Deterministic Cluster Deletion Approximation Algorithm

Vicente Balmaseda

Abstract

Cluster deletion is an edge modification problem that converts a graph into a disjoint union of cliques by deleting the minimum number of edges. We simplify a combinatorial 4-approximation deterministic algorithm based on strong triadic closure labeling and the deterministic Pivot algorithm for correlation clustering. We also rewrite the simplified algorithm with more details to ease its implementation and provide additional intuition on how it works. In our experiments, this algorithm performs significantly better than its theoretical guarantee, consistently achieving approximation ratios of around 2.

1 Introduction

Cluster deletion is a problem that belongs to the class of *edge modification problems* [1]. Cluster deletion consists in finding the fewest edges from an input graph that have to be deleted so the graph becomes a cluster graph, i.e., a disjoint union of cliques.

Cluster deletion was first introduced by Ben-Dor et al. [2], later formalized and proved to be NP-hard by Natanzon et al. [1], and then proved to be APX-hard by Shamir et al. [3]. Numerous results have been obtained for cluster deletion, which use fixed-parameter tractable algorithms to improve its time complexity [4, 5, 6]. Many of these algorithms use sophisticated heuristics to choose the set of edges to delete. Other results provide approximation algorithms for this problem, with the best being a 2-approximation factor [7] obtained through the relaxation of the cluster deletion linear program.

Cluster deletion has been extensively researched together with other edge modification problems. A related problem is the min clique-complement problem, which minimizes the number of deleted edges to obtain a single clique Bar-Yehuda [8] propose a linear time 2-approximation for this problem. A more general edge modification problem is edge edition, also known as correlation clustering. In correlation clustering, the nodes are clustered according to their similarity scores, which are given as positive edge weights if the nodes are similar or negative weights if they are different. The task is then defined as obtaining clusters that maximize the similarities and minimize the differences. Cluster deletion can be seen as a special case of correlation clustering when negative edges have an infinite penalty, thus requiring each cluster to be a clique and minimizing deleting positive edges.

Correlation clustering has a significant advantage over other clustering techniques, such as k-means, in that it can be applied without fixing the number of clusters. Instead, the algorithm will return the number of clusters that maximizes its objective. This has applications for clustering documents, such as web pages, or in computational biology [9, 10], especially for DNA physical mapping and clustering of genes based on similar expression profiles. Cluster deletion can be applied when clusters can only be formed between adjacent vertices, which may represent similar genes or other items.

There are two main approaches for approximation algorithms for correlation clustering, algorithms that focus on linear program relaxations or based on the Pivot algorithm. Algorithms that rely on solving a linear programming relaxation achieve the best approximation factors, such as a

[*] The code can be found at `https://github.com/vibalcam/cluster-deletion-approx`

$2.06 - \epsilon$ obtained by Chawla et al. [11]. However, solving the linear program is costly, which makes these algorithms inefficient when dealing with large graphs. Instead, approximation algorithms based on Pivot are faster. Pivot was proposed by Ailon et al. [12] and provides a randomized 3-approximation algorithm for complete unweighted correlation clustering. Building on Pivot, Veldt [13] proposed a combinatorial randomized 4-approximation algorithm for cluster deletion by leveraging the relationship between correlation clustering and strong triadic closure (STC). He also provided a derandomized version of the algorithm but only implemented and showed results for the randomized version.

In this work, the following has been accomplished:

- We have simplified and rewritten the deterministic clustering deletion algorithm proposed by Veldt [13], providing more details to ease its implementation
- We have provided additional insights and intuition on how the algorithm works
- We have implemented the algorithm and showed its performance on a set of graphs. We have also used these results to compare it with its randomized version

2 Preliminaries

Correlation clustering. Given a node set V and two sets of nonnegative weights W^+ and W^- defined on pairs of nodes that represent their level of similarity and difference, respectively. The goal of correlation clustering is to find clusters such that discrepancies are minimized. We consider a discrepancy of w_{ij}^- to occur when i and j are clustered together, and a discrepancy of w_{ij}^+ when i and j are in different clusters. Correlation clustering can be defined by the following integer linear program:

$$\begin{aligned} \text{minimize} \quad & \sum_{(i,j)\in\binom{V}{2}} w_{ij}^+ x_{ij} + w_{ij}^-(1-x_{ij}) \\ \text{subject to} \quad & x_{jk} + x_{ik} \geq x_i j \text{ for triplets } i,j,k \\ & x_{ij} \in \{0,1\} \text{ for } (i,j) \in \binom{V}{2} \end{aligned} \tag{1}$$

where $x_{ij} = \begin{cases} 0 & \text{if nodes } i \text{ and } j \text{ are in the same cluster} \\ 1 & \text{otherwise} \end{cases}$

