
Proceedings Paper:
Peng, P. and Zhang, J. (2021) Towards a query-optimal and time-efficient algorithm for clustering with a faulty oracle. In: Proceedings of the 34th Annual Conference on Learning Theory (COLT 2021). 34th Annual Conference on Learning Theory (COLT 2021), 15-19 Aug 2021, Boulder, Colorado. Proceedings of Machine Learning Research (134). PMLR.

Towards a Query-Optimal and Time-Efficient Algorithm for Clustering with a Faulty Oracle

Pan Peng* Jiapeng Zhang†

Abstract

Motivated by applications in crowdsourced entity resolution in database, signed edge prediction in social networks and correlation clustering, Mazumdar and Saha [NIPS 2017] proposed an elegant theoretical model for studying clustering with a faulty oracle. In this model, given a set of n items which belong to k unknown groups (or clusters), our goal is to recover the clusters by asking pairwise queries to an oracle. This oracle can answer the query that "do items u and v belong to the same cluster?". However, the answer to each pairwise query errs with probability ε, for some $\varepsilon \in (0, \frac{1}{2})$. Mazumdar and Saha provided two algorithms under this model: one algorithm is query-optimal while time-inefficient (i.e., running in quasi-polynomial time), the other is time efficient (i.e., in polynomial time) while query-suboptimal. Larsen, Mitzenmacher and Tsourakakis [WWW 2020] then gave a new time-efficient algorithm for the special case of 2 clusters, which is query-optimal if the bias $\delta := 1-2\varepsilon$ of the model is large. It was left as an open question whether one can obtain a query-optimal, time-efficient algorithm for the general case of k clusters and other regimes of δ.

In this paper, we make progress on the above question and provide a time-efficient algorithm with nearly-optimal query complexity (up to a factor of $O(\log^2 n)$) for all constant k and any δ in the regime when information-theoretic recovery is possible. Our algorithm is built on a connection to the stochastic block model.

1 Introduction

Clustering is a fundamental problem in machine learning with many applications. In this paper, we study an elegant theoretical model proposed by Mazumdar and Saha [2017a] for studying clustering with the help of a faulty oracle. The model is defined as follows:

Model Given a set $V = [n] := \{1, \cdots, n\}$ of n items which contains k latent clusters $V_1, \cdots, V_k$ such that $\cup_i V_i = V$ and for any $1 \leq i < j \leq k$, $V_i \cap V_j = \emptyset$. The clusters $V_1, \ldots, V_k$ are unknown. We wish to recover them by making pairwise queries to an oracle $\mathcal{O}$, which answers if the queried two vertices belong to the same cluster or not. This oracle gives correct answer with probability $1-\varepsilon$, for some $\varepsilon \in (0, \frac{1}{2})$. That is, for any vertices $u, v \in V$, if u and v belong to the same cluster, then

$$\mathcal{O}(u,v) = \begin{cases} + & \text{with probability } 1-\varepsilon, \\ - & \text{with probability } \varepsilon, \end{cases}$$

*Department of Computer Science, University of Sheffield. Email: p.peng@sheffield.ac.uk

†Department of Computer Science, University of Southern California. Email: jiapengz@usc.edu

and if u, v belong to two different clusters, then

$$\mathcal{O}(u,v) = \begin{cases} + & \text{with probability } \varepsilon, \\ - & \text{with probability } 1-\varepsilon. \end{cases}$$

Equivalently, the model can be formalized as follows: Define a function $\tau : V \times V \to \{\pm 1\}$ such that $\tau(u,v) = 1$ if u, v belong to the same cluster and $\tau(u,v) = -1$ if u, v belong to different clusters. For any u, v, let $\eta_{u,v} \in \{\pm 1\}$ be a random noise in the edge observation such that $\mathbf{E}[\eta_{u,v}] = \delta$. The noises $\eta_{u,v}$ are independent for all pairs $u, v \in V$. Then the oracle $\mathcal{O}$ returns the sign of

$$\tau(u,v)\eta_{u,v}$$

when the pair u, v is queried. Note that by the above two formalization, it holds that $1 - \varepsilon = \frac{1}{2} + \frac{\delta}{2}$. In the following, we call δ the *bias* of the model.

It is assumed that repeating the same question to the oracle $\mathcal{O}$, it always returns the same answer. (This was known as *persistent noise* in the literature; see e.g. [Goldman et al., 1990].) Our goal is to recover the latent clusters *efficiently* (i.e., within polynomial time) with high probability by making as few queries to the oracle $\mathcal{O}$ as possible.

