Correlation Clustering Beyond the Pivot Algorithm

Soheil Behnezhad [*1] **Moses Charikar** [*2] **Vincent Cohen-Addad** [*3] **Alma Ghafari** [*1] **Weiyun Ma** [*2]

Abstract

We study the classic correlation clustering problem. Given n objects and a complete labeling of the object-pairs as either "similar" or "dissimilar", the goal is to partition the objects into arbitrarily many clusters while minimizing disagreements with the labels.

A classic PIVOT algorithm for this problem, due to Ailon et al. (STOC'05), obtains a 3-approximation for this problem. Over the years, this algorithm has been successfully implemented in various settings. The downside of the PIVOT algorithm is that the approximation analysis of 3 is tight for it. While better approximations have been achieved in some settings, these algorithms are often hard to implement in various settings. For example, Behnezhad et al. (FOCS'19) showed that the output of PIVOT can be maintained in polylog time per update in a dynamic setting, a bound that was improved to constant by Dalirrooyfard et al. (ICML'24). But obtaining a better approximation remains open.

In this paper, we present MODIFIEDPIVOT, an algorithm that locally improves the output of PIVOT. Our MODIFIEDPIVOT algorithm can be implemented just as efficiently as PIVOT in various settings. Our experiments show that the output of MODIFIEDPIVOT on average makes less than 77% of the mistakes made by PIVOT. More surprisingly, we prove theoretically that MODIFIEDPIVOT has approximation ratio $3 - \varepsilon_0$ for some absolute constant $\varepsilon_0 > 0$. This, e.g., leads to a better than 3 approximation in the dynamic setting in polylog time, improving the 3-approximation obtained by Behnezhad et al. (FOCS'19).

[*]Equal contribution [1] Northeastern University [2] Stanford University [3]Google Research. Correspondence to: Soheil Behnezhad <s.behnezhad@northeastern.edu>, Moses Charikar < moses@cs.stanford.edu>, Vincent Cohen-Addad <vcohenad@gmail.com>, Alma Ghafari <ghafari.m@northeastern.edu>, Weiyun Ma < wyma@stanford.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

1. Introduction

Correlation clustering is a quintessential problem in data analysis, machine learning, and network science, where the task is to cluster a set of objects based on pairwise relationships. Each pair of objects is labeled as either "similar" or "dissimilar," and the goal is to produce clusters that best align with these labels. Formally, given n vertices and their pairwise labels, the task is to partition them into arbitrarily many clusters so as to minimize the number of dissimilar labels inside clusters plus the number of similar labels that go across clusters. This problem has applications for various tasks such as image segmentation (Kim et al., 2014), community detection (Shi et al., 2021), disambiguation tasks (Kalashnikov et al., 2008), automated labeling (Agrawal et al., 2009; Chakrabarti et al., 2008), and document clustering (Bansal et al., 2002), among others.

The correlation clustering problem was introduced by Bansal, Blum, and Chawla (2002; 2004), who showed that a (large) constant approximation can be achieved in polynomial time. There has been a series of polynomial-time algorithms improving the approximation ratio (Charikar et al., 2003; Ailon et al., 2005; 2008; Chawla et al., 2014; Cohen-Addad et al., 2022; 2023), with the current best known being the 1.437-approximation by Cao, Cohen-Addad, Lee, Li, Newman, and Vogl (2024). It is also known that the problem is APX-hard (Charikar et al., 2003).

The 3-Approximation Barrier

A particularly simple and influential algorithm for correlation clustering is the PIVOT algorithm of Ailon, Charikar, and Newman (2005). The PIVOT algorithm is remarkably simple: it picks a random vertex v, clusters it with vertices that are similar to v, then removes this cluster and recurses on the remaining vertices.

In (Ailon et al., 2005), it was shown that PIVOT obtains a 3-approximation for correlation clustering. Thanks to its simplicity, variants of the PIVOT algorithm have been efficiently implemented in various models, leading to 3- or almost 3-approximations. Examples include the fully dynamic model with polylogarithmic (Behnezhad et al., 2019) or constant Dalirrooyfard, Makarychev, and Mitrovic (2024) update-time, constant rounds of the strictly sublinear massively parallel computations (MPC) model (Cohen-Addad et al., 2021; Assadi & Wang, 2022; Behnezhad et al., 2022), a single-pass of the semi-streaming model (Cambus et al., 2024; Chakrabarty & Makarychev, 2023), distributed local and congest models (Behnezhad et al., 2022), and the classic RAM model where PIVOT takes linear-time to implement.

Unfortunately, the 3-approximation analysis of the PIVOT algorithm is tight. That is, there are various inputs on which the PIVOT algorithm does not obtain any better than a 3-approximation. Because of this, and the fact that all better approximations require solving large linear programs,[1] the 3-approximation has emerged as a barrier for correlation clustering in various settings. In the case of dynamic inputs, for example, the following problem has remained open for more than 5 years since the paper of (Behnezhad et al., 2019):

Open Problem 1. Is it possible to maintain a $3 - \Omega(1)$ approximation of correlation clustering in $\log^{O(1)} n$ time per update?

We note that the problem above has been open even if one allows a much larger update-time of, say, linear in n.

Our Contribution

We show how to break the 3-approximation bound by introducing a new algorithm, MODIFIEDPIVOT, which we formalize as Algorithm 1. Our algorithm modifies the output of PIVOT by locally moving some vertices to other existing clusters or new singleton clusters. We present an analysis showing that this modification does indeed improve the approximation to below 3. Importantly, our criteria for these local moves is extremely simple. This allows the MODIFIEDPIVOT algorithm to be implemented as efficiently as the pivot algorithm in the dynamic setting.

Theorem 1.1 (**Fully Dynamic**)**.** *There is an algorithm that maintains a* 2.99*-approximate correlation clustering by spending* $(poly \log n)$ *time per label update against an oblivious adversary. The bounds on the update-time and the approximation hold in expectation.*

The proof of the Theorem 1.1 appears in Appendix B Theorem 1.1 resolves Open Problem 1.

We also implement MODIFIEDPIVOT and compare its output with PIVOT on various publicly available data sets. Our empirical data suggests that MODIFIEDPIVOT makes less than 77% of the mistakes made by PIVOT on average. See Section 6.

[1] We also note that an independent work of (Cohen-Addad et al., 2024) proposes a correlation clustering algorithm that obtains a better than 3-approximation which can be implemented in some settings (such as sublinear time). But it is unclear if the same algorithm can be implemented in the dynamic setting. The techniques in the two papers are very different.

2. Our Techniques

In this section, we describe the informal intuition behind our new MODIFIEDPIVOT algorithm.

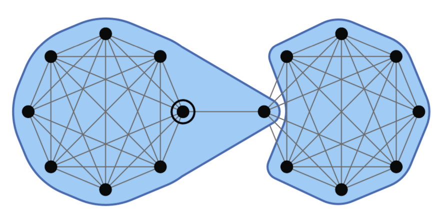
As standard, we model the input to correlation clustering as a graph $G = (V, E)$ with the vertex set V corresponding to the objects and the edge-set E representing the similar labels. In particular, an edge $(u, v) \in E$ implies u and v are similar and a non-edge $(u, v) \notin E$ implies u and v are dissimilar.

It would be useful to start with the PIVOT algorithm and discuss a few examples on which it only obtains a 3-approximation. We will then discuss how MODIFIED-PIVOT overcomes all of these examples and breaks the 3-approximation barrier.

With the graphic view discussed above, the PIVOT algorithm works as follows. It iteratively picks a vertex v, clusters v with its remaining neighbors, then removes this cluster from the graph. This continues until all vertices are removed.

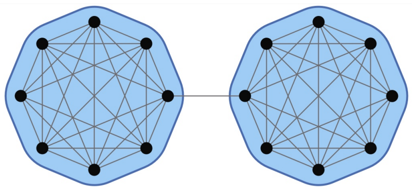
Problem 1: PIVOT **Clusters Dissimilar Pairs.** Our first example shows a scenario where the PIVOT algorithm, mistakenly, clusters together vertices that have very different neighborhoods. Such mistakes alone cause the PIVOT algorithm to pay 3 times the optimum cost in these examples.

Consider a graph composed of two disjoint cliques each on $n/2$ vertices connected by one edge (u, v). The optimal solution is to put the two cliques in disjoint clusters, paying only a cost of one for the edge (u, v). In fact, this is exactly the clustering that PIVOT reports so long as its first PIVOT is not one of the endpoints of the edge (u, v). However, if one of the endpoints of the edge (u, v) is selected as the first pivot, then the algorithm puts u and v in the same cluster, paying a cost of $n - 2$. The figure below illustrates this. On the left hand side, we have the optimal clustering. On the right hand side, we have the output of PIVOT if one of the endpoints of the edge connecting the two cliques is picked as a pivot.



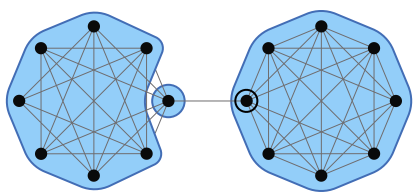
Note that the probability that one of u or v is chosen as the first pivot is $2/n$, therefore, the expected cost of PIVOT in this example is

$$\Pr[\text{first pivot} \notin \{u, v\}](1) + \Pr[\text{first pivot} \in \{u, v\}](n-2)$$
$$= (1 - 2/n) + \frac{2}{n}(n-2) \xrightarrow[n\to\infty]{} 3,$$



which is 3 times the optimum cost.

Fixing Problem 1: Moving Dissimilar Neighbors to Singleton Clusters. Our idea for fixing Problem 1 is a natural one. Whenever our MODIFIEDPIVOT algorithm picks a pivot v, we do not necessarily put all of its remaining neighbors in the cluster of v. Instead, if a neighbor u of v has a very different neighborhood than v, we move it to a singleton cluster. More formally, for some small constant $\delta > 0$, we first define the set D_v to include neighbors u of v such that $|N(u) \cap N(v)| \lesssim \delta N(v)$, where $N(x)$ denotes the neighbor-set of vertex x in the current graph. Note that for sufficiently small δ, a vertex $u \in D_v$ has non-edges to nearly all neighbors of v – so it can only improve the cost if we move such vertices to singleton clusters.

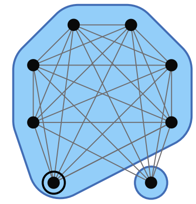


Let us now run this modified algorithm on the example of Problem 1. As before, if the first pivot is not one of the endpoints of (u, v), then the algorithm returns the optimal solution with a cost of 1. But now if one of the endpoints of (u, v) is picked as the first pivot, the other endpoint will move to a singleton cluster. It can be confirmed that the cost is only $n/2$ in this case. Therefore, the expected cost of the algorithm in this case will now be improved to 2 since

$$\Pr[\text{first pivot} \notin \{u, v\}](1) + \Pr[\text{first pivot} \in \{u, v\}](n/2) \\ = (1 - 2/n) + \frac{2}{n}(n/2) \leq 2.$$

Problem 2: PIVOT **Separates Similar Pairs.** It turns out that moving vertices to singleton clusters is not enough. Our next bad example for the PIVOT algorithm shows a scenario where the PIVOT algorithm, mistakenly, separates vertices that have to be clustered together, causing it to pay 3 times the optimum cost.

Consider a graph on n vertices where all pairs are edges except one pair (u, v) which is a non-edge. The optimum solution here is to put everything in the same cluster, paying only a cost of one for the non-edge. This is exactly what



the PIVOT algorithm does too, except when the first pivot chosen is one of the endpoints of the non-edge. In this case, the other endpoint of the non-edge will be put in a singleton cluster, resulting in a cost of $n-2$ as illustrated in the figure of the right hand side.

Note that the expected cost is 3 times the optimum cost of 1 in this case too, since:

$$\Pr[\text{first pivot} \notin \{u, v\}](1) + \Pr[\text{first pivot} \in \{u, v\}](n-2) \\ = (1 - 2/n) + \frac{2}{n}(n-2) \xrightarrow[n \to \infty]{} 3.$$

Fixing Problem 2: Moving Non-Neighbors to Pivot's Cluster. To fix Problem 2, whenever we pick a pivot v, we would like to identify a set A_v of non-neighbors of v whose neighborhoods are similar to $N(v)$ and move them to the cluster of v as well.

The problem with doing so is that the set A_v may be too large, and moving them all to the cluster of v will completely change its structure. This is best described via an example. Consider a complete bipartite graph with vertex parts V_1, V_2 where $|V_2| \gg |V_1|$. Here the solution that puts all vertices in singleton clusters pays a cost of $|V_1| \cdot |V_2|$. Therefore, $OPT \leq |V_1| \cdot |V_2|$. But now take the first pivot v, which with probability $|V_2|/(|V_1| + |V_2|) = 1 - o(1)$ belongs to the larger part V_2. Now note that all the rest of vertices in V_2 will have exactly the same neighborhood as v. Moving them all to the cluster of v results in clustering all the vertices of the graph together, resulting in a cost of $\binom{|V_1|}{2} + \binom{|V_2|}{2}$ for the non-edges inside V_1 and V_2. The approximation ratio will then be at least

$$\frac{\binom{|V_1|}{2} + \binom{|V_2|}{2}}{|V_1||V_2|} \geq \frac{\binom{|V_2|}{2}}{|V_1||V_2|} = \frac{|V_2| - 1}{2|V_1|} = \omega(1).$$

In other words, not only moving similar neighbors to the cluster of the pivot does not improve the approximation to below 3, but it worsens it to super-constant.

To fix this problem, we do not move all the vertices in A_v to the cluster of v. Instead, we subsample some $\delta|N(v)|$ vertices in A_v and only move these vertices to v's cluster. It is important to note that in case $|N(v)| \ll |A_v|$, as is the case in the complete bipartite example, we only move $o(1)$ fraction of the vertices of A_v to the cluster of v. Had this been a constant, our analysis would have been much simpler. However, we will need a much more global analysis to argue that in case A_v is much larger than $N(v)$, then the output of PIVOT is already better than 3-approximate.