Cluster deletion. Given a graph $G = (V, E)$, cluster deletion minimizes the amount of edges in a set $D \subseteq E$ such that $G' = (V, E \setminus D)$ is a cluster graph. In other words, it minimizes the amount of edges that have to be deleted to convert G into a disjoint union of cliques. This can be viewed as a special case of correlation clustering where only edges can be deleted. Following the notation from equation 1, cluster deletion can be reduced to correlation clustering where $(w_{ij}^+, w_{ij}^-) = (1, 0)$ if $(i, j) \in E$, and $(w_{ij}^+, w_{ij}^-) = (0, \infty)$ if $(i, j) \notin E$. In other words, it can be reduced to a correlation clustering problem where non-adjacent nodes cannot be clustered together ($x_{ij} = 0$ has a $w_{ij}^- = \infty$ cost), and where separating adjacent nodes into different clusters ($x_{ij} = 1$) produces a cost of 1. As a result, we obtain clusters where all nodes are adjacent, i.e., cliques, while minimizing the separations, i.e., the number of edges deleted. color=yellow]maybe include lp for cluster deletion

Open wedges. Given an unsigned graph $G = (V, E)$, a triplet of nodes (i, j, k) is an open wedge centered at k if $(j, k) \in E$ and $(i, k) \in E$ but $(i, j) \notin E$. We will denote the set of triplets that define open wedges as $\mathcal{W}$, and the set of open wedges centered at k as $\mathcal{W}_k \subseteq \mathcal{W}$.

Gallai graph [14]. The Gallai graph $\mathcal{G}$ of G is another graph that has a node v_{ij} for each edge $(i, j) \in G$, and an edge between v_{ik} and v_{jk} if $(i, j, k) \in \mathcal{W}_k$.

Strong Triadic Closure (STC) Labeling. Given a graph $G = (V, E)$, the STC labeling problem labels each edge in E as either weak or strong such that at least one of the edges in each open wedge is weak. In other words, it labels the edges such that each wedge has at least one weak edge. This is based on the STC principle, which states that it is not possible for two individuals to have a strong relationship and not know each other. The minimum weakness strong triadic closure (minSTC) problem [15] minimizes the number of weak edges while obtaining a strong triadic closure labeling of graph G.

Vertex Cover. Given a graph $G = (V, E)$, a vertex cover $\mathcal{C}$ is a set of nodes such that all edges in E have at least one endpoint in $\mathcal{C}$. In other words, a set of nodes such that all edges are covered.

Matching. Given a graph $G = (V, E)$, a matching $\mathcal{M}$ is a set of edges such that no two edges share a common endpoint. In other words, a set of node disjoint edges.

3 4-Approx Deterministic Cluster Deletion Algorithm

The algorithm presented for cluster deletion is mainly based on two claims [13]:

- A vertex cover in the Gallai graph of G is a feasible STC labeling $|E_W|$ for G
- Running Pivot on graph G with the edges from the STC labeling removed returns a feasible solution to cluster deletion. Moreover, it returns a solution with at most $2|E_W|$ deleted edges in expectation

The following claim will also be used:

Claim 1. *If graph G has no open wedges, then any connected component is a clique.*

Proof. For a connected component with 3 or less nodes, this fact is trivial. For a connected component with 4 or more nodes, let us assume it is not a clique. Thus, at least two nodes $iandj$ do not share an edge. Since they both belong to the same connected component, there must be a path $\{i, \cdots, v_i, \cdots, j\}$. However, since there are no open wedges, all triplets of adjacent nodes (v_{i-1}, v_i, v_{i+1}) in the path must form a triangle. This entails that node v_i can be removed from the path, and it will still be a path from i to j. If we apply this logic iteratively, since the path is finite, we will arrive at a path between i and j with three nodes $\{i, v_k, j\}$. Since there is no edge between i and j, this is an open wedge, which contradicts our assumption that there are no open wedges.