Motivations The above model captures several fundamental applications. In the *entity resolution* (also known as the *record linkage*) problem [Fellegi and Sunter, 1969], the goal is to find records in a data set that refer to the same entity across different data sources. Currently fully automated techniques for entity resolution has been unsatisfactory and current crowdsourcing platforms use human in the loop to help improve accuracy (see e.g. [Karger et al., 2011, Wang et al., 2012, Dalvi et al., 2013, Gokhale et al., 2014, Vesdapunt et al., 2014, Mazumdar and Saha, 2017b]). That is, the workers are asked to answer if any two items u, v represent the same entity. It has been noted that the answers from non-expert workers are inevitably noisy. Furthermore, the goal of these crowdsourcing platforms is to use minimal number of queries to reduce cost and time for recovering the entities (clusters), which can be well modelled by the clustering with a faulty oracle.

Another motivation is to predict the signed edges in a social network [Leskovec et al., 2010], where the sign ('$+$' or '$-$') on an edge indicates positive relation or negative relation between the corresponding two nodes. This problem can arise in many scenarios, e.g., voting on Wikipedia [Burke and Kraut, 2008] and making friends on Slashdot [Brzozowski et al., 2008]. Theoretically, there has been a line of work [Chen et al., 2014a, Mitzenmacher and Tsourakakis, 2016] that considers the model that allows the algorithm to query the sign of an edge (u, v), which in turn can indicate whether u, v belongs to the same cluster or not. It is further assumed that the answer to each query is correct with probability $1 - \varepsilon$, for some $\varepsilon \in (0, \frac{1}{2})$. Thus, their model is also well captured by the previous model of clustering with a faulty oracle. There is also some other related work on edge classification [Cesa-Bianchi et al., 2012].

In addition, the model of clustering with noisy oracle is closely related to the problem of correlation clustering. In the correlation clustering problem [Bansal et al., 2004], we are given an undirected signed graph, and our goal is to partition the vertex set into clusters so that the number of agreements[1] is maximized or the number of disagreements[2] is minimized. This problem is NP-hard and several approximation algorithms have been provided. In a variant formalization called noisy correlation clustering [Bansal et al., 2004, Mathieu and Schudy, 2010], after given the ground truth clustering, the sign of each edge is flipped

[1]These are the number of $+$ edges inside clusters plus the number of $-$ edges between clusters.

[2]These are the number of $-$ edges inside clusters plus the number of $+$ edges between clusters.

with some probability ε. If the original graph is complete, then this is exactly the input of the problem of clustering with a faulty oracle.

Finally, the model is strongly connected to the stochastic block model (SBM), which is popular model for studying graph clustering algorithms. In the SBM with parameters N, k, p, q such that $0 \leq q < p \leq 1$, denoted by SBM(N, k, p, q), there is a set V of N vertices with a hidden k-partition $V_1, \cdots, V_k$ such that $\cup_i V_i = V$, where each part V_i is called a *cluster*. A graph G is generated from the SBM(N, k, p, q) model, if for any two vertices $u, v \in V$, an edge is added between u, v with probability p if u, v are from the same cluster, and with probability q if u, v are from two different clusters. There has been a vast amount of research on recovering the underlying clusters from the SBM with different ranges of parameters in the past decade (see the recent survey [Abbe, 2017]). Consider the noisy clustering model with and parameters n, k, δ. Suppose that we make queries on all pairs $u, v \in V$, then the graph G that is obtained by adding all $+$ edges answered by the oracle $\mathcal{O}$ is exactly the graph that is generated from the SBM model with parameters $N = n$, k, $p = \frac{1}{2} + \frac{\delta}{2}$ and $q = \frac{1}{2} - \frac{\delta}{2}$. However, in our problem, our goal is to recover the clusters by making *sublinear* number of queries, i.e., without seeing the whole graph.

State-of-the-art Mazumdar and Saha [2017a] gave an *inefficient* algorithm that perform $O(\frac{nk \log n}{\delta^2})$ queries to the oracle that recovers all the clusters of size $\Omega(\frac{\log n}{\delta^2})$. The query complexity of this algorithm nearly matches an information-theoretic lower bound $\Omega(\frac{nk}{\delta^2})$ presented by the same authors. The running time of their algorithm is $O\left(\left(k \log n/\delta^2\right)^{(\log n)/\delta^2}\right)$, which is quasi-polynomial, and there is an inherent obstacle to push this algorithm to be efficient (see Section 1.3 for more details). Towards efficient algorithms, they designed another algorithm that runs in time $O(\frac{nk \log n}{\delta^2} + k(\frac{k^2 \log n}{\delta^4})^{\omega})$, makes $O(\frac{nk \log n}{\delta^2} + \min\{\frac{nk^2 \log n}{\delta^4}, \frac{k^5 \log^2 n}{\delta^8}\})$ queries and recovers all clusters of size at least $\Omega(\frac{k \log n}{\delta^4})$, where ω is the matrix multiplication exponent.