The Final Analysis: Up to this point, we've presented a number of instances where the approximation ratio of the PIVOT algorithm is no better than 3. We've also explored some local improvements that would improve the approximation on these instances. What remains to show is that these local improvements do indeed beat 3-approximation on all inputs.

Our analysis follows the standard framework of charging mistakes on *bad triangles*, but has an important twist. As standard, we say three vertices $\{u, v, w\}$ form a bad triangle if exactly two of the pairs $\{u, v\}$, $\{u, w\}$, $\{v, w\}$ are edges. It's important to note that regardless of how these vertices are clustered, at least one pair within a bad triangle must be incorrectly clustered. Consequently, if we can identify β edge-disjoint bad triangles within G, then we can infer that the optimum cluster cost is at least β. This holds even if we identify a *fractional* packing of bad triangles (Ailon et al., 2005). This naturally provides a framework for analyzing the approximation ratio of correlation clustering algorithms, where the mistakes made by the algorithm are blamed on bad triangles. The crux of the analysis will then be focused on formalizing the charging scheme, i.e., which triangle to charge for each mistake and analyzing how many times each pair (edge or non-edge) is charged.

The charging scheme used for the PIVOT algorithm by (Ailon et al., 2005) is highly local, in the sense that it charges any mistake to a bad triangle involving this mistake. Our charging scheme (formalized as Algorithm 2) differs from this in two crucial ways:

- **Charging triangles fractionally:** Instead of charging a single bad triangle *integrally* for each mistake, we charge various bad triangles *fractionally*. In other words, there is no one-to-one mapping between our mistakes and the triangles charged. Instead, we argue that sum of the charges to the bad triangles in total is as large as the mistakes we make (Lemma 4.5), and that sum of the charges involving each pair is not too large (Lemma 4.6).

- **Charging non-local triangles:** When a pivot v is picked in our MODIFIEDPIVOT algorithm, unlike the analysis of (Ailon et al., 2005), we do not just charge bad triangles involving the pivot. For instance, in the example of the complete bipartite graph discussed above, we charge many bad triangles that do not involve the pivot. This is the key in our analysis to show that when A_v is too large compared to C_v, the output of PIVOT is already good.

3. The MODIFIEDPIVOT Algorithm

Our MODIFIEDPIVOT algorithm is formalized below as Algorithm 1. We emphasize that, the time complexity of MODIFIEDPIVOT algorithm and the PIVOT algorithm are the same once the parameters of the algorithm are fixed.

Let us provide some intuition about MODIFIEDPIVOT. Similar to PIVOT, it iteratively picks a random pivot v, and based on it identifies the following sets:

- C_v: This is the set of neighbors of v still in the graph plus vertex v itself. This is exactly the cluster that PIVOT would output for v, but we will modify it.

- D_v: These are vertices that belong to C_v but have very different neighborhood than C_v. Intuitively, we would like to move vertices of D_v to singleton clusters instead of putting them in the cluster of v.

- D'_v: This is a subsample of D_v. Instead of moving all vertices of D_v to singleton clusters, we only move vertices of D'_v to singleton clusters to make sure that the cluster of v does not dramatically differ from C_v in size.

- A_v: These are vertices that are not adjacent to the pivot v, but their neighborhoods are almost the same as C_v. Moving each of these vertices to C_v will improve our cost, provided that we do not move too many of them inside.

- A'_v: This is a subsample of A_v. We only move vertices of A'_v to the cluster of v to ensure, again, that the cluster of v remains relatively close to C_v in size.

- A: The set A is initially empty. Whenever we pick a pivot v, we move all the vertices of A_v to A. We define this set because we do not want a vertex w to participate in A_v and A_u for two different pivots u and v.

The following observation shows that the output of MODIFIEDPIVOT is a valid clustering. What remains is to analyze its approximation ratio, which we do in Section 4.

Observation 1. The output of Algorithm 1 is always a valid clustering. That is, each vertex belongs to exactly one cluster of the output with probability 1.

Proof. First, observe that for every i, the set of vertices removed from V in the first i iterations of Algorithm 1 is identical to the set of vertices clustered in the first i iterations of PIVOT under the same random coin tosses. Since Algorithm 1 only removes a vertex from V if it has been clustered (either in the same iteration or an earlier iteration), this means that every vertex gets clustered at some point

Algorithm 1 The MODIFIEDPIVOT algorithm.

Parameters: $\varepsilon \in (0, \frac{1}{14}]$, $\delta \in [4\varepsilon, \frac{2}{7}]$, $k \geq 1$.
$A \leftarrow \emptyset$.
while $V \neq \emptyset$
Pick a vertex $v \in V$ u.a.r. and mark it as a *pivot*.
Let $C_v \leftarrow \{v\} \cup N(v)$, where $N(v)$ is the set of neighbors of v still in V.
Let $D_v \leftarrow \{u \mid u \in N(v) \text{ and } |N(u) \cap C_v| \leq \delta|C_v| - 1\}$.
Let D'_v include $\min\{|D_v|, \lfloor\delta|C_v|\rfloor\}$ vertices of D_v uniformly at random.
Let $A_v := \{w \mid w \in V \setminus C_v \text{ and } w \notin A \text{ and } |N(w)\Delta C_v| \leq \varepsilon|C_v| - 1\}$.
Let A'_v include $\min\{|A_v|, \lfloor\delta|C_v|\rfloor\}$ vertices of A_v uniformly at random.
Put each vertex of $(D'_v \setminus A) \cup (A_v \setminus A'_v)$ in a singleton cluster.
Put all vertices of $(C_v \cup A'_v) \setminus (D'_v \cup A)$ in the same cluster.
$A \leftarrow A \cup A_v$.
Remove vertices of C_v from V. ▷ We emphasize that even though vertices in A_v get clustered here, they are not removed from V in this step and so can be picked as pivots later on. To clarify this, the vertices in A_v will be allowed to be picked as pivots later on, but even if they start clusters they won't themselves be added to those clusters and will be put in singleton clusters.

in Algorithm 1. Moreover, if a vertex is clustered in some iteration of Algorithm 1, then it is either removed from V or added to the set A at the end of that iteration. Since Algorithm 1 never clusters a vertex that has been removed from V or is already in A, this means that a vertex cannot be clustered more than once. Thus Algorithm 1 always outputs a valid clustering. □

4. Analysis of MODIFIEDPIVOT

In this section, we analyze the approximation ratio of the MODIFIEDPIVOT algorithm, proving the following theorem:

Theorem 4.1. *The clustering output by the* MODIFIEDPIVOT *algorithm has cost at most* 2.997 *times the optimal cost in expectation.*

Remark 4.2. We note that we have not tried to optimize the approximation ratio in Theorem 4.1 as our main contribution is the qualitative result that there is an algorithm maintaining the properties of the pivot algorithm while ensuring a better approximation.

The analysis still fits into the framework of charging *bad triangles* as in the original 3-approximation analysis of the PIVOT algorithm (Ailon et al., 2008). However, the triangles charged in our analysis are very different from (Ailon et al., 2008). We first provide the needed background on charging bad triangles in Section 4.1, then formalize our analysis using this framework in Section 4.2.

4.1. Background on Charging Bad Triangles

Let us first overview the framework of *charging bad triangles* (Ailon et al., 2008). We say three distinct vertices $\{a, b, c\}$ in V form a *bad triangle* if exactly two of the pairs $\{a, b\}, \{a, c\}, \{b, c\}$ belongs to E. Let BT be the set of all bad triangles in the graph.

Definition 4.3. Let $\mathcal{A}$ be a (possibly randomized) algorithm for correlation clustering. We say an algorithm $\mathcal{S}$ is a *charging scheme of width w for $\mathcal{A}$* if for every given output clustering $\mathcal{C}$ of $\mathcal{A}$ and every bad triangle $t \in BT$, algorithm $\mathcal{S}$ specifies a real $y_t \geq 0$ such that:

1. $\sum_t y_t \geq \text{cost}(\mathcal{C})$.
2. For every distinct $u, v \in V$ (which may or may not belong to E), it holds that

$$\mathbf{E}_{\mathcal{A}}\left[\sum_{t \in BT: u,v \in t} y_t\right] \leq w.$$

The following lemma shows why charging schemes are useful.

Lemma 4.4. *Let $\mathcal{A}$ be any (possibly randomized) correlation clustering algorithm. If there exists a charging scheme of width w for $\mathcal{A}$, then for the clustering $\mathcal{C}$ produced by $\mathcal{A}$,*

$$\mathbf{E}_{\mathcal{A}}[\text{cost}(\mathcal{C})] \leq w \cdot \text{opt}(G).$$

Lemma 4.4 is a standard result in the literature and follows from a simple primal dual argument. See for example (Ailon et al., 2005) or (Behnezhad et al., 2022) for its proof.

4.2. Our Charging Scheme for Algorithm 1

We formalize our charging scheme for MODIFIEDPIVOT in Algorithm 2. Algorithm 2 proceeds exactly like MODIFIEDPIVOT and defines all the sets used by MODIFIEDPIVOT in

Algorithm 2 Our charging scheme for MODIFIEDPIVOT. This algorithm is only used for the analysis of Algorithm 1.

Parameters: $\varepsilon \in (0, \frac{1}{14}]$, $\delta \in [4\varepsilon, \frac{2}{7}]$, $k \geq 1$
$A \leftarrow \emptyset$
while $V \neq \emptyset$
  Pick a vertex $v \in V$ uniformly at random and mark it as a pivot
  $C_v \leftarrow \{v\} \cup N(v)$, where $N(v)$ is the set of neighbors of v still in V
  $D_v \leftarrow \{u \mid u \in N(v)$ and $|N(u) \cap C_v| \leq \delta|C_v| - 1\}$
  $D'_v \leftarrow \min\{|D_v|, \lfloor\delta|C_v|\rfloor\}$ vertices of D_v u.a.r.
  $A_v \leftarrow \{w \mid w \in V \setminus C_v, w \notin A, |N(w) \Delta C_v| \leq \varepsilon|C_v| - 1\}$
  $A'_v \leftarrow \min\{|A_v|, \lfloor\delta|C_v|\rfloor\}$ vertices of A_v u.a.r.
  for every $(u, w) \notin E$ such that $u, w \in C_v$
    if $u \notin D'_v$ and $w \notin D'_v$ **then**
      $y_{(v,u,w)} \leftarrow 1$
    else
      $y_{(v,u,w)} \leftarrow 2\delta/(1 - \frac{3}{2}\delta)$
  if $|A_v| \leq k|C_v|$ **then**
    for every $(u, w) \in E$ where $u \in C_v$, $w \in V \setminus C_v$
      if $w \in A$ **then**
        Do not charge a new triangle for (u, w)
      else
        if $w \notin A_v$ **then**
          $y_{(v,u,w)} \leftarrow 1$
        else if $w \in A_v$ **then**
          if $w \in A'_v$ **then**
            $y_{(v,u,w)} \leftarrow \delta$
          else
            $y_{(v,u,w)} \leftarrow 1 + \frac{\varepsilon}{1-\varepsilon}$
  else
    for every $(u, w) \in E$ where $u \in C_v$, $w \in V \setminus C_v$
      if $w \in A$ **then**
        Do not charge a new triangle for (u, w)
      else
        if $w \notin A_v$ **then**
          $y_{(v,u,w)} \leftarrow 1$
        else
          $y_{(v,u,w)} \leftarrow 1 - \frac{\varepsilon}{1-\varepsilon}$
    for every bad triangle (u, w, x) such that $u \in N(v)$, $w, x \in A_v$, $(w, x) \notin E$, $(u, w), (u, x) \in E$
      $y_{(u,w,x)} \leftarrow \frac{5\varepsilon/(1-\varepsilon)}{|A_v|-1}$
  $A \leftarrow A \cup A_v$
  Remove vertices of C_v from V
Output y

forming its clusters. However, instead of returning a clustering, Algorithm 2 returns a charge $y_t \geq 0$ for each bad triangle $t \in BT$. By proving Lemma 4.5 in Appendix A.1 we show that Algorithm 2 charges as many bad triangles as the cost paid by MODIFIEDPIVOT.

Lemma 4.5. *Let y be the vector of charges returned by Algorithm 2 and let $\mathcal{C}$ be the corresponding clustering returned by* MODIFIEDPIVOT *(Algorithm 1). Then it holds that*

$$\sum_{t \in BT} y_t \geq \text{cost}(\mathcal{C}).$$

We then in Appendix A.2 prove Lemma 4.6 that shows Algorithm 2 has width at most 2.997.

Lemma 4.6. *Let y be the charges returned by Algorithm 2.*

For every pair (a,b) of vertices,

$$\mathbf{E}_{\mathcal{A}}\left[y_{(a,b)}\right] \leq 2.997.$$

Combining these lemmas and plugging them into Lemma 4.4 proves Theorem 4.1 that MODIFIEDPIVOT obtains a 2.997-approximation.

Let us start with some intuition on the type of bad triangles we charge based on the sets created in Algorithm 1, and how much we charge each triangle such that it satisfies two properties. First, the total charge of bad triangles is more than the cost of the algorithm, and second, each pair of vertices is charged at most by the approximation guarantee.

Note that, to charge a pair, we need to blame it on a bad triangle. To identify these costs in the proof of Lemma 4.5, at some iteration of the Algorithm 1 we go over all pairs with disagreement, that are distinguished based on C, D, D', A, A' sets corresponding to the pivot of that round. We assign a set of bad triangles to these disagreements, such that the charges on the bad triatngles cover the cost of disagreement.

The last step of the analysis is to show that, each pair is charged at most 2.997. To do so, for a given pair of vertices we need to go over all the bad triangles that is charged and contains this pair. This separates the analysis to different cases based on the choice of the pivot.

5. Implementation in the Fully Dynamic Model

In this section, we prove Theorem 1.1 that a $(3-\Omega(1))$-approximation of correlation clustering can be maintained by spending polylogarithmic time per update. Our starting point is the algorithm of Behnezhad, Derakhshan, Hajiaghayi, Stein, and Sudan (2019) which maintains a randomized greedy maximal independent set, or equivalently, the output of the PIVOT algorithm in polylogarithmic time.

