SBM

Matemática Contemporânea, Vol. 44, 1–12

Editing to Cliques: A Survey of FPT Results and Recent Applications in Analyzing Large Datasets

Frances A. Rosamond

Abstract

The CLUSTER EDITING problem aims to transform an undirected graph into a vertex-disjoint union of cliques by adding or deleting at most k edges. It has been proven NP-hard several times as it has been discovered and rediscovered in important application areas. Most biologists, social scientists and other practitioners have databases and networks that need clustering. This paper describes four approaches to more realistically model what is found in practice: 1) Parameterized Complexity, kernelization, and the Cluster Editing Crown Reduction Rule, 2) Bigger, more aggressive, aggregate parameterization, 3) FUZZY CLUSTER EDITING and moving approximation into the modeling, and 4) Working "bottom-up" uniting kernelization and heuristics. Some applications are discussed.

1 Introduction

Clustering of information is important in various domains or in data-mining for related sets of objects. A graph is obtained by setting a thresh-

2000 AMS Subject Classification: 68R10 and 05C69.

Key Words and Phrases: Computational Complexity, Parameterized Complexity, Multivariate Algorithmics, Cluster Editing

old on some measure of pairwise relatedness or similarity. Vertices represent objects. Similar objects are connected by an edge. The aim of clustering is to partition the vertices into disjoint subsets, so that genes (or other objects) that correspond to the vertices within each subset display some measure of homogeneity. Likely, the input graph is corrupted and we have to clean (edit) the graph to reconstruct the clustering by adding edges in the case of omissions or deleting edges when there are false positives. There are many variations on cluster editing because there are many applications. An interpretation may be the distance between objects as a measure of their similarity: the larger the distance, the more similar the objects. An objective, parameterized by the desired number k of clusters, may be to find a k-clustering that minimizes the sum of distances between pairs of objects in different clusters (minimum k-cut). Another interpretation may be that the larger the distance, the more dissimilar the objects. Objectives parameterized by the desired number k of clusters may be: minimize the maximum diameter of a cluster (k-clustering), minimize the average distance to a centroid object (k-median), minimize average squared distance to an arbitrary centroid object (k-means), maximize the sum of distances between pairs of objects in different clusters (maximum k-cut).

The Cluster Editing problem was first introduced in the context of machine learning under the name, Correlation Clustering [BBC04]. The underlying model is that objects can be truly categorized, and probabilities are given about pairs of objects belonging to common categories. Bansal *et al.* [BBC04] addressed minimizing disagreements and maximizing agreements, proved NP-hardness and gave a constant-factor approximation algorithm for the special case in which the graph is complete (full information) and every edge has the same weight.

An $O(\log n)$-approximation [DFHT05] has been shown for the general case based on a linear-programming rounding technique, and an $O(r^3)$-approximation for $K_{r,r}$-minor-free graphs. The problem is equivalent to minimum multicut, and therefore APX-hard and difficult to approximate better than $\Theta(\log n)$ [DFHT05]. The problem remains NP-hard on graphs with maximum degree six [KU12]. There is no PTAS unless P = NP, but there is a polynomial-time 4-approximation [CGW05].

Cluster Editing is fixed-parameter tractable under various parameterizations. Parameterized complexity is briefly reviewed in the next section.

2 Parameterized Complexity

Parameterized complexity studies a *two-dimensional* generalization of polynomial-time where in addition to the overall input size n, every problem comes with a secondary measurement, the *parameter* [DF13]. The parameter allows us to examine various aspects of the data. For example, the NP-complete Closest Substring problem, an important consensus problem in computational biology, has an input of k strings over an alphabet Σ and non-negative integers d and L. The question asks if there is a string s of length L, and each of the given strings has a length L substring that differs from s by not more than distance d. The complexity of the problem has been examined when parameterized by d alone, by k alone, by d and k together, and for L, d and k combined. Adding parameters can lead to increased realism, better understanding of the problem properties, new formulations of the question, and ultimately to more efficient and practical algorithms.

A problem is *fixed-parameter tractable* (FPT) if it can be decided in time $O(f(k)n^c)$ or (additively) $O(f(k) + n^c)$, where n is the input size and k is a feature of the problem called the *parameter*, f is an arbitrary function and c is a constant independent of both n and k. For examples, consider clique and vertex cover. A *clique* in a graph is a subset of the vertices such that every two distinct vertices in the subset are adjacent. A clique is also called a *cluster* or *complete* graph. A *vertex cover* is a subset of the vertices such that every edge in the graph has an endpoint in the subset (edges are "covered" by vertices). Those vertices not in the cover form an independent set. Clique and vertex cover are *duals*.