In a follow-up work, Larsen et al. [2020] proposed an improved algorithm for the case $k = 2$, i.e., two clusters. This algorithm runs in time $O(\frac{n \log n}{\delta^2} + \frac{\log^3 n}{\delta^8})$ and makes $O(\frac{n \log n}{\delta^2} + \frac{\log^2 n}{\delta^6})$ queries. See Table 1 for a comparison of these results.

Note that the above two efficient algorithms are query-suboptimal when δ is small, i.e., $\delta = o(n^{-1/4})$, even for $k = 2$. Due to this, Larsen et al. [2020] raised the following open question:

> "*Can we design a query-optimal, time-efficient algorithm that performs $O(\frac{kn \log n}{\delta^2})$ queries for all $0 < \delta < 1$?*"

It is the main question we are trying to address in this paper. Note that for any non-trivial algorithm with query complexity $O(\frac{nk \log n}{\delta^2})$, it suffices to assume that $\delta \geq (k \log n/n)^{1/2}$, as the maximum number of queries one can make is n^2.

1.1 Our results

We give an algorithm with the following performance guarantee for the problem of clustering with a faulty oracle.

Theorem 1. *There exists a polynomial time algorithm* NOSIYCLUSTERING *that recovers all the clusters of size $\Omega(\frac{k^4 \log n}{\delta^2})$ with success probability $1 - o_n(1)$. The total number queries that* NOSIYCLUSTERING *performs to the faulty oracle $\mathcal{O}$ is $O(\frac{nk \log n}{\delta^2} + \frac{k^{10} \log^2 n}{\delta^4})$.*

# clusters	query complexity	time-efficient ?	reference
k	$O(\frac{nk\log n}{\delta^2})$	**No**	[Mazumdar and Saha, 2017a]
	$O(\frac{nk\log n}{\delta^2}+\min\{\frac{nk^2\log n}{\delta^4},\frac{k^5\log^2 n}{\delta^8}\})$	**Yes**	
	$\Omega(\frac{nk}{\delta^2})$	**Lower bound**	
2	$O(\frac{n\log n}{\delta^2}+\frac{\log^2 n}{\delta^6})$	**Yes**	[Larsen et al., 2020]
k	nearly-balanced: $O(\frac{nk\log n}{\delta^2}+\frac{k^4\log^2 n}{\delta^4})$	**Yes**	**this work**
	$O(\frac{nk\log n}{\delta^2}+\frac{k^{10}\log^2 n}{\delta^4})$	**Yes**	

Table 1: Comparison of algorithms for clustering with a faulty oracle. We say an algorithm is time-efficient, if it runs in polynomial time (in $n, k, 1/\delta$). We stress that all the upper bound holds for algorithms success probability at least $1-o_n(1)$, while the lower bound is for any algorithm with constant success probability.

Note that for any constant k, the query complexity of our algorithm NOISYCLUSTERING in Theorem 1 is

$$O(\frac{n\log n}{\delta^2}+\frac{\log^2 n}{\delta^4})=\begin{cases} O(\frac{n\log n}{\delta^2}) & \text{if } \delta=\omega((\frac{\log n}{n})^{1/2}) \\ O(\frac{n\log^2 n}{\delta^2}) & \text{if } \delta\in[\Omega((\frac{1}{n})^{1/2}),O((\frac{\log n}{n})^{1/2})) \end{cases}$$

Thus, as long as $\delta=\Omega((\frac{1}{n})^{1/2})$ (i.e., δ is in the regime when information-theoretic recovery is possible), our algorithm achieves nearly-optimal query complexity (up to a factor of $O(\log^2 n)$). On the other hand, if $\delta=o((\frac{1}{n})^{1/2})$, it is impossible to recover the latent clusters, which follows from the information-theoretic lower bound $\Omega(\frac{n}{\delta^2})$ and an inherent restriction on the maximum number of queries, i.e., n^2, as there are at most n^2 edges. Therefore, *we almost fully resolve the aforementioned open question by Larsen et al. [2020] for any constant $k\geq 2$.*

The main focus on this paper is to optimize the dependency on δ. We do not attempt to optimize the dependency on k. By combining ideas from Mazumdar and Saha [2017a], we believe it is possible to slightly improve the term k^{10}. However several evidences suggested there is an inherent obstacle to match the information theoretical lower bound by efficient algorithms. See Section 1.3 for more details. The algorithm NOISYCLUSTERING is built upon a simple algorithm for the case that the underlying clustering $V_1,\ldots,V_k$ are nearly-balanced, i.e., each cluster V_i has size $\Omega(\frac{n}{k})$. For the latter case, we achieve a slightly better algorithm. Formally, we define a b-balanced partition as follows.