□

Algorithm. The simplified deterministic 4-approximation algorithm for cluster deletion is given in Algorithm 1. The algorithm follows these steps:

1. Get STC labeling E_W (2-approx)
   (a) Build Gallai graph $\mathcal{G}$
   (b) Get maximal matching in Gallai graph $\mathcal{M}$
   (c) Get vertex cover using $\mathcal{M}$ (2-approx)
2. Get clustering $\{C_1, \cdots, C_{nc}\}$ by running deterministic Pivot on $\hat{G} = (V, \hat{E})$ with $\hat{E} = E - E_W$. While there are unclustered nodes:
   (a) Calculate scores for each node by looping through all open wedges
   (b) Choose the node with minimum score as pivot
   (c) Cluster pivot and neighbors
   (d) Update graph by removing the clustered nodes

Overview. The main idea behind the algorithm is to find an STC labeling E_W and run the deterministic Pivot algorithm on $G' = (V, E - E_W)$. Therefore, the first step is finding an STC labeling E_W. As mentioned, an STC labeling can be obtained by building the Gallai graph and finding a vertex cover. To find a vertex cover, we use a linear time 2-approximation algorithm based on a maximal matching. We then remove the edges labeled as weak ($G' = (V, E - E_W)$) from the original graph and run the deterministic Pivot algorithm. The deterministic Pivot algorithm iteratively selects a node as pivot, clusters that node together with its neighbors, and deletes them from the graph. This is done until all nodes have been clustered. The pivot node is chosen by scoring each node according to a cost (numerator) and budget (denominator). The node with the lowest $\frac{numerator}{denominator}$ is chosen as pivot. The scoring is done by looking at each open wedge (i, j, k) centered at k in G' and:

- $denominator[k] += 1$. This makes node k more likely to be chosen as pivot
- $numerator[i] += 1$ and $numerator[j] += 1$. This makes nodes i and j less likely to be chosen as pivot

If no open wedges are left in $\hat{G}$, the remaining clusters are the connected components of $\hat{G}$ (Claim 1). This means that no matter which unclustered node is chosen as pivot, the result will be the same. Thus, we choose as pivot any unclustered node left in $\hat{G}$.

The 4-approximation nature of this algorithm comes from the approximation algorithm used to find an STC labeling (2-approximation for vertex cover on Gallai graph) and from using Pivot on $G' = (V, E - E_W)$ (2-approximation for cluster deletion).

Correctness. The correctness of this algorithm follows from the fact that an STC labeling labels at least one edge of each wedge as weak. Since we remove the edges labeled as weak from the graph, all open wedges in G are also removed. This ensures that, when Pivot takes the pivot node p and its neighbors to form a cluster, for every pair of nodes $i, j \in \mathcal{N}(p)$, the edge $(i, j) \in E$. This must be true since, otherwise, (i, j, p) would be an open wedge that has not been removed, which is a contradiction. Therefore, the cluster formed by pivot is a clique in the original graph, which makes the clustering returned a feasible solution to cluster deletion.

A fact that follows this same logic is that each open wedge $(i, j, k) \in \hat{G}$ is a triangle in the original graph G. This provides an alternative way of proving the correctness of the algorithm. For every $i, j \in \mathcal{N}(p)$, the triplet (i, j, p) is either a triangle or an open wedge in $\hat{G}$. However, in either case, it will be a triangle in G, making the clustering a clique.

Intuition pivot node. From the fact that each open wedge in $\hat{G}$ is a triangle in the original graph G, we can derive some intuition on how the pivot node is chosen. Observe that choosing as pivot a node that is the center of an open wedge in $\hat{G}$ entails that triangle in G being clustered together. Therefore, maximizing the number of wedges centered on the pivot node seems to be a good idea for choosing a pivot that will provide a large clique and not split it into multiple smaller cliques. This intuition is consistent with the fact that Deterministic Pivot scores the nodes in a way that makes nodes that are the center of wedges more likely to be selected.

This intuition refers to the denominator of the score. The score also has a numerator that increases every time a node is part of a wedge but not the center. Observe that all nodes that are not part of a wedge are not scored (score of $\frac{0}{0}$). However, following the same logic as Claim 1, the connected components to which these nodes belong are cliques in $\hat{G}$, so the moment in which they are clustered has no effect on the solution obtained[1].

4 Experiments

We have implemented the cluster deletion 4-approximation deterministic algorithm and tested its performance on a set of graphs. We have also tested the randomized version in this same set of graphs to compare their approximation ratios. The set of graphs tested and the implementation for the randomized version have been obtained from the work[2] by Veldt [13].