Finding a Vertex Cover or finding a Clique in a graph are both NP-complete problems. They can be solved by brute force: "generate and check all k-subsets," naively $O(n^{(k+1)})$. In the parameterized setting they differ. Clique, parameterized by k, the size of the clique, is not FPT and remains brute force, but Vertex Cover is FPT with running time $O(1.2738^k + kn)$, where the parameter k is the size of the vertex cover [CM12].

A parameterized problem is said to admit a *kernel* if, in **polynomial time**, the size of the instance I can be reduced to a function in k, the parameter, while preserving the answer. A problem is FPT if and only if it is *kernelizable.* Kernelization can be thought of as *preprocessing with guarantee* via a suite of *reduction rules* in which an input to the problem is replaced by a smaller input, called a *kernel.* Vertex Cover can be reduced to an instance of size $2k$ by using only three *reduction rules.*

> (1)Degree 0 Rule: If G has a vertex of degree 0, delete that vertex since it cannot cover any edges, and reduce the size of G. The size of the parameter k stays the same.
>
> (2)Degree 1 Rule: If G has a vertex u of degree 1, delete u and its edge. Put its neighbor v in the vertex cover (since v covers the edge (u, v) and possibly more edges). Reduce G to $G' = G - u - v$ and k reduces to $k - 1$.
>
> (3)Large Degree Rule: If G has a vertex u and the degree $u \geq k$, then put u into the vertex cover (else we must take all k of its neighbors). Reduce G to $G' = G - u - v$ and k reduces to $k - 1$.

Each of these rules must be proved sound. That is, G' is a Vertex Cover if and only if G is. Reduction rules often cascade, reducing the input even further (e.g., invoking the Large Degree Rule may introduce vertices of degree 1 so that the Degree 1 Rule can be invoked again). Running time can be improved by interleaving kernelization with depth-bounded search trees: kernelize, begin a bounded search tree, rekernelize the children, repeat. The reduced instance still must be solved, by brute force or other methods.

The parameterized CLUSTER EDITING problem is defined as follows:

CLUSTER EDITING
Input: (G, k)
Parameter: k
Question: Can G be transformed into a disjoint union of complete graphs
(a cluster graph) by adding or deleting $\leq k$ edges?

Cai showed that it is FPT to decide whether an input graph can be transformed into a graph with a specified hereditary property by deleting vertices and/or edges, and adding edges, when the hereditary property can be characterized by a finite set of forbidden induced subgraphs [Cai96].

CLUSTER EDITING is equivalent to destroying (by adding/deleting edges) all occurrences of an induced P_3, and therefore by [Cai96] is FPT by a search tree algorithm that runs in time $O(3^k n^4)$.

There has been steady improvement in run time and kernel size for CLUSTER EDITING. For references, see the survey [BB13] and the *Table of Races* on the Parameterized Complexity wiki (http://www.fpt.wikidot.com) which reports the current best known $f(k)$ and kernel size for many FPT problems. The current best running time is $O(1.62^k + m + n)$, found by searching for a conflict triple, then branching on integer-weighted instances [Böc12]. Current best kernel size is $2k$ vertices, using kernelization based on edge cuts [CM12].

3 Crown Reduction Rule

The Degree 1 Reduction Rule for VERTEX COVER was generalized to the hugely useful Crown Reduction Rule for VERTEX COVER [Fel06], and has led to crown rules for other problems including CLUSTER EDITING

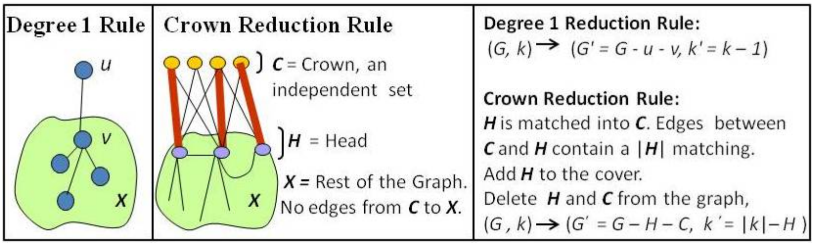
Degree 1 Rule
u
v
X
Crown Reduction Rule
} C = Crown, an independent set
}H = Head
X = Rest of the Graph.
No edges from C to X.
X
Degree 1 Reduction Rule:
(G, k) → (G' = G - u - v, k' = k – 1)
Crown Reduction Rule:
H is matched into C. Edges between C and H contain a |H| matching.
Add H to the cover.
Delete H and C from the graph,
(G , k) → (G' = G – H – C, k ' = |k|–H)

Figure 1: Degree 1 Reduction Rule and its extension to the Crown Reduction Rule for kernelizing Vertex Cover

[FLRS07], see Figure 1. These rules are important in applications such as the Clustal XP portal that allows biochemists an extended and parallel version of Clustal [DFRS04], and in clustering projects such as Neurosciences and Alcoholism, Optimization and Metaheuristic Methods for Cancer Research, Mouse Phenotype Analysis, Combinatorial Analysis of North Sea Historical Data, and others [AKFLS07]. Cluster editing reduction rules have been used to reduce leukemia gene expression datasets [BB13].