Definition 2. *Let $b\in[0,1]$. Given a vertex set V and a partition $V_1,\ldots,V_k$ such that $\cup_i V_i=V$, we call $V_1,\ldots,V_k$ a b-balanced partition, if for each i, $|V_i|\geq bn/k$.*

We show the following result for the case that the underlying partition is the b-balanced.

Theorem 3. *Let $b\in(0,1]$. Let $n\geq\frac{C_0k^2\log^2 n}{b^2\delta^2}$ for some constant $C_0>0$. Suppose that the underlying partition $V_1,\ldots,V_k$ of $V=[n]$ is b-balanced. There is a polynomial time algorithm that recovers all the clusters with success probability $1-o_n(1)$. The total number queries that the algorithm performs to the faulty oracle $\mathcal{O}$ is $O(nk\cdot\log n/\delta^2+k^4\cdot\log^2 n/(b^4\delta^4))$.*

For any constant $b>0$, the query complexity of the above algorithm is $O(\frac{k^4\log^2 n}{\delta^4}+\frac{kn\log n}{\delta^2})$, which is in comparison to the information-theoretic lower bound $\Omega(\frac{nk}{\delta^2})$ that also holds for the nearly-balanced instance [Mazumdar and Saha, 2017a]. The query complexity almost matches the lower bound when $k=o((\delta^2\cdot n)^{1/3})$, which leaves open in the range $(\delta^2\cdot n)^{1/3}\leq k\leq\delta^2\cdot n$. Interestingly, there exists evidence suggesting that there is no *efficient* algorithm matching the information theoretical lower bound when k is large. We refer to Section 1.3 for a more detailed discussion.

1.2 Discussion of previous approaches and an overview of our algorithms

We first sketch the main idea underlying the algorithms in [Mazumdar and Saha, 2017a, Larsen et al., 2020]. Their algorithms do the following:

1. select a subset T of $t = \text{poly}(k \log n/\delta)$ vertices, and build a graph $H_T = (T, E_T)$ by making queries for all pairs $u, v \in T$ and defining the edge set E_T according to the query answers;
2. find all sub-clusters X of size $\Omega(\frac{\log n}{\delta^2})$ from the T by making use of the graph H_T, where a set X is a sub-cluster if $X \subseteq V_i$ for some cluster V_i;
3. grow each of the sub-clusters X to V_i: arbitrarily select a subset $X_0 \subseteq X$ of size $\Theta(\frac{\log n}{\delta^2})$ and add all vertices $v \in V$ to X such that the number of '+' neighbors of v in X_0 is more than $\frac{|X_0|}{2}$.

Then the algorithm removes all the identified clusters and repeat the above process if the number of remaining vertices is still large and more clusters need to be identified.

Both of the previous two efficient algorithms are based on some 'local' approaches of finding sub-clusters from H_T (in Step 2 above), i.e., by counting the number of '+' neighbors and/or shared neighbors of vertices in T. Such 'local' approaches require the algorithm to choose a large subset T whose size eventually results in the sub-optimality of the total number of queries to the oracle. We also note that the query-optimal algorithm in [Mazumdar and Saha, 2017a] is a 'global' approach in the sense that it makes use of a large subgraph of H_T to cluster the vertices in T. However their subroutine for finding the subgraph requires quasi-polynomial time, which can not be improved to polynomial time, assuming that the hidden clique problem is hard in average case, which is a well-believed assumption in complexity theory.

Our approach. Our algorithm is built upon the same framework, while uses several new ideas. One of our key observations is that we can make use of the 'global' and time-efficient algorithms for clustering graphs generated from SBM with appropriate parameters to find sub-clusters in the small representative graph H_T, when the input instance is nearly-balanced. Slightly more precisely, note that for any subset $T \subset V$, if we let E_T be the set of all '+' edges from the query answers and let $H_T = (V, E_T)$, then we can equivalently view H_T as generated from the stochastic block model SBM($|T|, k, p, q$) with $p = \frac{1}{2} + \frac{\delta}{2}$, $q = \frac{1}{2} - \frac{\delta}{2}$. Previous research (e.g,. [McSherry, 2001, Vu, 2018]) suggests that if H_T contains k nearly-balanced clusters and the parameters $|T|, p, q, k$ satisfy certain conditions (see Theorem 14), then with high probability, we can efficiently recover all the clusters in T. Now if the original instance $V_1, \ldots, V_k$ is nearly-balanced (i.e., $|V_i| \geq \frac{bn}{k}$, $i \leq k$, for some constant $0 < b < 1$), then we can show that a randomly sample set T with $\Theta(\frac{k^2 \log n}{\delta^2})$ vertices will satisfy both the nearly-balanced requirement of H_T and the condition for clustering SBM. Then by applying one algorithm (specifically, Vu's algorithm; see Theorem 4) for clustering the graph H_T from SBM($|T|, k, p, q$) to find all the sub-clusters $X_1, \ldots, X_k$, and growing each sub-cluster as described before, we obtain our algorithm for clustering the nearly-balanced instance with improved performance guarantee. We give details in Section 3.

