponding value of $D(\partial C)$ can be updated as follows:

$$D(\partial C)[a,b] = \min\{D(\partial C')[a,b], D(\partial C)[a,v] + D(\partial C)[b,v]\}.$$

– Computing $H(\partial C)$ from $H(\partial C')$.

1. We add a new column for vertex v.

2. For each equivalence class $[u]_{\partial C'}$, let $\delta_{uv} = \min_{a \in N_{G[X_i]}(v)} H(\partial C')[u,a]$. Since there is no edge uv such that $u \in$ int(C), it follows that

$$H(\partial C)[u,v] = \begin{cases} \infty, & \text{if } \delta_{uv} \in \{s, \infty\} \\ 1 + \delta_{uv}, & \text{otherwise.} \end{cases}$$

3. By Observation 1, for each pair of vertices $u \in$ int(C') and $a \in \partial C'$, the corresponding value of $H(\partial C)$ can be updated as follows:

$$H(\partial C)[u,a] = \min\{H(\partial C')[u,a], H(\partial C)[u,v] + D(\partial C)[v,a]\}.$$

– Computing $Q(\partial C)$ from $Q(\partial C')$. Note that the addition of v cannot lead to new requests but it may actually yield the update of some request in $Q(\partial C')$.

1. For each request R_{wz} in $Q(\partial C')$, we verify whether, as a consequence of the introduction of v, there exists a cell $R_{wz}[a,b]$ such that $D(\partial C)[a,b] \leq R_{wz}[a,b]$. If such a cell exists, we say that R_{wz} is *fulfilled*. We add R_{wz} to $Q(\partial C)$ if and only if R_{wz} is not fulfilled.

2. If R_{wz} is not fulfilled, before adding it to $Q(\partial C)$, we update it as follows:
   a) We add a row and a column for v.
   b) For each pair $a, b \in \partial C'$, we compute

   $$\delta_{ab} = \min\{(H(\partial C)[w,a] + H(\partial C)[z,b], H(\partial C)[z,a] + H(\partial C)[w,b]\}.$$

   Observe that $\delta_{ab} + D(\partial C)[a,b] > s$, otherwise the request would have been fulfilled before.
   c) By definition of request, we have $R_{wz}[a,b] = s - \delta_{ab}$, if $\delta_{ab} < s - 1$, and $R_{wz}[a,b] = \star$, otherwise.

Finally, in both **Case A** and **Case B**, we observe that, in order to obtain the edge-counter of the new solution, k_j^l needs to be increased by the number of edges incident to v whose other end-vertex is in X_i but not in C. If the resulting edge-counter is greater than k, the solution is discarded.

3. Discussion and Open Problems

We have shown that the s-Club Cluster Edge Deletion problem parameterized by $s + \mathrm{tw}$ (where tw bounds the treewidth of the input graph) belongs to FPT. On the other hand, we know that the problem parameterized by s alone is paraNP-hard. It remains open the complexity of s-Club Cluster Edge Deletion parameterized by tw alone. We conclude by remarking that our approach can be slightly modified to solve a related problem, namely s-Club Cluster Vertex Deletion, in which we seek for k vertices whose removal yields a set of disjoint s-clubs.

References

[1] S. E. Schaeffer, Graph clustering, Comput. Sci. Rev. 1 (2007) 27–64. doi:10.1016/j.cosrev.2007.05.001.

[2] A. Ben-Dor, R. Shamir, Z. Yakhini, Clustering gene expression patterns, J. Comput. Biol. 6 (1999) 281–297. doi:10.1089/106652799318274.

[3] Z. Wu, R. M. Leahy, An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 15 (1993) 1101–1113. doi:10.1109/34.244673.

[4] N. Bansal, A. Blum, S. Chawla, Correlation clustering, Mach. Learn. 56 (2004) 89–113. doi:10.1023/B:MACH.0000033116.57574.95.

[5] R. Shamir, R. Sharan, D. Tsur, Cluster graph modification problems, Discret. Appl. Math. 144 (2004) 173–182. doi:10.1016/j.dam.2004.01.007.