For any vertex v, we draw a real $\pi(v)$ from $[0,1]$ uniformly and independently. We say $\pi(v)$ is the *rank* of v. Recall that the PIVOT algorithm iteratively picks a pivot uniformly from the unclustered vertices and clusters it with its unclustered neighbors. Instead of doing this, we can process the vertices in the increasing order of their ranks, discarding vertices encountered that are already clustered. The resulting clustering is equivalent. We can do the same for MODIFIEDPIVOT as well. Namely, each iteration of the while loop in Algorithm 1 picks the vertex in V with the smallest rank. Again, the resulting clustering is equivalent.

Background on the algorithm of (Behnezhad et al., 2019): The algorithm of (Behnezhad et al., 2019), for each vertex v, maintains the following data structures dynamically:

- $elim(v)$: This represents the pivot by which vertex v is clustered. If v itself is a pivot, then $elim(v) = v$.
- $N^-(v) := \{u \in N(v) \mid \pi(elim(u)) \leq \pi(elim(v))\}$: Intuitively, these are the neighbors of v clustered no later than v. The algorithm stores $N^-(v)$ in a balanced binary search tree where each vertex u is indexed by $\pi(elim(u))$.
- $N^+(v) := \{u \in N(v) \mid \pi(elim(u)) \geq \pi(elim(v))\}$: These are neighbors of v clustered no sooner than v. The algorithm stores $N^+(v)$ in a BST indexed by the static vertex IDs.

Lemma 5.1 (Lemma 4.1 of (Behnezhad et al., 2019))**.** *Let $\mathcal{A}$ be the set of vertices whose pivot changes after inserting or deleting an edge (a,b). There is an algorithm to update all the data structures above in time*

$$\widetilde{O}\left(|\mathcal{A}| \cdot \min\left\{\Delta, \frac{1}{\min\{\pi(a), \pi(b)\}}\right\}\right).$$

Combined with the following lemma also proved in (Behnezhad et al., 2019), this implies that all the data structures can be updated in polylogarithmic time.

Lemma 5.2 (Lemma 5.1 of (Behnezhad et al., 2019))**.** *Let $\mathcal{A}$ be as in Lemma 5.1. It holds for every $\lambda \in (0,1]$ that*

$$\mathbf{E}\left[|\mathcal{A}| \mid \frac{1}{\min\{\pi(a), \pi(b)\}} = \lambda\right] = O(\log n).$$

These two lemmas combined, imply that the update-time is polylogarithmic in expectation. We prove the following claims in Appendix B.

Claim 1. Take vertices u and v such that v is a pivot and $\pi(elim(u)) \geq \pi(v)$. Having access to the data structures above stored by (Behnezhad et al., 2019), it is possible to determine the values of $|N(u) \cap C_v|$ and $|N(u) \Delta C_v|$ exactly in $O(\log n)$ time.

Proof of Theorem 1.1. In addition to the data structures maintained by the algorithm of (Behnezhad et al., 2019), for each vertex u and each $S \in \{C, D, D', A, A'\}$, we store a pointer $I_S(u)$ which takes the value of either a vertex v or $\perp$. If $I_S(u) = v$, this implies that $u \in S_v$. If $I_S(u) = \perp$, then $u \notin S_v$ for any v. For instance, if $I_D(u) = v$, we get that $u \in D'_v$. Note that by having these pointers, we can also immediately maintain the sets $C_v, D_v, D'_v, A_v, A'_v$ for each pivot v. To do so, whenever $I_S(u)$ changes from v to v', we delete u from S_v and insert it to $S_{v'}$. This can be done in $O(\log n)$ time by storing these sets as BSTs.

Claim 2. The data structures $I_C(u)$, $I_D(u)$, $I_A(u)$, $I_{D'}(u)$ and $I_{A'}(u)$ can be maintained in $O(\log n)$ time.

This wraps up the discussion on how we efficiently maintain our data structures. Having all the sets $C_v, D_v, D'_v, A_v, A'_v$

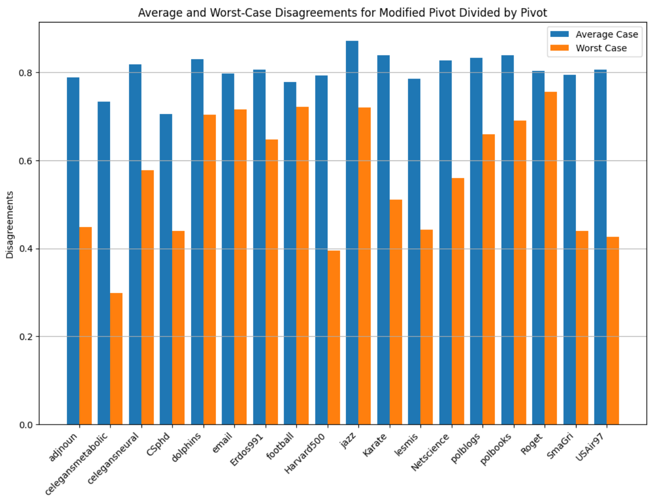


Figure 1. The x-axis of this plot represents each dataset. Note that disagreements of MODIFIEDPIVOT over PIVOT is less than 1, that shows the improvement of our algorithm compared to PIVOT.

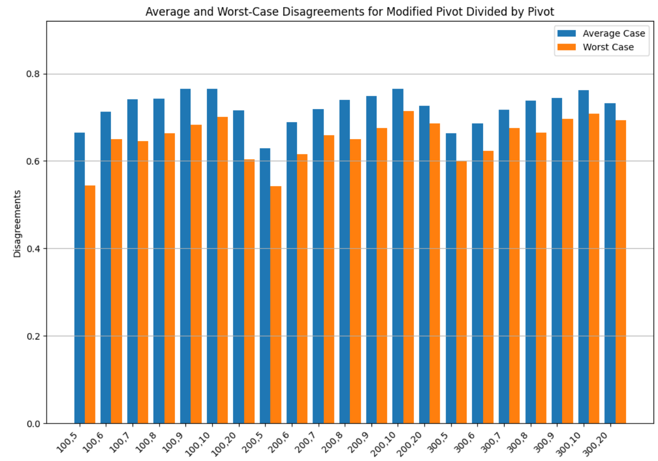


Figure 2. The x-axis of this plot represents stochastic block models with i vertices and j clusters as i, j.

maintained explicitly, we can also maintain the cluster of each vertex formed by Algorithm 1 in polylogarithmic time, concluding the proof of Theorem 1.1. □

6. Empirical Results

In this set of experiments, we compare the objective values of PIVOT and MODIFIEDPIVOT algorithms, ensuring that both utilize the same randomness for the ordering of pivots.

Dataset The first set of data consists of selected graphs, which represent real-life applications in (Davis & Hu, 2011). Our dataset is a selection that covers the following categories: communication networks (email), collaboration networks (Erdos991, Netscience), citation networks(SmaGri), biological networks (celegans-neural, celegans-metabolic) and others (Harvard500, polblogs). Furthermore, this dataset has been featured in other studies on correlation clustering (Veldt, 2022; Veldt et al., 2018).

In most of these graphs, we observe that PIVOT outputs small clusters while ignoring vertices that have a similar neighborhood to the cluster. MODIFIEDPIVOT, however, accounts for these vertices by allowing them to join the cluster when they have a sufficiently similar neighborhood.

By increasing the parameter ε and consequently expanding the size of the A_v set, these sparse instances tend to exhibit smaller objective values. This demonstrates the advantage of MODIFIEDPIVOT in handling sparse real-world graphs.

Stochastic Block Models As a benchmark for dense graphs, we use Stochastic Block Models (SBM). To generate an SBM, we define k clusters (or blocks), each containing a specific number of nodes. The probability of an edge between two nodes within the same cluster is denoted as p_{in}, while the probability of an edge between nodes in different clusters is p_{out}. The steps are as follows: Partition the set of nodes into k disjoint subsets representing clusters. For each pair of nodes within the same cluster, add an edge with probability p_{in}. For each pair of nodes in different clusters, add an edge with probability p_{out}. By setting p_{in} sufficiently large and p_{out} small, we have dense and separable clusters. In our experiments, we set $p_{in} = 0.1$ and $p_{out} = 0.9$.

An optimal correlation clustering algorithm should ideally cluster all nodes within the same block into a single cluster. In these scenarios, PIVOT often fails to achieve this, producing suboptimal clusters. PIVOT does not prune the neighbors of the chosen pivot between to blocks and misses a portion of nodes inside each block.. In contrast, MODIFIEDPIVOT has the strategy to overcome both of this cases.

6.1. Experiments

For these experiments, we generate multiple permutations of vertex orderings to simulate different sequences in which pivots are selected. We emphasize that multiple random seeds per experiment have been used. For each permutation, we select values in the range [0,3, 0.8] for the parameters ε and δ. For each combination of ε and δ, we compute the clustering produced by MODIFIEDPIVOT and calculate its disagreements. We tested at most 8 choices for each of epsilon and delta in all the runs, which adds up to at most 64 combinations. We identify the parameter combination (ε and δ) that yields the minimum disagreements for that specific permutation. We record the best parameters, the resulting clustering, and the corresponding disagreement values. Experiments in Figures 1 and 2 showcase the average and worst-case performance of MODIFIEDPIVOT over PIVOT across all permutations that highlight improvements by MODIFIEDPIVOT over PIVOT. Note that in the plots, values less than 1 represent the number of disagreements of MODIFIEDPIVOT is less than that of PIVOT.

In certain permutations, PIVOT can make drastic mistakes by creating poorly formed clusters (see the example Section 2). In contrast, MODIFIEDPIVOT has the flexibility to account for such bad permutations by adjusting the clusters of pivot, that is more amplified by adjusting the parameters ε and δ. By doing so, MODIFIEDPIVOT accounts for the effects of an unfavorable pivot ordering, resulting in better cluster formations and significantly reduced disagreements. This adaptability highlights the robustness of MODIFIEDPIVOT compared to PIVOT, which is visualized in Figures 1 and 2 by worst-case disagreement ratios.

Acknowledgements

Soheil Behnezhad and Alma Ghafari were supported by NSF CAREER award CCF-2442812 and a Google Research Award.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Agrawal, R., Halverson, A., Kenthapadi, K., Mishra, N., and Tsaparas, P. Generating labels from clicks. In *Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, February 9-11, 2009*, pp. 172–181, 2009.

Ailon, N., Charikar, M., and Newman, A. Aggregating inconsistent information: ranking and clustering. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pp. 684–693. ACM, 2005.

Ailon, N., Charikar, M., and Newman, A. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5):23:1–23:27, 2008.

Assadi, S. and Wang, C. Sublinear time and space algorithms for correlation clustering via sparse-dense decompositions. In *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 - February 3, 2022, Berkeley, CA, USA*, pp. 10:1–10:20, 2022.

Bansal, N., Blum, A., and Chawla, S. Correlation clustering. In *43rd Symposium on Foundations of Computer Science (FOCS 2002), 16-19 November 2002, Vancouver, BC, Canada, Proceedings*, pp. 238, 2002.

Bansal, N., Blum, A., and Chawla, S. Correlation clustering. *Mach. Learn.*, 56(1-3):89–113, 2004.

Behnezhad, S., Derakhshan, M., Hajiaghayi, M., Stein, C., and Sudan, M. Fully dynamic maximal independent set with polylogarithmic update time. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pp. 382–405, 2019.

Behnezhad, S., Charikar, M., Ma, W., and Tan, L. Almost 3-approximate correlation clustering in constant rounds. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pp. 720–731, 2022.

Cambus, M., Kuhn, F., Lindy, E., Pai, S., and Uitto, J. A $(3+\varepsilon)$-Approximate Correlation Clustering Algorithm in Dynamic Streams. In *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024*, 2024.

Cao, N., Cohen-Addad, V., Lee, E., Li, S., Newman, A., and Vogl, L. Understanding the cluster lp for correlation clustering. In *Proceedings of STOC'24*, 2024.

Chakrabarti, D., Kumar, R., and Punera, K. A graph-theoretic approach to webpage segmentation. In *Proceedings of the 17th International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008*, pp. 377–386, 2008.

Chakrabarty, S. and Makarychev, K. Single-pass pivot algorithm for correlation clustering. keep it simple! In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Charikar, M., Guruswami, V., and Wirth, A. Clustering with qualitative information. In *44th Symposium on Foundations of Computer Science (FOCS 2003), 11-14 October 2003, Cambridge, MA, USA, Proceedings*, pp. 524–533. IEEE Computer Society, 2003.

Chawla, S., Makarychev, K., Schramm, T., and Yaroslavtsev, G. Near optimal LP rounding algorithm for correlation clustering on complete and complete k-partite graphs. *CoRR*, abs/1412.0681, 2014.

Cohen-Addad, V., Lattanzi, S., Mitrovic, S., Norouzi-Fard, A., Parotsidis, N., and Tarnawski, J. Correlation clustering in constant many parallel rounds. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 2069–2078. PMLR, 2021.

Cohen-Addad, V., Lee, E., and Newman, A. Correlation clustering with sherali-adams. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pp. 651–661, 2022.

Cohen-Addad, V., Lee, E., Li, S., and Newman, A. Handling correlated rounding error via preclustering: A 1.73-approximation for correlation clustering. In *Proceedings of the 64th Annual Symposium on Foundations of Computer Science (FOCS 2023)*, pp. 123–134. IEEE Computer Society, 2023.

Cohen-Addad, V., Pilipczuk, M., Lolck, D. R., Thorup, M., Yan, S., and Zhang, H. Combinatorial local search. In *Proceedings of STOC'24*, 2024.

Dalirrooyfard, M., Makarychev, K., and Mitrovic, S. Pruned pivot: Correlation clustering algorithm for dynamic, parallel, and local computation models. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=saP7s0ZgYE.

Davis, T. A. and Hu, Y. The university of florida sparse matrix collection. *ACM Trans. Math. Softw.*, 38(1), December 2011. ISSN 0098-3500. doi: 10.1145/2049662.2049663. URL https://doi.org/10.1145/2049662.2049663.