Using the 4-approximation [CGW05], Fellows *et al.* [Fel06] found a kernel size of $24k$ vertices for Cluster Editing and predicted a $6k$ kernel, later found by using a Cluster Editing Crown Rule [FLRS07].

4 Use big, aggregate parameterization

Contrary to expectation, adding parameters does not make a problem more difficult, and it is still elegant [AK13]. Abu-Khzam [AK13] noted that in some applications, it is important to distinguish between the expected number of false positives and that of false negatives. Frequency of errors could be low in general (noise), meaning that errors per vertex are few. In practice, total number of errors per data element may be small. Abu-Khzam designed a big aggregate parameter (k, a, d, s) where k denotes number of edge additions or deletions, each vertex is incident with at most a edge additions, and at most d edge deletions, s is the minimum size of the cliques in the resulting cluster graph. Additional parameters, say for outliers could be added. Abu-Khzam provides 17 data reduction rules and shows the problem is FPT if $s > 2(a+d)$.

Other variants of Cluster Edit utilize aggregate parameterization. The p-Cluster Edit problem asks whether G can be transformed into a cluster graph with at most p cliques by adding/deleting at most k edges. The aggregate parameter is (k, p). The (p, t)-Constrained Cluster Editing problem transforms G into a cluster graph with at most p clusters, applying at most k edits, and every vertex v has edit degree at most $r(v) < t$, e.g., individual edge-edit constraints for every vertex. References can be found in [BB13, FKP$^+$13]. Including more aspects of a problem via big, aggregate parameterization (a "Santa Claus sack" filled with various sized complexity parameter "gifts") can more realistically and usefully describe what is found in practice.

5 Move approximation into the modeling

Features of complicated clustering problems can be moved into the problem definition as parameters, allowing more realistic modeling. Complete information about the input often is not available. Some input vertex pairs may have an undetermined, unknown or undecided relation. Bodlaender *et al.* [BFH$^+$10] investigated clustering with partial information, a more realistic model. They defined a *fuzzy graph* where E is the set of real edges, F is the set of unknown relationship (fuzzy edges), and between all other pairs of vertices in the graph are (definite) non-edges. Fuzzy Cluster Editing has kernel size $O(k^2 + r)$ when the input is a fuzzy graph and the parameter is (k, r), where k is a cost parameter of editing, and r is a structural parameter, the minimum number of vertices required to cover the undecided edges of the fuzzy graph. The structural parameter is motivated by applications where only a small number of "trouble-maker" vertices are the "cause" of the uncertain information about the input. It is not known if the problem is FPT when the parameter is k alone.

6 "Bottom-up" kernelization and heuristics

The approach to algorithm design of abstracting the messy, complicated real-world problem into a simplified, clean special case which is often proved NP-complete, could be labeled "top down". As proposed by Michael Fellows in discussions, a viable alternative is to work "bottom up," starting with heuristics that are successful in practice, and incorporating parameterization. Bastos *et al.* [BOP+14] coupled reduction rules for CLUSTER EDITING with two greedy constructive heuristics. Their two–phase algorithm constructs and evaluates step-by-step, a feasible solution. A local search phase then attempts to improve the initial solution. Experimental results show that the algorithms are able to find high-quality solutions in practical runtime.

7 Conclusion

Parameterized complexity algorithm design has a mission to realistically model what is found in practice and communicate our work to practitioners. We offer kernelization and crown reduction rules, aggressive, aggregate parameterization, moving approximation into the modeling and working bottom-up by finding heuristic subroutines to parameterize as programmatic directions in this regard.