Table 1 presents the results obtained in our experiments. We can observe that the ratios obtained are similar to those obtained by the randomized version for most of the graphs tested. Even though the deterministic algorithm is ensured to return a 4-approximation, in practice, the values obtained for the tested graphs are consistently around 2, which is a significantly better approximation ratio.

5 Conclusions and Future Work

In this work, we have simplified the clustering deletion deterministic 4-approximation algorithm by Veldt [13]. We have also rewritten the simplified algorithm providing more details to ease its implementation. We have also provided instructions for some edge situations, such as when there are no wedges left. We have also implemented it and run it on multiple graphs to test its performance and compare it to the randomized version. We have observed that it consistently returns around a 2-approximation ratio, which is better than the approximation theoretical guarantee.

An interesting open question is whether it is possible to tighten the approximation guarantee of the algorithm from 4 to 2. This work would also benefit from developing faster and better approximations

[1]The algorithm clusters them last when no wedges are left in $\hat{G}$

[2]The repository is available at `https://github.com/nveldt/FastCC-via-STC`

Algorithm 1 DetPivotViaStcCD(G)

Input. Graph $G = (V, E)$
Output. Feasible clustering for the cluster deletion problem on G

Reduce: Build Gallai graph $\mathcal{G} = (V_{\mathcal{G}}, E_{\mathcal{G}})$
Match: Find maximal matching $\mathcal{M} \subseteq E_{\mathcal{G}}$
Cover: Get vertex cover $\mathcal{C} = \{v_{ij} \in V_{\mathcal{G}} : v_{ij} \in w \text{ for some } w \in \mathcal{M}\}$
STC Labeling: $E_W = \{(i,j) \in E : v_{ij} \in \mathcal{C}\}$
Construct graph $\hat{G} = (V, \hat{E})$ where $\hat{E} = E - E_W$
$nc \leftarrow 0$
while $|V| > 0$ **do** ▷ while there are unclustered nodes
  $nc+ = 1$
  $numerator \leftarrow$ array of zeros of length $|V|$
  $denominator \leftarrow$ array of zeros of length $|V|$
  for $(i,j,k) \in \mathcal{W}$ **do** ▷ for each wedge
    $denominator[k]+ = 1$
    $numerator[i]+ = 1$
    $numerator[j]+ = 1$
  end for
  if $|\mathcal{W}| == 0$ **then** ▷ if no wedges in G
    $p \leftarrow$ any unclustered node ▷ choose pivot
  else
    $p \leftarrow argmin(numerator/denominator)$ ▷ choose pivot
  end if
  $C_{nc} \leftarrow \mathcal{N}(p) \cup \{p\}$ ▷ form cluster
  $G = G - C_{nc}$ ▷ remove clustered nodes from graph
end while
return $\{C_1, \cdots, C_{nc}\}$ ▷ return clusters

Table 1: Lower (LB) and upper (UB) bounds, and approximation ratios (Ratio) for the deterministic (Det) and randomized (Rand) algorithms for cluster deletion via STC. The runtime (Run) of the deterministic algorithm is included together with the number of nodes and edges of each graph

Graph	LB	UB Det	UB Rand	Ratio Det	Ratio Rand	|V|	|E|	Run (s)
Karate	36	71	71	1.97	1.97	34	78	0.01
Dolphins	72	145	143	2.01	1.99	62	159	0.02
Les-Miserables	92	164	187	1.78	2.03	77	254	0.04
PolBooks	211	424	414	2.01	1.96	105	441	0.04
Adjnoun	211	418	422	1.98	2.00	112	425	0.06
Football	256	498	538	1.95	2.10	115	613	0.13
NetScience	315	546	669	1.73	2.12	379	914	0.45
Erdos991	674	1353	1338	2.01	1.99	446	1413	0.22
Celegans-Metabolic	966	1933	1917	2.00	1.98	453	2025	0.48
Harvard500	776	1535	1548	1.98	1.99	500	2043	0.57
Celegans-Neural	1062	2130	2117	2.01	1.99	297	2148	0.25
Roget	1788	3583	3571	2.00	2.00	994	3640	0.78
SmaGri	2410	4856	4811	2.01	2.00	1024	4916	1.25
Email	2616	5230	5169	2.00	1.98	1133	5451	2.30
PolBlogs	8336	16690	16660	2.00	2.00	1222	16714	5.55

for STC labeling. Improvements in solving this problem could be directly applied to the algorithm presented. Lastly, it would be interesting to look at the possibility of adapting the algorithm or the logic used for this problem to solve similar graph problems, such as finding a maximum clique.