Kalashnikov, D. V., Chen, Z., Mehrotra, S., and Nuray-Turan, R. Web people search via connection analysis. *IEEE Trans. Knowl. Data Eng.*, 20(11):1550–1565, 2008.

Kim, S., Yoo, C. D., Nowozin, S., and Kohli, P. Image segmentation usinghigher-order correlation clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(9):1761–1774, 2014.

Shi, J., Dhulipala, L., Eisenstat, D., Lacki, J., and Mirrokni, V. S. Scalable community detection via parallel correlation clustering. *Proc. VLDB Endow.*, 14(11):2305–2313, 2021.

Veldt, N. Correlation clustering via strong triadic closure labeling: Fast approximation algorithms and practical lower bounds. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S. (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22060–22083. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/veldt22a.html.

Veldt, N., Gleich, D. F., and Wirth, A. A correlation clustering framework for community detection. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, pp. 439–448, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee. ISBN 9781450356398. doi: 10.1145/3178876.3186110. URL https://doi.org/10.1145/3178876.3186110.

A. Analysis of Algorithm 2

A.1. Algorithm 2 Charges Enough Bad Triangles

In this section, we show that Algorithm 2 charges enough bad triangles.

Lemma A.1. *Let y be the vector of charges returned by Algorithm 2 and let $\mathcal{C}$ be the corresponding clustering returned by* MODIFIEDPIVOT *(Algorithm 1). Then it holds that*

$$\sum_{t \in BT} y_t \geq \text{cost}(\mathcal{C}).$$

Proof. We prove by induction that at the end of every iteration i of the while loop, $\sum_{t \in BT} y_t$ upper bounds the number of mistakes made by Algorithm 1 so far. Clearly this holds for the base case $i = 0$.

Now consider iteration $i \geq 1$. The set of vertices newly clustered in this iteration is $C_v \cup A_v \setminus A$. (To avoid ambiguity, any mention of the set A during iteration i in this proof specifically refers to its state before it is updated by A_v in Line 12 of Algorithm 1 or Line 38 of Algorithm 2.) To prove the inductive step, it suffices to show that the number of mistakes newly made by Algorithm 1 in iteration i, which are precisely the mistakes that have at least one endpoint in $C_v \cup A_v$ and no endpoint in A, are upper bounded by the total amount of charge to bad triangles in Lines 12, 14, 21, 24, 26, 33, 35 and 37 in this iteration. Note that each of these mistakes (x, z) satisfies *exactly* one of the following conditions:

1. $(x, z) \notin E$ and $x, z \in C_v \setminus D'_v$.
2. $(x, z) \in E$, $x \in D'_v$ and $z \in C_v \cup A'_v$.
3. $(x, z) \in E$, $x \in C_v$ and $z \in V \setminus (C_v \cup A_v \cup A)$.
4. $(x, z) \notin E$ and $x, z \in A'_v$.
5. Either $(x, z) \in E$, $x \in A'_v$ and $z \in V \setminus (C_v \cup A'_v)$, or $(x, z) \notin E$, $x \in A'_v$ and $z \in C_v \setminus D'_v$.
6. $(x, z) \in E$, $x \in A_v \setminus A'_v$ and $z \in C_v$.
7. $(x, z) \in E$, $x \in A_v \setminus A'_v$ and $z \in V \setminus (C_v \cup A'_v)$.

We refer to the mistakes that satisfy condition (j) as Type (j) mistakes. Let c_j denote the number of mistakes of Type (j) and let y_l denote the total amount of charge to bad triangles in Line l of Algorithm 2 in iteration i. We now prove the following statements (a)-(d) one by one, which collectively imply the inductive step:

1. $c_1 \leq y_{11}$.

   To see this holds, we observe that each Type (1) mistake (x, z) where $(x, z) \notin E$ and $x, z \in C_v \setminus D'_v$ corresponds to a bad triangle (v, x, z) that is charged by 1 in Line 12.

2. $c_2 \leq y_{13}$.

   The total number of Type (2) mistakes (x, z) where $(x, z) \in E$, $x \in D'_v$ and $z \in C_v \cup A'_v$ is at most

$$\sum_{x \in D'_v} (|N(x) \cap C_v| + |A'_v|) \leq |D'_v| \left(\delta|C_v| - 1 + \lfloor \delta|C_v| \rfloor\right) \leq 2\delta|D'_v||C_v|.$$

   On the other hand, the number of pairs $(u, w) \notin E$ such that $u, w \in C_v$ and at least one of u or w is in D'_v, or equivalently, the number of bad triangles (v,u,w) that are charged in Line 14, is equal to

$$\begin{aligned}
&\sum_{u\in D'_v}\left(|(C_v\setminus D'_v)\setminus N(u)|+\frac{1}{2}\left|D'_v\setminus(N(u)\cup\{u\})\right|\right)\\
=&\sum_{u\in D'_v}\left(|C_v\setminus(N(u)\cup\{u\})|-\frac{1}{2}\left|D'_v\setminus(N(u)\cup\{u\})\right|\right)\\
\geq&\left(\sum_{u\in D'_v}(|C_v|-|C_v\cap N(u)|-1)\right)-\binom{|D'_v|}{2}\\
\geq&|D'_v|\left(|C_v|-(\delta|C_v|-1)-1-\frac{1}{2}\left(\lfloor\delta|C_v|\rfloor-1\right)\right)\\
\geq&\left(1-\frac{3}{2}\delta\right)|D'_v||C_v|.
\end{aligned}$$

Thus the total amount of charge in Line 14 is at least

$$\frac{2\delta}{1-\frac{3}{2}\delta}\left(1-\frac{3}{2}\delta\right)|D'_v||C_v|=2\delta|D'_v||C_v|,$$

which upper bounds the total number of Type (2) mistakes.

3. *If* $|A_v|\leq k|C_v|$, *then* $c_3\leq y_{19}$, $c_4+c_5\leq y_{22}$, *and* $c_6+c_7\leq y_{24}$.

   In the case of $|A_v|\leq k|C_v|$, Algorithm 2 charges in Lines 21, 24 and 26. We show the three inequalities separately.

   To see that $c_3\leq y_{19}$, we observe that each Type (3) mistake (x,z) where $(x,z)\in E$, $x\in C_v$ and $z\in V\setminus C_v\setminus A_v\setminus A$ corresponds to a bad triangle (v,x,z) that is charged by 1 in Line 21.

   Next, we show $c_4+c_5\leq y_{22}$. The total number of Type (4) mistakes (x,z) where $(x,z)\notin E$ and $x,z\in A'_v$ is at most

$$\binom{|A'_v|}{2}=\frac{1}{2}|A'_v|(|A'_v|-1)\leq\frac{1}{2}|A'_v|(\lfloor\delta|C_v|\rfloor-1)\leq\frac{\delta}{2}|A'_v||C_v|.$$

   For type (5) mistakes (x,z) where either $(x,z)\in E$, $x\in A'_v$ and $z\in V\setminus C_v\setminus A'_v$, or $(x,z)\notin E$, $x\in A'_v$ and $z\in C_v\setminus D'_v$, note that in both cases we have $z\in N(x)\Delta C_v$. Thus the total number of Type (5) mistakes is at most

$$\sum_{x\in A'_v}|N(x)\Delta C_v|\leq|A'_v|(\varepsilon|C_v|-1)\leq\varepsilon|A'_v||C_v|.$$

   On the other hand, the number of pairs $(u,w)\in E$ such that $u\in C_v$ and $w\in A'_v$, or equivalently, the number of bad triangles (v,u,w) that are charged in Line 24, is equal to

$$\sum_{w\in A'_v}|N(w)\cap C_v|=\sum_{w\in A'_v}|C_v\setminus(N(w)\Delta C_v)|\geq|A'_v|(|C_v|-(\varepsilon|C_v|-1))\geq(1-\varepsilon)|A'_v||C_v|.$$

   Thus the total amount of charge in Line 24 is at least

$$\delta(1-\varepsilon)|A'_v||C_v|\geq(\delta-\varepsilon)|A'_v||C_v|\geq\left(\frac{\delta}{2}+\varepsilon\right)|A'_v||C_v|,$$

   where the last two inequalities follows from $\varepsilon\in(0,\frac{1}{14}]$ and $\delta\in[4\varepsilon,\frac{2}{7}]$. This upper bounds the total number of Type (4) and (5) mistakes.

   Last, we show $c_6+c_7\leq y_{24}$. Note that each Type (6) mistake (x,z) where $(x,z)\in E$, $x\in A_v\setminus A'_v$ and $z\in C_v$ corresponds to a bad triangle (v,z,x) that is charged by $1+\frac{\varepsilon}{1-\varepsilon}$ in Line 26. For each such (v,z,x), we allocate a charge of 1 to cover Type (6) mistakes. It remains to show that the sum of remaining charge of $\frac{\varepsilon}{1-\varepsilon}$ to each of these triangles in Line 26 is sufficient to cover Type (7) mistakes as well. To that end, let us count the number of bad triangles charged in Line 26, which is

$$\begin{aligned}\sum_{w\in A_v\setminus A'_v}|N(w)\cap C_v| &= \sum_{w\in A_v\setminus A'_v}|C_v\setminus(N(w)\Delta C_v)|\\ &\geq |A_v\setminus A'_v|(|C_v|-(\varepsilon|C_v|-1))\\ &\geq (1-\varepsilon)|A_v\setminus A'_v||C_v|.\end{aligned}$$

Thus the total amount of remaining charge we can allocate for Type (7) mistakes is at least

$$\frac{\varepsilon}{1-\varepsilon}(1-\varepsilon)|A_v\setminus A'_v||C_v| = \varepsilon|A_v\setminus A'_v||C_v|.$$

We now show that the total number of Type (7) mistakes does not exceed this amount. Indeed, the total number of Type (7) mistake (x,z) where $(x,z)\in E$, $x\in A_v\setminus A'_v$ and $z\in V\setminus C_v\setminus A'_v$ is at most

$$\sum_{x\in A_v\setminus A'_v}|N(x)\Delta C_v| \leq |A_v\setminus A'_v|(\varepsilon|C_v|-1)\leq \varepsilon|A_v\setminus A'_v||C_v|.$$

4. *If $|A_v|>k|C_v|$, then $c_3\leq y_{30}$ and $c_4+c_5+c_6+c_7\leq y_{31}+y_{33}$.*

   In the case of $|A_v|>k|C_v|$, Algorithm 2 charges in Lines 33, 35 and 37.

   We first show $c_3\leq y_{30}$. To see this holds, we observe that each Type (3) mistake (x,z) where $(x,z)\in E$, $x\in C_v$ and $z\in V\setminus C_v\setminus A_v\setminus A$ corresponds to a bad triangle (v,x,z) that is charged by 1 in Line 33.

   We then show $c_4+c_5+c_6+c_7\leq y_{31}+y_{33}$. Recall that in the case of $|A_v|\leq k|C_v|$, we showed $c_4+c_5+c_6+c_7\leq y_{22}+y_{24}$. Suppose for a moment that Algorithm 2 had charged each bad triangle (v,u,w) in Line 35 by $\max(\delta,1+\frac{\varepsilon}{1-\varepsilon})=1+\frac{\varepsilon}{1-\varepsilon}$. Then by the exactly same argument as we had for the case of $|A_v|\leq k|C_v|$, we could show that $c_4+c_5+c_6+c_7\leq y_{31}$ holds as well. However, in reality, Algorithm 2 only charges an amount of $(1-\frac{\varepsilon}{1-\varepsilon})$ to each bad triangle (v,u,w) in Line 35. Since there are at most $|A_v|$ choices for $w\in A_v$ and at most $(|C_v|-1)$ choices for $u\in C_v\setminus\{v\}$, this results in a total charge deficit of at most $\frac{2\varepsilon}{1-\varepsilon}|A_v|(|C_v|-1)$.

   To cover this deficit, we show that $y_{33}\geq\frac{2\varepsilon}{1-\varepsilon}|A_v|(|C_v|-1)$. To that end, we need to show that Algorithm 2 charges enough bad triangles in Line 37. The total number of triplets (u,w,x) such that $u\in N(v)$ and $w,x\in A_v$ is equal to

$$\binom{|A_v|}{2}(|C_v|-1).$$

   Note that each pair (u,w) where $u\in N(v)$ and $w\in A_v$ can appear in at most $|A_v|-1$ such triplets, and each pair (w,x) where $w,x\in A_v$ can appear in at most $|C_v|-1$ such triplets. Thus the total number of such triplets (u,w,x) that do not satisfy the condition in Line 32 and are not charged in Line 37 is at most

$$\begin{aligned}&\sum_{\substack{(u,w):(u,w)\notin E,\\ u\in N(v),\\ w\in A_v}}(|A_v|-1)+\sum_{\substack{(w,x):(w,x)\in E,\\ w,x\in A_v}}(|C_v|-1)\\ &=\sum_{w\in A_v}\left(\sum_{u\in C_v\setminus N(w)}(|A_v|-1)+\frac{1}{2}\sum_{x\in N(w)\cap A_v}(|C_v|-1)\right)\\ &\leq\sum_{w\in A_v}|N(w)\Delta C_v|\max\left(|A_v|-1,\frac{1}{2}(|C_v|-1)\right)\\ &\leq|A_v|(\varepsilon|C_v|-1)(|A_v|-1),\end{aligned}$$

   where the last inequality follows from $|A_v|>k|C_v|$ and $k\geq 1$. Thus the number of bad triangles charged in Line 37 is at least

$$\binom{|A_v|}{2}(|C_v|-1)-|A_v|(\varepsilon|C_v|-1)(|A_v|-1)\geq(\frac{1}{2}-\varepsilon)|C_v||A_v|(|A_v|-1).$$

Thus the total amount of charge in Line 37 is at least

$$\frac{5\varepsilon/(1-\varepsilon)}{|A_v|-1}(\frac{1}{2}-\varepsilon)|C_v||A_v|(|A_v|-1) \geq \frac{5\varepsilon(1/2-\varepsilon)}{1-\varepsilon}|A_v||C_v| \geq \frac{2\varepsilon}{1-\varepsilon}|A_v||C_v|,$$

where the last inequality follows from $\varepsilon \leq \frac{1}{14}$. This is sufficient to cover the total deficit of at most $\frac{2\varepsilon}{1-\varepsilon}|A_v|(|C_v|-1)$ from Line 35.