References

[AK13] Faisal N. Abu-Khzam, *The multi-parameterized cluster editing problem*, Combinatorial optimization and applications, Lecture Notes in Comput. Sci., vol. 8287, Springer, Cham, 2013, pp. 284–294. MR 3204069

[AKFLS07] Faisal N. Abu-Khzam, Michael R. Fellows, Michael A. Langston, and W. Henry Suters, *Crown structures for vertex* *cover kernelization*, Theory Comput. Syst. **41** (2007), no. 3, 411–430. MR 2352539

[BB13] Sebastian Böcker and Jan Baumbach, *Cluster editing*, The nature of computation, Lecture Notes in Comput. Sci., vol. 7921, Springer, Heidelberg, 2013, pp. 33–44. MR 3102002

[BBC04] Nikhil Bansal, Avrim Blum, and Shuchi Chawla, *Correlation clustering*, Machine Learning **56** (2004), no. 1-3, 89–113.

[BFH$^+$10] Hans L. Bodlaender, Michael R. Fellows, Pinar Heggernes, Federico Mancini, Charis Papadopoulos, and Frances Rosamond, *Clustering with partial information*, Theoret. Comput. Sci. **411** (2010), no. 7-9, 1202–1211. MR 2606055

[Böc12] Sebastian Böcker, *A golden ratio parameterized algorithm for cluster editing*, J. Discrete Algorithms **16** (2012), 79–89. MR 2960346

[BOP$^+$14] Lucas Bastos, Luiz Satoru Ochi, Fábio Protti, Anand Subramanian, Ivan César Martins, and Rian Gabriel S. Pinheiro, *Efficient algorithms for cluster editing*, Journal of Combinatorial Optimization (2014), 1–25.

[Cai96] Leizhen Cai, *Fixed-parameter tractability of graph modification problems for hereditary properties*, Inform. Process. Lett. **58** (1996), no. 4, 171–176. MR 1413637

[CGW05] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth, *Clustering with qualitative information*, J. Comput. System Sci. **71** (2005), no. 3, 360–383. MR 2168358 (2006f:68141)

[CM12] Jianer Chen and Jie Meng, *A $2k$ kernel for the cluster editing problem*, J. Comput. System Sci. **78** (2012), no. 1, 211–220. MR 2896358

[DF13] Rodney G. Downey and Michael R. Fellows, *Fundamentals of parameterized complexity*, Texts in Computer Science, Springer, London, 2013. MR 3154461

[DFHT05] Erik D. Demaine, Fedor V. Fomin, Mohammadtaghi Hajiaghayi, and Dimitrios M. Thilikos, *Subexponential parameterized algorithms on bounded-genus graphs and H-minor-free graphs*, J. ACM **52** (2005), no. 6, 866–893. MR 2179550 (2006g:68210)

[DFRS04] Frank Dehne, Mike Fellows, Frances Rosamond, and Peter Shaw, *Greedy localization, iterative compression, and modeled crown reductions: New fpt techniques, an improved algorithm for set splitting, and a novel 2k kernelization for* VERTEX COVER, Parameterized and Exact Computation, First International Workshop, IWPEC 2004, Bergen, Norway, September 14–17, 2004. (Rod Downey, Michael Fellows, and Frank Dehne, eds.), Lecture Notes in Computer Science, vol. 3162, Springer Berlin Heidelberg, 2004, pp. 271–280.

[Fel06] Michael R. Fellows, *The lost continent of polynomial time: Preprocessing and kernelization*, Parameterized and Exact Computation. Second International Workshop, IWPEC 2006, Zürich, Switzerland, September 13–15, 2006. (HansL. Bodlaender and MichaelA. Langston, eds.), Lecture Notes in Computer Science, vol. 4169, Springer Berlin Heidelberg, 2006, pp. 276–277.

[FKP$^+$13] Fedor V. Fomin, Stefan Kratsch, Marcin Pilipczuk, Michał Pilipczuk, and Yngve Villanger, *Tight bounds for parameterized complexity of cluster editing*, 30th International Symposium on Theoretical Aspects of Computer Science, LIPIcs. Leibniz Int. Proc. Inform., vol. 20, Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2013, pp. 32–43. MR 3089968

[FLRS07] Michael Fellows, Michael Langston, Frances Rosamond, and Peter Shaw, *Efficient parameterized preprocessing for cluster editing*, Fundamentals of Computation Theory. 16th International Symposium, FCT 2007, Budapest, Hungary, August 27–30, 2007. (Erzsébet Csuhaj-Varjú and Zoltán Ésik, eds.), Lecture Notes in Computer Science, vol. 4639, Springer Berlin Heidelberg, 2007, pp. 312–321.

[KU12] Christian Komusiewicz and Johannes Uhlmann, *Cluster editing with locally bounded modifications*, Discrete Appl. Math. **160** (2012), no. 15, 2259–2270. MR 2954767

Frances A. Rosamond
Department of Informatics
University of Bergen
Norway
http://www.cdu.edu.au/engit/staff-profiles/frances-rosamond