[6] C. Komusiewicz, J. Uhlmann, Cluster editing with locally bounded modifications, Discret. Appl. Math. 160 (2012) 2259–2270. doi:10.1016/j.dam.2012.05.019.

[7] S. Böcker, A golden ratio parameterized algorithm for cluster editing, J. Discrete Algorithms 16 (2012) 79–89. doi:10.1016/j.jda.2012.04.005.

[8] J. Chen, J. Meng, A 2k kernel for the cluster editing problem, J. Comput. Syst. Sci. 78 (2012) 211–220. doi:10.1016/j.jcss.2011.04.001.

[9] F. V. Fomin, S. Kratsch, M. Pilipczuk, M. Pilipczuk, Y. Villanger, Tight bounds for parameterized complexity of cluster editing with a small number of clusters, J. Comput. Syst. Sci. 80 (2014) 1430–1447. doi:10.1016/j.jcss.2014.04.015.

[10] B. Balasundaram, F. M. Pajouh, Graph Theoretic Clique Relaxations and Applications, Springer New York, 2013, pp. 1559–1598. doi:10.1007/978-1-4419-7997-1_9.

[11] C. Komusiewicz, Multivariate algorithmics for finding cohesive subnetworks, Algorithms 9 (2016) 21. doi:10.3390/a9010021.

[12] R. Alba, A graph-theoretic definition of a sociometric clique, Journal of Mathematical Sociology 3 (1973) 113–126. doi:10.1080/0022250X.1973.9989826.

[13] B. Balasundaram, S. Butenko, S. Trukhanov, Novel approaches for analyzing biological networks, J. Comb. Optim. 10 (2005) 23–39. doi:10.1007/s10878-005-1857-x.

[14] s. Laan, M. Marx, R. Mokken, Close communities in social networks: Boroughs and 2-clubs, SSRN Electronic Journal (2015). doi:10.2139/ssrn.2686127.

[15] R. Mokken, Cliques, clubs and clans, Quality & Quantity: International Journal of Methodology 13 (1979) 161–173. doi:10.1007/BF00139635.

[16] R. Mokken, S. Laan, Close communication and 2-clubs in corporate networks: Europe 2010, Social Network Analysis and Mining 6 (2016). doi:10.1007/s13278-016-0345-x.

[17] H. Liu, P. Zhang, D. Zhu, On editing graphs into 2-club clusters, in: FAW-AAIM, volume 7285 of *LNCS*, Springer, 2012, pp. 235–246. doi:10.1007/978-3-642-29700-7_22.

[18] F. N. Abu-Khzam, N. Makarem, M. Shehab, An improved fixed-parameter algorithm for 2-club cluster edge deletion, CoRR abs/2107.01133 (2021). URL: https://arxiv.org/abs/2107.01133.

[19] N. Misra, F. Panolan, S. Saurabh, Subexponential algorithm for d-cluster edge deletion: Exception or rule?, J. Comput. Syst. Sci. 113 (2020) 150–162. doi:10.1016/j.jcss.2020.05.008.

[20] R. G. Downey, M. R. Fellows, Parameterized Complexity, Monographs in Computer Science, Springer, 1999.

[21] N. Robertson, P. D. Seymour, Graph minors. II. Algorithmic aspects of tree-width, J. Algorithms 7 (1986) 309–322.

[22] R. Dondi, M. Lafond, On the tractability of covering a graph with 2-clubs, in: L. A. Gasieniec, J. Jansson, C. Levcopoulos (Eds.), FCT 2019, volume 11651 of *LNCS*, Springer, 2019, pp. 243–257. doi:10.1007/978-3-030-25027-0_17.

[23] F. Montecchiani, G. Ortali, T. Piselli, A. Tappini, On the parameterized complexity of the s-club cluster edge deletion problem, CoRR abs/2205.10834 (2022). doi:10.48550/arXiv.2205.10834.

[24] H. L. Bodlaender, T. Kloks, Efficient and constructive algorithms for the pathwidth and treewidth of graphs, J. Algorithms 21 (1996) 358–402. doi:10.1006/jagm.1996.0049.

[25] T. Kloks, Treewidth, Computations and Approximations, volume 842 of *LNCS*, Springer, 1994.