References

[1] A. Natanzon, R. Shamir, and R. Sharan, "Complexity classification of some edge modification problems," *Discrete Applied Mathematics*, vol. 113, no. 1, pp. 109–128, Sep. 2001. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0166218X00003917

[2] A. Ben-Dor, R. Shamir, and Z. Yakhini, "Clustering Gene Expression Patterns," *Journal of Computational Biology*, vol. 6, no. 3-4, pp. 281–297, Oct. 1999. [Online]. Available: http://www.liebertpub.com/doi/10.1089/106652799318274

[3] R. Shamir, R. Sharan, and D. Tsur, "Cluster graph modification problems," *Discrete Applied Mathematics*, vol. 144, no. 1-2, pp. 173–182, Nov. 2004. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0166218X04001957

[4] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier, "Graph-Modeled Data Clustering: Fixed-Parameter Algorithms for Clique Generation," in *Algorithms and Complexity*, ser. Lecture Notes in Computer Science, R. Petreschi, G. Persiano, and R. Silvestri, Eds. Berlin, Heidelberg: Springer, 2003, pp. 108–119.

[5] P. Damaschke, "Bounded-Degree Techniques Accelerate Some Parameterized Graph Algorithms," in *Parameterized and Exact Computation*, ser. Lecture Notes in Computer Science, J. Chen and F. V. Fomin, Eds. Berlin, Heidelberg: Springer, 2009, pp. 98–109.

[6] S. Böcker and P. Damaschke, "Even faster parameterized cluster deletion and cluster editing," *Information Processing Letters*, vol. 111, no. 14, pp. 717–721, Jul. 2011. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0020019011001232

[7] N. Veldt, D. F. Gleich, and A. Wirth, "A Correlation Clustering Framework for Community Detection," in *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*. Lyon, France: ACM Press, 2018, pp. 439–448. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3178876.3186110

[8] R. Bar-Yehuda, "A Linear Time 2-Approximation Algorithm for the Min Clique-Complement Problem," Computer Science Department, Technion, Tech. Rep. CS Technion report CS0933, 1998. [Online]. Available: http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-info.cgi/1998/CS/CS0933

[9] M. Golumbic, H. Kaplan, and R. Shamir, "On the Complexity of DNA Physical Mapping," *Advances in Applied Mathematics*, vol. 15, no. 3, pp. 251–261, Sep. 1994. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0196885884710098

[10] K. Cirino, S. Muthukrishnan, N. S. Narayanaswamy, and H. Ramesh, "Graph editing to bipartite interval graphs: Exact and asymptotic bounds," in *Foundations of Software Technology and Theoretical Computer Science*, S. Ramesh and G. Sivakumar, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 37–53.

[11] S. Chawla, K. Makarychev, T. Schramm, and G. Yaroslavtsev, "Near Optimal LP Rounding Algorithm for Correlation Clustering on Complete and Complete k-partite Graphs," Jun. 2015. [Online]. Available: http://arxiv.org/abs/1412.0681

[12] N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: Ranking and clustering," in *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*, ser. STOC '05. New York, NY, USA: Association for Computing Machinery, May 2005, pp. 684–693. [Online]. Available: https://doi.org/10.1145/1060590.1060692

[13] N. Veldt, "Correlation Clustering via Strong Triadic Closure Labeling: Fast Approximation Algorithms and Practical Lower Bounds," in *Proceedings of the 39th International Conference on Machine Learning*. PMLR, Jun. 2022, pp. 22 060–22 083. [Online]. Available: https://proceedings.mlr.press/v162/veldt22a.html

[14] V. B. Le, "Gallai graphs and anti-Gallai graphs," *Discrete Mathematics*, vol. 159, no. 1-3, pp. 179–189, Nov. 1996. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0012365X9500109A

[15] S. Sintos and P. Tsaparas, "Using strong triadic closure to characterize ties in social networks," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York New York USA: ACM, Aug. 2014, pp. 1466–1475. [Online]. Available: https://dl.acm.org/doi/10.1145/2623330.2623664