We have proved statements (a)-(d) for iteration i. By induction, the proof is complete. □

A.2. Algorithm 2 Has Width Smaller than 3

In this section, we prove that Algorithm 2, for any fixed pair of vertices, charges at most 2.997 bad triangles involving them in expectation. This upper bounds the width of Algorithm 2 by 2.997, and thus combined with Lemma 4.5 and Lemma 4.4 proves that Algorithm 1 obtains a 2.997-approximation.

Let us for every pair (a, b) of the vertices use $y_{(a,b)} := \sum_{t \in BT: a,b \in t} y_t$ to denote the total charges to the bad triangles involving both a and b. Our main result of this section is the following lemma.

Lemma A.2. *Let y be the charges returned by Algorithm 2. For every pair (a, b) of vertices,*

$$\mathbf{E}_{\mathcal{A}}\left[y_{(a,b)}\right] \leq 2.997.$$

In order to prove Lemma 4.6, we start with a number of useful observations. When we say a pair (a, b) of vertices is charged in Algorithm 2, we mean that Algorithm 2 charges some bad triangle involving (a, b).

Observation 2. Except for the bad triangles charged in Line 37 of Algorithm 2, whenever a bad triangle t is charged in Algorithm 2, the pivot v chosen in that iteration must be part of t.

Proof. Follows directly from the description of Algorithm 2. □

Observation 3. Any edge $(a, b) \in E$ is charged in at most one iteration of Algorithm 2. Any non-edge $(a, b) \notin E$ is charged in at most two iterations of Algorithm 2, and in particular, is charged in at most one iteration if none of the charges involving it take place in Line 37.

Proof. First, as shown in Observation 2, except for when a triangle is charged in Line 37 of Algorithm 2, the pivot v must be part of the bad triangle. This means that either a or b should be chosen as the pivot v or at least one of them must be adjacent to v. In either case, at least one of u or v gets removed from V in iteration i. Note that, at least one of (a, b) is corresponded to either u or v, as a result of this at least one of the endpoints of (a, b) is removed from V, and therefore, (a, b) won't be charged again.

Now, if $(a, b) \in E$, consider the case where a bad triangle (u, w, x) is charged in Line 37. In this case, $u \in C_v$ gets removed from V in this iteration but w and x remain in V. Crucially, observe that the two edges of this bad triangle, which are (u, v) and (u, w), are both adjacent to u. Therefore, in this case too, any edge that is part of a charged bad triangle has at least one endpoint removed. Note that, (a, b) is corresponded to either (w, u) or (x, u). This means after charging (a, b) in Line 37 of Algorithm 2, we remove at least one of (a, b) from V, and consequently, we will not charge (a, b) in any future iterations.

If $(a, b) \notin E$, then it can be involved in multiple bad triangles (u, w, x) charged in Line 37 of Algorithm 2 in one iteration. However, we will not be charging this non-edge in Line 37 again in any future iteration of Algorithm 2. This is because we will be appending w and x to the set A, which means that we will not be charging this pair as a member of $A_{v'}$ for a pivot v' in a future iteration. However, we might still charge this non-edge (a, b) in one more future iteration in a single line other than Line 37. □

Let us group the bad triangles charged in Algorithm 2 in iteration i based on the position of the pivot. Note that each charging line in the algorithm processes a particular kind of bad triangle. We define these sets based on whether a bad triangle includes a pivot v or not, and if yes what the adjacency state of v is.

Definition A.3. Let v be the pivot chosen in some iteration i of Algorithm 2. Let X_v be the set of bad triangles t in the graph of iteration i which involve the pivot v and v is adjacent to the other two vertices in t. Let Y_v be the set of bad triangles t in the graph of iteration i which involve the pivot v and v is adjacent to exactly one other vertex of t. Finally, let Z_v be the set of all bad triangles in the graph of iteration i that are charged in this iteration but do not include the pivot v.

Now, we investigate the charges for each type of bad triangles.

Observation 4. By the assumption that pivot v was picked in iteration i of Algorithm 2 it holds that:

1. Any $t \in X_v$ is charged by either one of the Lines 12 and 14 and therefore is charged at most by 1.
2. Any $t \in Y_v$ is charged by either one of the Lines 21, 24, 26, 33 and 35 and therefore is charged at most by $1 + \frac{\varepsilon}{1-\varepsilon}$.
3. Any $t \in Z_v$ is charged $\frac{5\varepsilon/(1-\varepsilon)}{|A_v|-1}$ by only Line 37.

Proof. We prove the three cases one by one below.

1. Note that followed by the charging scheme in Lines 12 and 14 of Algorithm 2 we charge bad triangles including a pivot v and its neighbors u and w in iteration i of the algorithm. That is by description, all the bad triangles in set X_v. Note that the charge of t is bounded by maximum charge of Lines 12 and 14 that is equal to $\max(1, \frac{2\delta}{1-\frac{3}{2}\delta})$. Note that by the choice of parameter $\delta \leq \frac{2}{7}$ in Algorithm 1, we have $\frac{2\delta}{1-\frac{3}{2}\delta} \leq 1$, and therefore, $\max(1, \frac{2\delta}{1-\frac{3}{2}\delta}) = 1$.
2. The structure of triangles in Y_v, is also the same as our charging cases in Lines 21, 24, 26, 33 and 35. Note that we charge bad triangles in iteration i including the pivot v, vertex $u \in C_v$ and, $w \in V \setminus C_v$. In this case, each triangle is charged at most by $\max(\delta, 1, 1 - \frac{\varepsilon}{1-\varepsilon}, 1 + \frac{\varepsilon}{1-\varepsilon}) = 1 + \frac{\varepsilon}{1-\varepsilon}$.
3. Finally, by description any bad triangle in set Z_v is charged by Line 37, we charge each triangle in this set by $\frac{5\varepsilon/(1-\varepsilon)}{|A_v|-1}$.

This completes the proof. □

Definition A.4. We define $N(a)$ in iteration i of Algorithm 1 as the set of the remaining neighbors of a in V.

Definition A.5. Note that, for analyzing different bad triangles containing vertices a and b we need to define the sets where the third vertex c is chosen from. Confirm that vertex c should be in a neighborhood of a or b. We define the following sets based on adjacency of vertex c to a, b, or, both:

$$\begin{aligned} N_a &:= N(a) \setminus (N(b) \cup b), \\ N_b &:= N(b) \setminus (N(a) \cup a), \\ N_{a,b} &:= (N(a) \cap N(b)) \setminus \{a, b\}. \end{aligned}$$

Note that these sets are defined based on the vertices remaining in the graph in iteration i of Algorithm 1.

Definition A.6. Let us define $y_{(a,b),S}$ as the sum of the charges returned from Algorithm 2 for any bad triangle t containing vertices (a, b, c) such that $c \in S$. That is, we define

$$y_{(a,b),S} := \sum_{t \in BT: a,b,c \in t, c \in S} y_t.$$

To prove Lemma 4.6, we need to separate the analysis into two parts. Particularly, the analysis of the edges and non-edges is different, this is because the charging scheme is not symmetric with respect to the adjacency of two vertices.

A.2.1. Width Analysis for Edges

Claim 3. For any $(a, b) \in E$ we have:

1. $\mathbf{E}[y_{(a,b),N_{a,b}}] = 0$,

2. $\mathbf{E}[y_{(a,b)} \mid v \in (N(a)\Delta N(b)) \setminus \{a,b\}] \leq 1 + \frac{4\varepsilon}{1-\varepsilon}$,

3. $\mathbf{E}[y_{(a,b),N_a} \mid v = a] \leq |N_a|$,

4. $\mathbf{E}[y_{(a,b),N_b} \mid v = a] \leq (1 + \frac{\varepsilon}{1-\varepsilon})|N_b|$.

Proof. Here we prove each statement separately.

1. We do not charge t in Algorithm 2 if $c \in N(a) \cap N(b)$, as t will not form a bad triangle.

2. In this case, v is adjacent to exactly one of a or b due to the conditional event $v \in (N(a)\Delta N(b)) \setminus \{a,b\}$. Let us assume without loss of generality that v is adjacent to a. We consider the following three cases which cover all possibilities:
   - $|A_v| \leq k|C_v|$:
     Confirm that, Algorithm 2 implies that in this setting we will only charge bad triangle $t = (a,b,v) \in Y_v$. The charges include Lines 21, 24 and 26. The maximum charge for t is $1 + \frac{\varepsilon}{1-\varepsilon}$.
   - $|A_v| > k|C_v|$ and $b \notin A_v$: In this case if $b \notin A_v$ the only charge that applies to bad triangles t including (a,b) is the charge in Line 33, this bounds the charge of (a,b) by 1 for each choice of the pivot.
   - $|A_v| > k|C_v|$ and $b \in A_v$:
     In this case, there are two types of bad triangles that involve (a,b): bad triangles of type $(a,b,c) \in Z_v$ charged in Line 37 and those of type $(a,b,v) \in Y_v$ charged in Line 35. Note that we charge (a,b,v) in Line 35 by $1 - \frac{\varepsilon}{1-\varepsilon}$. In Line 37, for any vertex x such that $x \in N_a \cap A_v$ we charge (a,b,x) by $\frac{5\varepsilon/(1-\varepsilon)}{|A_v|-1}$. Since $x \in A_v$, there are at most $|A_v| - 1$ choices of x and so the total charge from such triangles involving (a,b) is at most $\frac{5\varepsilon/(1-\varepsilon)}{|A_v|-1} \cdot (|A_v| - 1) = \frac{5\varepsilon}{1-\varepsilon}$. Combined with the charge of $1 - \frac{\varepsilon}{1-\varepsilon}$ incurred in Line 35, this sums up to at most a charge of $1 + \frac{4\varepsilon}{1-\varepsilon}$.

3. In this case, since $v = a$ and a is adjacent to both endpoints of any bad triangle counted in $y_{(a,b),N_a}$, all such bad triangles belong to X_v by Definition A.3. By Observation 4, any bad triangle in X_v is charged at most by 1. Since there are at most $|N_a|$ choices of the third vertex in bad triangles counted in $y_{(a,b),N_a}$ and each is charged by at most 1 as discussed earlier, the total charges sum up to at most $|N_a|$.

4. In this case, since $v = a$ the pivot is adjacent to b and is not adjacent to any vertex $c \in N_b$, this means that all bad triangles t in this form are an element in Y_v by Definition A.3. Note that, by Observation 4 we charge any triangle in Y_v at most by $1 + \frac{\varepsilon}{1-\varepsilon}$. Confirm that, if we fix a, b and the pivot, there are only $|N_b|$ choices for the third vertex of the bad triangles charged in $y_{(a,b),N_b}$ and each triangle is charged by at most $1 + \frac{\varepsilon}{1-\varepsilon}$ as mentioned. Therefore, the total charge of such triangles is at most $(1 + \frac{\varepsilon}{1-\varepsilon})|N_b|$.

This wraps up the proof of Claim 3. □

Claim 4. For any $(a,b) \in E$, it holds that

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] = {} & \Pr[v = a] \cdot \mathbf{E}[y_{(a,b)} \mid v = a] \\ & + \Pr[v = b] \cdot \mathbf{E}[y_{(a,b)} \mid v = b] \\ & + \Pr[v \in (N(a)\Delta N(b)) \setminus \{a,b\}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in (N(a)\Delta N(b)) \setminus \{a,b\}].\end{aligned}$$

where v is the first pivot chosen at some iteration in Algorithm 2 that after processing v, at least one of a or b is removed.

Proof. Let us condition on iteration i of the while loop in Algorithm 2 being the first iteration where at least one of a or b gets removed from V. Note that conditioned on this event, the pivot v of iteration i must be in set $N(a) \cup N(b)$, and note that a and b themselves are part of this set too since $(a,b) \in E$. Moreover, v is chosen uniformly from this set.

By Observation 3, no triangle involving (a,b) is charged before or after iteration i. Thus, it suffices to calculate the expected charge to the triangles of (a,b) exactly in iteration i. For the rest of the proof, we use $N(u)$ to denote the neighbors of any vertex u still in V in iteration i. Let us expand $\mathbf{E}[y_{(a,b)}]$ based on whether the pivot v of iteration i is chosen from the common neighbors of a and b or not. We have:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &= \Pr[v \in N(a)\Delta N(b)] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N(a)\Delta N(b)] \\ &\quad + \Pr[v \in N(a) \cap N(b)] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N(a) \cap N(b)].\end{aligned}$$

First, by Claim 3 we have $\mathbf{E}[y_{(a,b)} \mid v \in N(a) \cap N(b)] = 0$. From this, we get that:

$$\mathbf{E}[y_{(a,b)}] = \Pr[v \in N(a)\Delta N(b)] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N(a)\Delta N(b)].$$

Note that the structure of our analysis varies when pivot v is chosen as vertex a, b, or from the set of $(N(a)\Delta N(b)) \setminus \{a, b\}$. To understand the differences we further expand $\mathbf{E}[y_{(a,b)}]$ conditioning each event describing whether a, b, or a vertex from the union of their neighborhood is chosen as a pivot.

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &= \Pr[v = a] \cdot \mathbf{E}[y_{(a,b)} \mid v = a] \\ &\quad + \Pr[v = b] \cdot \mathbf{E}[y_{(a,b)} \mid v = b] \\ &\quad + \Pr[v \in (N(a)\Delta N(b)) \setminus \{a, b\}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in (N(a)\Delta N(b)) \setminus \{a, b\}].\end{aligned}$$

□

Claim 5. For any $e = (a, b) \in E$ the expected charge on e is at most

$$\frac{(3 + \frac{5\varepsilon}{1-\varepsilon})\,(|N_a| + |N_b|)}{|N_a| + |N_b| + |N_{a,b}| + 2}.$$

Proof. By Claim 4, we expand $\mathbf{E}[y_{(a,b)} \mid v \in N(a)\Delta N(b)]$ as follows:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &= \Pr[v = a] \cdot \mathbf{E}[y_{(a,b)} \mid v = a] \\ &\quad + \Pr[v = b] \cdot \mathbf{E}[y_{(a,b)} \mid v = b] \\ &\quad + \Pr[v \in (N(a)\Delta N(b)) \setminus \{a, b\}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in (N(a)\Delta N(b)) \setminus \{a, b\}].\end{aligned}$$

Here we proceed with exploring each possible event for the pivot using Claim 3. In the case where $v = a$ for any bad triangle including a, b, we charge different values based on the third vertex. Here the charges for each choice of the third vertex c are when $c \in N_a$ and $c \in N_b$:

$$\begin{aligned}\mathbf{E}[y_{(a,b)} \mid v = a] =& \mathbf{E}[y_{(a,b),N_a} \mid v = a] \\ &+ \mathbf{E}[y_{(a,b),N_b} \mid v = a] \leq \left(1 + \frac{\varepsilon}{1-\varepsilon}\right)|N_b| + |N_a|.\end{aligned}$$

By rewriting the above inequality for the case where $v = b$ we have:

$$\begin{aligned}\mathbf{E}[y_{(a,b)} \mid v = b] =& \mathbf{E}[y_{(a,b),N_a} \mid v = b] \\ &+ \mathbf{E}[y_{(a,b),N_b} \mid v = b] \leq \left(1 + \frac{\varepsilon}{1-\varepsilon}\right)|N_a| + |N_b|.\end{aligned}$$

In the last case, where the pivot is not picked as any of a or b, we have:

$$\mathbf{E}[y_{(a,b)} \mid v \in (N(a)\Delta N(b)) \setminus \{a, b\}] \leq 1 + \frac{4\varepsilon}{1-\varepsilon}.$$

Since $\Pr[v = a] = \Pr[v = b] = \frac{1}{|N(a)\cup N(b)|}$ and $\Pr[v \in (N(a)\Delta N(b)) \setminus \{a, b\}] = \frac{|N_a|+|N_b|}{|N(a)\cup N(b)|}$, combining the above inequalities we give the following upper bound for $\mathbf{E}[y_{(a,b)}]$:

$$\mathbf{E}[y_{(a,b)}] \leq \frac{1}{|N_a| + |N_b| + |N_{a,b}| + 2}\left[\left(3 + \frac{5\varepsilon}{1-\varepsilon}\right)(|N_a| + |N_b|)\right]. \quad \square$$

Now, we separate the analysis for three cases, $(C1)-(C3)$, and based on the properties in each case, we determine an upper bound for the expected charge of any edge. We introduce a parameter θ that will be set to minimize the charge over edges. For any of the following cases, we will use Claim 4 to expand the expected charge on each edge. To calculate the expected charge of the edge (a,b) conditioned on any event representing the state of the pivot with respect to the pair of (a,b), we need to determine all the bad triangles charged in Algorithm 2 in iteration i. Note that for the events where $v \in \{a,b\}$, the choices of the third vertex of a bad triangle t in the form of (a,b,c), determines the charges on t.

1. $\max\{|N_a|, |N_b|\} \leq \frac{\theta}{\delta}$.
2. $\max\{|N_a|, N_b|\} > \frac{\theta}{\delta}$, $|N(a) \cap N(b)| + 2 < \frac{\delta}{2-\delta}|N(a) \cup N(b)|$.
3. $\max\{|N_a|, N_b|\} > \frac{\theta}{\delta}$, $|N(a) \cap N(b)| + 2 \geq \frac{\delta}{2-\delta}|N(a) \cup N(b)|$.

Claim 6. In 1, the expected charge on (a,b) is at most $\left(1-\frac{\delta}{\theta+\delta}\right)\left(3+\frac{5\varepsilon}{1-\varepsilon}\right)$.

Proof. To prove the claim, we use the upper bound from Claim 5 and the condition in 1:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] \leq & \frac{1}{|N_a|+|N_b|+|N_{a,b}|+2}\left[\left(3+\frac{5\varepsilon}{1-\varepsilon}\right)(|N_a|+|N_b|)\right] \\ \leq & \left(1-\frac{2}{|N_a|+|N_b|+2}\right)\left(3+\frac{5\varepsilon}{1-\varepsilon}\right) \leq \left(1-\frac{\delta}{\theta+\delta}\right)\left(3+\frac{5\varepsilon}{1-\varepsilon}\right). \quad \square\end{aligned}$$

Claim 7. In 2, the expected charge on (a,b) is at most $3+\frac{5\varepsilon}{1-\varepsilon}-\frac{\theta\delta+\delta^2-\delta}{2(\theta+\delta)}\cdot\frac{2-7\delta}{2-3\delta}$.

Proof. Let us assume that $N(b) \leq N(a)$, by this distinction between a and b, we investigate each event representing different states for pivot:

1. $v = a$:

   In this event, we charge the pair (a,b) for any remaining vertex c in the union of the neighborhood of a and b, this is because, any bad triangle has 2 adjacent vertices, and since we are charging all the bad triangles involving a,b, the third vertex should be either adjacent to a or b. Now, by investigating any choice of vertex $c \in N(a) \cup N(b)$ that creates a bad triangle with a,b, we compute the total charges on a,b. Note that, the different cases affecting the analysis, are related to whether c is picked from N_a, N_b, or $N_{a,b}$, we expand $\mathbf{E}[y_{(a,b)} \mid v=a]$ based on these choices for the third vertex:

$$\begin{aligned}\mathbf{E}[y_{(a,b)} \mid v=a] &= \mathbf{E}[y_{(a,b),N_a} \mid v=a] \\ &+ \mathbf{E}[y_{(a,b),N_{a,b}} \mid v=a] \\ &+ \mathbf{E}[y_{(a,b),N_b} \mid v=a] \\ &\leq \mathbf{E}[y_{(a,b),N_a} \mid v=a] + \left(1+\frac{\varepsilon}{1-\varepsilon}\right)|N_b|.\end{aligned}$$

   Note that the inequality is resulted from Claim 3. Now we explore $\mathbf{E}[y_{(a,b),N_a} \mid v=a]$.

   Since $N(b) \leq N(a)$, and based on the assumption of this claim, we have

$$|N(a)\cap N(b)| + 2 < \frac{\delta}{2-2\delta}(|N_a|+|N_b|) \leq \frac{\delta}{1-\delta}|N_a|.$$

   Moving the terms, this implies

$$(1-\delta)(|N(a)\cap N(b)|+2) < \delta(|N_a|),$$

   which using the fact that $N_a = N(a) \setminus (N(b) \cup b)$ it holds that:

$$|N(a)\cap N(b)| + 1 < \delta(|N(a)|+2) - 1 < \delta|C_v|.$$

Note that the above inequality implies that $|N(a) \cap N(b)| + 1 < \delta|C_v|$ by Algorithm 1, we have $b \in D_a$. Thus, vertex b joins D'_v with probability $\frac{\min\{|D_v|, \lfloor \delta|C_v| \rfloor\}}{|D_v|}$. Here we find a lower bound for this probability using the condition in 2:

$$\frac{\min\{|D_v|, \lfloor \delta|C_v| \rfloor\}}{|D_v|} \geq \frac{\delta|C_v| - 1}{|D_v|} \geq \delta - \frac{\delta}{\theta + \delta}$$

Note that by Observation 3 any edge is charged once, and then at least one of its endpoints is removed from the graph. The only choices of c that change the charging of t depending on whether D'_a contains b or not, are the vertices in N_a. At this step, we can expand $\mathbf{E}[y_{(a,b),N_a} \mid v = a]$ conditioning on state of b with respect to D'_v:

$$\begin{aligned}\mathbf{E}[y_{(a,b),N_a} \mid v = a] = \Pr[b \notin D'_v | v = a] \cdot \mathbf{E}[y_{(a,b),N_a} \mid v = a, b \notin D'_v] \\ + \Pr[b \in D'_v | v = a] \cdot \mathbf{E}[y_{(a,b),N_a} \mid v = a, b \in D'_v].\end{aligned}$$

In the first case, if $b \notin D'_a$: if $c \notin D'_a$ we charge t by Line 12, otherwise we charge it by Line 14. Therefore in this case for each choice of c, we charge t at most 1, and since we have $|N_a|$ such bad triangles then:

$$\mathbf{E}[y_{(a,b),N_a} \mid v = a, b \notin D'_v] \leq |N_a|.$$

In the case where $b \in D'_a$ we always charge t by Line 14. This implies the following:

$$\mathbf{E}[y_{(a,b),N_a} \mid v = a, b \in D'_v] = \frac{2\delta}{1 - \frac{3}{2}\delta}|N_a|.$$

Based on the bounds above, we get:

$$\begin{aligned}\mathbf{E}[y_{(a,b),N_a} \mid v = a] &\leq \left(\left(1 - \frac{\min\{|D_v|, \lfloor \delta|C_v| \rfloor\}}{|D_v|} \right) + \frac{\min\{|D_v|, \lfloor \delta|C_v| \rfloor\}}{|D_v|} \cdot \frac{2\delta}{1 - \frac{3}{2}\delta} \right) |N_a| \\ &\leq \left(1 - \frac{\min\{|D_v|, \lfloor \delta|C_v| \rfloor\}}{|D_v|} \left(1 - \frac{2\delta}{1 - \frac{3}{2}\delta} \right) \right) |N_a| \\ &\leq \left(1 - \frac{\theta\delta + \delta^2 - \delta}{\theta + \delta} \cdot \frac{2 - 7\delta}{2 - 3\delta} \right) |N_a|\end{aligned}$$

2. $v = b$:

   As explored in event $v = a$, we differentiate between triangles by choices of the third vertex in t. Following this we expand $\mathbf{E}[y_{(a,b)} \mid v = b]$:

$$\begin{aligned}\mathbf{E}[y_{(a,b)} \mid v = b] &= \mathbf{E}[y_{(a,b),N_b} \mid v = b] + \mathbf{E}[y_{(a,b),N_{a,b}} \mid v = b] + \mathbf{E}[y_{(a,b),N_a} \mid v = b] \\ &\leq \left(1 + \frac{\varepsilon}{1 - \varepsilon} \right) |N_a| + |N_b|.\end{aligned}$$

   Confirm that the above inequality is simply resulted from Claim 3.

3. $v \in (N(a) \Delta N(b)) \setminus \{a, b\}$:

   Directly by Claim 3 we have:

$$\mathbf{E}[y_{(a,b)} \mid v \in (N(a) \Delta N(b)) \setminus \{a, b\}] \leq \left(1 + \frac{4\varepsilon}{1 - \varepsilon} \right).$$

Finally, we have:

$$
\begin{aligned}
\mathbf{E}[y_{(a,b)}] &= \Pr[v=a]\cdot\left(\left(1-\frac{\theta\delta+\delta^2-\delta}{\theta+\delta}\cdot\frac{2-7\delta}{2-3\delta}\right)|N_a|+\left(1+\frac{\varepsilon}{1-\varepsilon}\right)|N_b|\right)\\
&+\Pr[v=b]\cdot\left[\left(1+\frac{\varepsilon}{1-\varepsilon}\right)|N_a|+|N_b|\right]\\
&+\Pr[v\in(N(a)\Delta N(b))\setminus\{a,b\}]\cdot\left(1+\frac{4\varepsilon}{1-\varepsilon}\right)\\
&=\frac{\left(3+\frac{5\varepsilon}{1-\varepsilon}-\frac{\theta\delta+\delta^2-\delta}{\theta+\delta}\cdot\frac{2-7\delta}{2-3\delta}\right)|N_a|+\left(3+\frac{5\varepsilon}{1-\varepsilon}\right)|N_b|}{|N_a|+|N_b|+|N_{a,b}|+2}.
\end{aligned}
$$

Let $\alpha=\frac{\frac{\theta\delta+\delta^2-\delta}{\theta+\delta}\cdot\frac{2-7\delta}{2-3\delta}}{3+\frac{5\varepsilon}{1-\varepsilon}}$. Now, we give an upper bound on $E[y_{(a,b)}]$ based on α:

$$
\begin{aligned}
E[y_{(a,b)}] &\le \frac{3+\frac{5\varepsilon}{1-\varepsilon}}{|N_a|+|N_b|+|N_{a,b}|+2}\left[(1-\alpha)|N_a|+(1-\frac{\alpha}{2})|N_b|+\frac{\alpha}{2}|N_b|\right]\\
&\le \frac{3+\frac{5\varepsilon}{1-\varepsilon}}{|N_a|+|N_b|+|N_{a,b}|+2}\left[(1-\frac{\alpha}{2})|N_a|+(1-\frac{\alpha}{2})|N_b|\right]\\
&\le \left(3+\frac{5\varepsilon}{1-\varepsilon}\right)\left(1-\frac{\alpha}{2}\right)\\
&= 3+\frac{5\varepsilon}{1-\varepsilon}-\frac{\theta\delta+\delta^2-\delta}{2(\theta+\delta)}\cdot\frac{2-7\delta}{2-3\delta}. \quad \square
\end{aligned}
$$

Claim 8. In 3, the expected charge on (a,b) is at most $\left(1-\frac{\delta}{2-\delta}\right)\left(3+\frac{5\varepsilon}{1-\varepsilon}\right)$.

Proof. Note that by the condition in 3, we have:

$$|N(a)\cap N(b)|+2\ge\frac{\delta}{2-\delta}|N(a)\cup N(b)|,$$

this implies that:

$$|N_a|+|N_b|\le\left(1-\frac{\delta}{2-\delta}\right)|N(a)\cup N(b)|.$$

Using the inequality on the sum of $|N_a|$ and $|N_b|$, and also the upper bound from Claim 5 we have:

$$\mathbf{E}[y_{(a,b)}]\le\frac{1}{|N_a|+|N_b|+|N_{a,b}|+2}\left[\left(3+\frac{5\varepsilon}{1-\varepsilon}\right)(|N_a|+|N_b|)\right]\le\left(1-\frac{\delta}{2-\delta}\right)\left(3+\frac{5\varepsilon}{1-\varepsilon}\right). \quad \square$$

A.2.2. WIDTH ANALYSIS FOR NON-EDGES

Claim 9. For any $(a,b)\notin E$ we have:

1. $\mathbf{E}[y_{(a,b),N_a\cup N_b}]=0$.
2. $\mathbf{E}[y_{(a,b)}\mid v\in N_{a,b}]\le 1$.
3. $\mathbf{E}[y_{(a,b),N_{a,b}}\mid v=a]\le(1+\frac{\varepsilon}{1-\varepsilon})|N_{a,b}|$.

Proof. We prove the three parts one by one.

1. Note that, the triangle $t=(a,b,c)$ such that $c\in N_a\cup N_b$ does not form a bad triangle as there exists only one edge in t.

2. In this case, we have $v \in N_{a,b}$ that means the pivot v is adjacent to both a and b. By Definition A.3 any bad triangle of this structure belongs to the set X_v. By Observation 4 we charge such bad triangles at most by 1. Note that, for any fixed pair of vertices given the pivot, we have one such bad triangle, and therefore the total charge is bounded by 1.

3. Note that any triangle charged in this case is in Y_v. This is because for any fixed pair of non-edge (a, b), any bad triangle charged in $y_{(a,b),N_{a,b}}$ with the condition that $v = a$, we have v is not adjacent to b but it is adjacent to the third vertex c chosen from the set $N_{a,b}$. By Definition A.3 any such bad triangle is in Y_v and is charged at most by $1 + \frac{\varepsilon}{1-\varepsilon}$ as we discussed in Observation 4. Summing up over choices of the third vertex, we get an upper bound of $(1 + \frac{\varepsilon}{1-\varepsilon})|N_{a,b}|$ over charges to all such bad triangles.

The proof is complete. □

Claim 10. The expected charge over a pair of vertices $(a, b) \notin E$ is expandable as follows in case the pair does not belong to the set E:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &= \Pr[v = a] \cdot \mathbf{E}[y_{(a,b)} \mid v = a] \\ &+ \Pr[v = b] \cdot \mathbf{E}[y_{(a,b)} \mid v = b] \\ &+ \Pr[v \in N_{a,b}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N_{a,b}] + \frac{5}{k} \cdot \frac{\varepsilon}{1-\varepsilon}.\end{aligned}$$

where v is the first pivot chosen at some iteration in Algorithm 2 that after processing v, at least one of a or b is removed.

Proof. Let us condition on iteration i of the while loop in Algorithm 2 being the first iteration where at least one of a or b gets removed from V. Note that conditioned on this event, the pivot v of iteration i must be in set $N(a) \cup N(b) \cup \{a, b\}$. Moreover, v is chosen uniformly from this set.

By Observation 3, any triangle involving (a, b) is charged in at most two iterations. We consider the charge from the iteration that results in removing at least one of the endpoints of this pair (iteration i), and sum it up with the maximum possible charge that could have happened in Line 37 of an earlier iteration in Algorithm 2. For the rest of the proof, we use $N(u)$ to denote the neighbors of any vertex u still in V in iteration i.

Let us expand $\mathbf{E}[y_{(a,b)}]$ based on whether the pivot v of iteration i is chosen from the common neighbors of a and b or not. We use $\mathbf{E}[y_{(a,b)} \mid v']$ to denote the additive expected charge for (a, b) resulted from the case where (a, b) is charged once before iteration i, and we use v' to denote the pivot picked at that earlier iteration. Taking this charge into account, it holds that:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &= \Pr[v \in N(a)\Delta N(b)] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N(a)\Delta N(b)] \\ &+ \Pr[v \in N_{a,b} \cup \{a, b\}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N_{a,b} \cup \{a, b\}] + \mathbf{E}[y_{(a,b)} \mid v']\end{aligned}$$

First, note that by Claim 9, $\mathbf{E}[y_{(a,b)} \mid v \in N(a)\Delta N(b)] = 0$. From this, we get that

$$\mathbf{E}[y_{(a,b)}] = \Pr[v \in N_{a,b} \cup \{a, b\}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N_{a,b} \cup \{a, b\}] + \mathbf{E}[y_{(a,b)} \mid v'].$$

Note that the structure of our analysis varies when pivot v is chosen as vertex a, b, or from the set of $N_{a,b}$. To understand the differences we further expand $\mathbf{E}[y_{(a,b)}]$ conditioning on each event describing whether a, b, or a vertex from the intersection of their neighborhood is chosen as a pivot.

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &= \Pr[v = a] \cdot \mathbf{E}[y_{(a,b)} \mid v = a] \\ &+ \Pr[v = b] \cdot \mathbf{E}[y_{(a,b)} \mid v = b] \\ &+ \Pr[v \in N_{a,b}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in N_{a,b}] + \mathbf{E}[y_{(a,b)} \mid v'].\end{aligned}$$

Now, it only remains to prove that $\mathbf{E}[y_{(a,b)} \mid v'] \le \frac{5}{k} \cdot \frac{\varepsilon}{1-\varepsilon}$. Note that, there exists at most one pivot v' charging any non-edge by Line 37 in Algorithm 2, however, for any third vertex c holding the properties of vertex u in Line 37 in iteration where we remove v', we charge (a, b, c) by $\frac{\frac{5\varepsilon}{1-\varepsilon}}{|A_{v'}|-1}$. Since there are most $|C_{v'}|$ choices for c, this gives an upper bound of $\frac{\frac{5\varepsilon}{1-\varepsilon}}{|A_{v'}|-1} \cdot |C_{v'}|$ for this particular charges on (a, b). Since we only charge such bad triangles if $|A_{v'}| > k|C_{v'}|$, this implies

$$\mathbf{E}[y_{(a,b)} \mid v'] \le \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}. \quad \square$$

Claim 11. For any $e = (a, b) \notin E$ the expected charges over e is at most

$$\frac{(3 + \frac{2\varepsilon}{1-\varepsilon})|N_{a,b}|}{|N_a| + |N_b| + |N_{a,b}| + 2} + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}.$$

Proof. Note that by Claim 10 we have:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] = \Pr[v = a] \cdot \mathbf{E}[y_{(a,b)} \mid v = a] \\ + \Pr[v = b] \cdot \mathbf{E}[y_{(a,b)} \mid v = b] \\ + \Pr[v \in N_{a,b}] \cdot \mathbf{E}[y_{(a,b)} \mid v \in] + \frac{5}{k} \cdot \frac{\varepsilon}{1-\varepsilon}.\end{aligned}$$

Here we proceed with exploring each event using Claim 9. In the case where $v = a$ for any bad triangle including a, b, we charge different values based on the third vertex. Here the charges for each choice of the third vertex c are when $c \in N_{a,b}$:

$$\begin{aligned}&\mathbf{E}[y_{(a,b)} \mid v \in \{a, b\}] \\ &= \mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a] + \mathbf{E}[y_{(a,b),N_{a,b}} \mid v = b] \\ &\leq 2(1 + \frac{\varepsilon}{1-\varepsilon})|N_{a,b}|.\end{aligned}$$

For the case that the pivot is picked from the common neighbors of a and b, we get:

$$\mathbf{E}[y_{(a,b)} \mid v \in N_{a,b}] \leq 1.$$

Since $\Pr[v = a] = \Pr[v = b] = \frac{1}{|N(a) \cup N(b) \cup \{a,b\}|}$ and $\Pr[v \in N_{a,b}] = \frac{|N_{a,b}|}{|N(a) \cup N(b) \cup \{a,b\}|}$, combining the above inequalities we give the following upper bound for $\mathbf{E}[y_{(a,b)}]$:

$$\mathbf{E}[y_{(a,b)}] \leq \frac{1}{|N_a| + |N_b| + |N_{a,b}| + 2}\left[\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right)|N_{a,b}|\right] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}. \quad \square$$

Now, we separate the analysis for three cases, $(D1) - (D3)$, and based on the properties in each case, we determine an upper bound for the expected charge of any edge. We introduce a parameter λ that will be set to minimize the charge over non-edges. For any of the following cases, we will use Claim 10 to expand the expected charge on each edge. To calculate the expected charge of the non-edge (a, b) conditioned on any event representing the state of the pivot with respect to the pair of (a, b), we need to determine all the bad triangles charged in Algorithm 2 in iteration i. Note that for the events where $v \in \{a, b\}$, the choices of the third vertex of a bad triangle t in the form of (a, b, c), determines the charges on t.

1. $\min\{|N(a)|, |N(b)|\} \leq \frac{\lambda}{\delta}$.
2. $\min\{|N(a)|, |N(b)|\} > \frac{\lambda}{\delta}$, $|N(a)\Delta N(b)| + 2 < \frac{\varepsilon}{1+\varepsilon}|N(a) \cup N(b)|$.
3. $\min\{|N(a)|, |N(b)|\} > \frac{\lambda}{\delta}$, $|N(a)\Delta N(b)| + 2 \geq \frac{\varepsilon}{1+\varepsilon}|N(a) \cup N(b)|$.

Claim 12. Let us assume that $|N_a| \geq |N_b|$ w.l.o.g. In 1, the expected charge on (a, b) is at most $\frac{\lambda+\delta}{(2\delta+\lambda)}\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}$.

Proof. By Claim 11 and the condition in 1 we have:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] &\leq \frac{1}{|N_a| + |N_b| + |N_{a,b}| + 2}\left[\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right)|N_{a,b}|\right] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} \\ &\leq \left(\frac{|N(b)|}{|N(b)| + 2}\right)\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} \\ &\leq \left(1 - \frac{1}{|(N(b)| + 2}\right)\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} \\ &\leq \frac{\lambda + \delta}{(2\delta + \lambda)}\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}. \quad \square\end{aligned}$$

Note that the second inequality holds since we have $|N(b)| \geq |N_{a,b}|$ and the claim assumption implies $|N(b)| + 2 \leq |N_a| + |N_b| + |N_{a,b}| + 2$. Also, the last inequality holds since we have $|N(b)| \leq \frac{\lambda}{\delta}$, this concludes that $1 - \frac{1}{|N(b)|+2} \leq 1 - \frac{1}{\lambda/\delta+2} = \frac{\lambda+\delta}{\lambda+2\delta}$.

Claim 13. In 2, the expected charge on (a, b) is at most

$$\max\left[3 + \frac{2\varepsilon}{1-\varepsilon} + 2\left(-\frac{\delta}{k} + \frac{\delta/k}{\delta+\lambda}\right)\cdot\left(1 + \frac{\varepsilon}{1-\varepsilon} - \delta\right), 3 - \frac{2\varepsilon}{1-\varepsilon}\right] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}.$$

Proof. Here the analysis varies when pivot v is chosen as vertex a, b, or from the set of $N_{a,b}$. To understand the differences we further expand $\mathbf{E}[y_{(a,b)}]$ by Claim 10 conditioning on whether a or b is chosen as a pivot or not:

$$\begin{aligned}\mathbf{E}[y_{(a,b)}] \leq &\Pr[v = a]\cdot\mathbf{E}[y_{(a,b)} \mid v = a]\\ &+ \Pr[v = b]\cdot\mathbf{E}[y_{(a,b)} \mid v = b]\\ &+ \Pr[v \in N_{a,b}]\cdot\mathbf{E}[y_{(a,b)} \mid v \in N_{a,b}] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}.\end{aligned}$$

We determine all the bad triangles charged in Algorithm 2 in iteration i by investigating each event based on the pivot separately:

1. $v \in \{a, b\}$:

   Now, by checking any vertex $c \in N_{a,b}$, we find about each charging in Algorithm 2 that charges triangle $t = (a, b, c)$. We explore $\mathbf{E}[y_{(a,b)} \mid v = a]$, and note that the analysis for the case where $v = b$ is the same as that for $v = a$. Now, the condition in 2 implies

   $$(1+\varepsilon)(|N_a| + |N_b|) + 2 < \varepsilon|N(a) \cup N(b)|,$$

   which in turn, results in

   $$|N_a| + |N_b| + 2 < \varepsilon(|N_{a,b}| + 2) < \varepsilon|N_{a,b}| + 1.$$

   Note that we have

   $$N(a)\Delta N(b) = N_a \cup N_b.$$

   This implies that:

   $$|N(a)\Delta N(b)| + 1 \leq \varepsilon|N_{a,b}| \leq \varepsilon|N(a)|.$$

   Note that the above inequality implies that $|N(a)\Delta N(b)| < \varepsilon(|N(a)| + 1) - 1 = \varepsilon|C_v| - 1$ and therefore we can conclude $b \in A_v$. Observe that by Algorithm 2, the vertex b joins A'_v with probability $\frac{\min\{|A_v|, \lfloor\delta|C_v|\rfloor\}}{|A_v|}$.Note that $t \in Y_i$, and therefore the charges on different triangles vary whether of $b \in A'_v$ or not. We also have two different charging schemes based on the size of A_v.

   - $|A_v| \leq k|C_v|$: In this case, by the condition in Claim 13, we have $-\frac{1}{k|C_v|} \geq -\frac{1/k}{1+\lambda/\delta}$. Thus we have:

     $$\frac{\min\{|A_v|, \lfloor\delta|C_v|\rfloor\}}{|A_v|} \geq \frac{\delta|C_v| - 1}{k|C_v|} \geq \frac{\delta}{k} - \frac{\delta/k}{\delta+\lambda}.$$

     When the size of A_v is not too large compared to that of C_v, we charge any triangle t by δ if $b \in A'_v$ and $1 + \frac{\varepsilon}{1-\varepsilon}$ otherwise. Based on the probability that b is chosen as a member of A'_v, the expected number of triangles charged containing (a, b) can be written as follows:

     $$\begin{aligned}\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a] = &\Pr[b \notin A'_v | v = a]\cdot\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a, b \notin A'_v]\\ &+ \Pr[b \in A'_v | v = a]\cdot\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a, b \in A'_v].\end{aligned}$$

In the first case, if $b \notin A'_v$ we charge t by Line 26, Therefore in this case for each choice of c, we charge t at most $1 + \frac{\varepsilon}{1-\varepsilon}$, precisely we have:

$$\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a, b \notin A'_v] = \left(1 + \frac{\varepsilon}{1-\varepsilon}\right)|N_{a,b}|.$$

In the case where $b \in A'_v$ we always charge t by Line 24. This implies the following:

$$\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a, b \in A'_v] = \delta|N_{a,b}|.$$

Using the expected charges above the following equality holds:

$$\begin{aligned}
\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a] &= \left(1 - \frac{\min\{|A_v|, \lfloor\delta|C_v|\rfloor\}}{|A_v|}\right)\left(1 + \frac{\varepsilon}{1-\varepsilon}\right)|N_{a,b}| \\
&\quad + \left(\frac{\min\{|A_v|, \lfloor\delta|C_v|\rfloor\}}{|A_v|} \cdot \delta\right)|N_{a,b}| \\
&= \left(1 + \frac{\varepsilon}{1-\varepsilon} - \frac{\min\{|A_v|, \lfloor\delta|C_v|\rfloor\}\left(1 + \frac{\varepsilon}{1-\varepsilon} - \delta\right)}{|A_v|}\right)|N_{a,b}| \\
&\le \left(1 + \frac{\varepsilon}{1-\varepsilon} + \left(-\frac{\delta}{k} + \frac{\delta/k}{\delta+\lambda}\right)\cdot\left(1 + \frac{\varepsilon}{1-\varepsilon} - \delta\right)\right)|N_{a,b}|.
\end{aligned}$$

- $|A_v| > k|C_v|$: When the size of A_v is significantly larger than that of C_v, we always charge triangle t by $1 - \frac{\varepsilon}{1-\varepsilon}$ in Line 35:

$$\mathbf{E}[y_{(a,b),N_{a,b}} \mid v = a] = \left(1 - \frac{\varepsilon}{1-\varepsilon}\right)|N_{a,b}|.$$

2. $v \in N_{a,b}$: Directly by Claim 9 we have:

$$\mathbf{E}[y_{(a,b)} \mid v \in N_{a,b}] \le 1.$$

Finally, we can give an upper bound for the expected charges on (a,b) by the maximum charge in the above cases:

$$\begin{aligned}
\mathbf{E}[y_{(a,b)}] &\le \Pr[v \in \{a,b\}] \cdot \max\left[\left(1 + \frac{\varepsilon}{1-\varepsilon} + \left(-\frac{\delta}{k} + \frac{\delta/k}{\delta+\lambda}\right)\cdot\left(1 + \frac{\varepsilon}{1-\varepsilon} - \delta\right)\right), 1 - \frac{\varepsilon}{1-\varepsilon}\right]|N_{a,b}| \\
&\quad + \Pr[v \in N_{a,b}] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} \\
&= \frac{\max\left[3 + \frac{2\varepsilon}{1-\varepsilon} + 2\left(-\frac{\delta}{k} + \frac{\delta/k}{\delta+\lambda}\right)\cdot\left(1 + \frac{\varepsilon}{1-\varepsilon} - \delta\right), 3 - \frac{2\varepsilon}{1-\varepsilon}\right]}{|N_a| + |N_b| + |N_{a,b}| + 2}|N_{a,b}| + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} \\
&\le \max\left[3 + \frac{2\varepsilon}{1-\varepsilon} + 2\left(-\frac{\delta}{k} + \frac{\delta/k}{\delta+\lambda}\right)\cdot\left(1 + \frac{\varepsilon}{1-\varepsilon} - \delta\right), 3 - \frac{2\varepsilon}{1-\varepsilon}\right] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}. \quad \square
\end{aligned}$$

Claim 14. In 3, the expected charge on (a,b) is at most $\left(1 - \frac{\varepsilon}{1+\varepsilon}\right)\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}$.

Proof. By Claim 11 and the condition in 3 we have:

$$\mathbf{E}[y_{(a,b)}] = \frac{1}{|N_a| + |N_b| + |N_{a,b}| + 2}\left[\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right)|N_{a,b}|\right] \le \left(1 - \frac{\varepsilon}{1+\varepsilon}\right)\left(3 + \frac{2\varepsilon}{1-\varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k}. \quad \square$$

Finally, we are ready to wrap up the proof of Lemma 4.6:

Proof of Lemma 4.6 for any pair (a, b). Now, looking through the width analysis for edges and non-edges, to prove Lemma 4.6, for any case described in Appendix A.2.1 and Appendix A.2.2, we introduce a set of values for parameters $\varepsilon, \delta, \lambda$, and θ that imply a 2.997-approximation. We set $\varepsilon = 0.007$, $\delta = 0.179$, $\lambda = 7.613$, $\theta = 7.055$, and $k = 12.295$.

For any edge in E, we investigate the three cases $(C1) - (C3)$. For each case, we prove that $E[Y_{a,b}] < 2.997$.

- In 1, by the upper bound in Claim 6 and plugging in the parameters with introduced values we get:

$$E[Y_{a,b}] \leq \left(1 - \frac{\delta}{\theta + \delta}\right)\left(3 + \frac{5\varepsilon}{1 - \varepsilon}\right) < 2.961.$$

- In 2, by the upper bound in Claim 7 and plugging in the parameters with introduced values we get:

$$E[Y_{a,b}] \leq 3 + \frac{5\varepsilon}{1 - \varepsilon} - \frac{\theta\delta + \delta^2 - \delta}{2(\theta + \delta)} \cdot \frac{2 - 7\delta}{2 - 3\delta} < 2.996.$$

- In 3, by the upper bound in Claim 8 and plugging in the parameters with introduced values we get:

$$E[Y_{a,b}] \leq 3 + \left(1 - \frac{\delta}{2 - \delta}\right)\left(3 + \frac{5\varepsilon}{1 - \varepsilon}\right) < 2.737.$$

For any non-edge in E, we investigate the three cases $(D1) - (D3)$. For each case, we prove that $E[Y_{a,b}] < 2.997$.

- In 1, by the upper bound in Claim 12 and plugging in the parameters with introduced values we get:

$$E[Y_{a,b}] \leq \frac{\lambda + \delta}{2\delta + \lambda}\left(3 + \frac{2\varepsilon}{1 - \varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} < 2.95.$$

- In 2, by the upper bound in Claim 13 and plugging in the parameters with introduced values we get:

$$E[Y_{a,b}] \leq \max\left[3 + \frac{2\varepsilon}{1 - \varepsilon} + 2\left(-\frac{\delta}{k} + \frac{\delta/k}{\delta + \lambda}\right) \cdot \left(1 + \frac{\varepsilon}{1 - \varepsilon} - \delta\right), 3 - \frac{2\varepsilon}{1 - \varepsilon}\right] + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} < 2.996.$$

- In 3, by the upper bound in Claim 14 and plugging in the parameters with introduced values we get:

$$E[Y_{a,b}] \leq \left(1 - \frac{\varepsilon}{1 + \varepsilon}\right)\left(3 + \frac{2\varepsilon}{1 - \varepsilon}\right) + \frac{\frac{5\varepsilon}{1-\varepsilon}}{k} < 2.997.$$

This concludes the proof of Lemma 4.6. □

B. Implementation in the Fully Dynamic Model

Claim 15. Take vertices u and v such that v is a pivot and $\pi(elim(u)) \geq \pi(v)$. Having access to the data structures above stored by (Behnezhad et al., 2019), it is possible to determine the values of $|N(u) \cap C_v|$ and $|N(u)\Delta C_v|$ exactly in $O(\log n)$ time.

Proof. To see this, recall first that for each vertex $w \in C_v$, we have $elim(w) = v$. Therefore, for any edge $(u, w) \in E$, because of the assumption $\pi(elim(u)) \geq \pi(v)$, it holds that $w \in N^-(u)$. Recalling that $N^-(u)$ is indexed by the eliminator ranks, and noting that in a BST, we can count how many elements are indexed by the same value in $O(\log n)$ time, we get that we can immediately compute the value of $|N(u) \cap C_v|$ in $O(\log n)$ time. Also note that $|N(u)\Delta C_v| = d_u - |N(u) \cap C_v|$, where d_u is the total number of neighbors of u whose eliminator rank is at least $\pi(v)$. Such neighbors of u can be both in $N^-(u)$ and $N^+(u)$. We can count the ones in $N^-(u)$ by simply using the properly indexed BST in $O(\log n)$ time, and can simply sum it up to $|N^+(u)|$ since all neighbors of u in $N^+(u)$ contribute to d_u. This concludes the proof. □

Claim 16. The data structures $I_C(u), I_D(u), I_A(u), I_{D'}(u), I_{A'}(u)$ can be maintained in $O(\log n)$ time.

Proof. Below, we discuss how these data structures can be maintained in the same time as Lemma 5.1.

- $I_C(u)$: Note that $I_C(u)$ is equivalent to $elim(u)$, which is already maintained by (Behnezhad et al., 2019).
- $I_D(u)$: Suppose that $I_D(u) = v$, i.e., $u \in D_v$. An update may change the value of $I_D(u)$ under one of these events: (i) the pivot of u changes, (ii) some vertices leave or are added to C_v, changing the criteria $|N(u) \cap C_v| \leq \delta|C_v| - 1$ for u, or (iii) an edge is inserted or deleted from u to some other vertex in C_v. We discuss how to efficiently update $I_D(u)$ in each of these scenarios.

  (i) Suppose that a vertex v is now marked as a pivot after some update. We argue that we can identify D_v in $O(|C_v| \log n)$ time. To do so, we go over all vertices of C_v one by one, and apply the algorithm of Claim 15 on each to check whether they belong to D_v. Since, from our earlier discussion, we already explicitly maintain C_v which requires $\Omega(|C_v|)$ time when v is marked as a pivot, this only increases the update-time by a $O(\log n)$ factor.

  (ii) Now suppose that a vertex w is added to C_v. In this case, we go over all vertices of $N^+(w)$, and for each one u, recompute the value of $|N(u) \cap C_v|$ in $O(\log n)$ time as discussed to decide whether $I_D(u) = v$. Note that w must belong to set $\mathcal{A}$ (defined in Lemma 5.1), and its neighborhood $N^+(w)$ has size at most $O(\log n/\pi(w))$ (see Proposition 3.1 of (Behnezhad et al., 2019)). Since $\pi(w) \geq \min\{\pi(a), \pi(b)\}$ where (a, b) is the edge update causing this change, the total running time of this step is upper bounded by

  $$\widetilde{O}\left(|\mathcal{A}| \cdot \min\left\{\Delta, \frac{1}{\min\{\pi(a), \pi(b)\}}\right\}\right),$$

  which is also spent by the algorithm of (Behnezhad et al., 2019) (Lemma 5.1). The process for when a vertex w is removed from C_v is similar.

  (iii) In this case, we simply re-evaluate $|N(u) \cap C_v|$, which can be done in $O(\log n)$ using Claim 15.

- $I_A(u)$: To maintain $I_A(u)$, we maintain another pointer $I_A(u, v)$ for every pair of vertices u and v which is 1 iff v is a pivot, $\pi(elim(u)) > \pi(v)$, and $|N(u)\Delta C_v| \leq \varepsilon|C_v| - 1$ (where with a slight abuse of notation, $N(u)$ is the neighbors of u remained in the graph at the time that v is chosen as a pivot). This way, $I_A(u)$ is exactly the vertex v minimizing $\pi(v)$ such that $I_A(u, v) = 1$. So let us see how we maintain $I_A(u, v)$ efficiently.

  An update may change the value of $I_A(u, v)$ under one of these events: (i) whether v is a pivot changes, (ii) some vertices leave or are added to C_v, changing the criteria $|N(u)\Delta C_v| \leq \varepsilon|C_v| - 1$ for u, or (iii) an edge is inserted or deleted from u to some other vertex in C_v. We discuss how to efficiently update $I_A(u, v)$ in each of these scenarios.

  (i) Suppose that v is marked as a pivot after an edge update. We will show how to find all vertices w that satisfy $|N(u)\Delta C_v| \leq \varepsilon|C_v| - 1$ in total time $O((\log^3 n)/\pi(v))$. Since we can afford to spend this much time for every vertex in $\mathcal{A}$ due to Lemma 5.1, this will keep the update-time polylogarithmic.

  To do so, we subsample $\Theta(\log n)$ vertices in C_v without replacement and call it S_v. We then take $\hat{A} = \cup_{x \in S_v} N^+(x)$. Note that we have $|N^+(x)| \leq O(\log n/\pi(x)) \leq O(\log n/\pi(v))$ for each $x \in C_v$ by (see Proposition 3.1 of (Behnezhad et al., 2019)). Hence, $\hat{A}$ has size at most $O(\log^2 n/\pi(v))$. We go over all vertices u in $\hat{A}$ and check, using Claim 15, whether $I_A(u, v) = 1$ by spending $O(\log n)$ time.

  It remains to show that if $I_A(u, v) = 1$, then u must belong to $\hat{A}$. Indeed, we show this holds with probability $1 - 1/poly(n)$. To see this, note that $I_A(u, v) = 1$ iff $|N(u)\Delta C_v| \leq \varepsilon|C_v| - 1$. This means u must be adjacent to at least a constant fraction of vertices in C_v. Since S_v includes $\Theta(\log n)$ random samples from C_v, u is adjacent to at least one with probability $1 - 1/poly(n)$.

  (ii) Suppose a vertex w is added to C_v. In this case, we go over all vertices x of $N^+(w)$ and on each reevaluate whether $I_A(x, v) = 1$ in $O(\log n)$ time using Claim 15. The needed running time is $O(\log n/\pi(w))$ for w. Similar to case (ii) of updating $I_D(v)$, this, overall, takes the same time as in Lemma 5.1 keeping the update-time polylogarithmic.

  (iii) In this case, we just reevaluate $I_A(u, v)$ in $O(\log n)$ time using Claim 15.

- $I_{D'}(u), I_{A'}(u)$: Note that A'_v and D'_v are simply random-subsamples of A_v and D_v respectively. Since we explicitly maintain A_v and D_v, we can also explicitly maintain these random subsamples as efficiently, and thus can maintain $I_{D'}(u)$ and $I_{A'}(u)$ accordingly.

□