For the unbalanced instance, i.e., there exists at least one cluster of size less than $\frac{bn}{k}$, we have to modify this algorithm since unbalanced instance is a barrier to algorithms for the stochastic block model. Our second observation is that there must exist a *size-gap* between different clusters, which allows us to filter out the small size clusters. The remaining large clusters are again nearly-balanced (with different balance ratio), which can be clustered as before. Concretely, let $s_1 \geq \cdots \geq s_k$ be the size of each cluster. If $s_k < \frac{bn}{k}$, we show there is a $\mu > 0$ and $h \in [k]$ such that,

$$s_1 \geq \cdots \geq s_h \geq \mu \cdot n > (\mu - b \cdot k^{-2}) \cdot n \geq s_{h+1} \cdots \geq s_k.$$

Notice that for every $i \le h$ and $v \in V_i$, the expectation of the degree of v in the random graph G is

$$\mathop{\mathbf{E}}_{G}\left[|\{u : (u,v) \in E(G)\}|\right] = \left(\frac{1}{2}+\frac{\delta}{2}\right)|V_i| + \left(\frac{1}{2}-\frac{\delta}{2}\right)(n-|V_i|) \ge \left(\frac{1}{2}-\frac{\delta}{2}\right)n + \delta\mu n$$

On the other hand, for each $i' > h$ and $v' \in V_{i'}$, the expectation of degree of v is

$$\mathop{\mathbf{E}}_{G}\left[|\{u : (u,v') \in E(G)\}|\right] = \left(\frac{1}{2}+\frac{\delta}{2}\right)|V_i| + \left(\frac{1}{2}-\frac{\delta}{2}\right)(n-|V_i|) \le \left(\frac{1}{2}-\frac{\delta}{2}\right)n + \delta\mu n - \delta\cdot b\cdot k^2 n$$

Therefore, there is a $\delta \cdot b \cdot k^2 n$ gap between large clusters and small clusters (in expectation). It is easy to show that the gap also exists with high probability by applying the standard concentration bound.

Now if we sample a subset T of size at least $\Omega(\frac{k^4 \log n}{\delta^2})$, then we can guarantee that with high probability, for all vertices in large clusters V_i ($i \le h$), they have degree larger than some threshold d_h in H_T, while for all vertices in small clusters V_i ($i > h$), they have degree smaller than d_h in H_T. In this way, we can filter out all vertices in T that belong to small clusters and let the remaining vertex set be T' and the corresponding subgraph be $H_{T'}$. Then we can run Vu's algorithm on $H_{T'}$ to identify all the sub-clusters in T' that corresponding to large clusters in G. However, there is one subtle issue in the above approach, that is, we do not know the index h that corresponds to the size-gap. To resolve this issue, we simply try all possible candidates h: for each $h \in [k]$, we pretend that h is the index corresponding to the size-gap of the clusters. Then we use h to obtain a filtered subgraph $H_{T'}$ and invoke Vu's algorithm on $H_{T'}$ to find h sets $X_1, \ldots, X_h$. Now we give a simple algorithm to test if h is the 'right' index, by testing if all sets X_i are *biased* towards some true cluster C or not, i.e., if the the majority of X_i belong to C. We can show that if for an index h, all the sets $X_1, \ldots, X_h$ pass the bias testing, then we can still use each X_i to grow the cluster. Finally, if h is the index that corresponds to size-gap, then it will pass the test with high probability by the previous argument, which ensures that we can always find some clusters in this way. We give details in Section 4.

1.3 Towards optimal dependency on the number of clusters

As mentioned before, our algorithm (in Theorem 3) for clustering nearly-balanced instances makes $O(k \cdot n\log n/\delta^2 + k^4\log^2 n/\delta^4)$ queries, which is in comparison to the known lower bound $\Omega(k\cdot n/\delta^2)$ [Mazumdar and Saha, 2017a]. There exists evidence indicating that our query complexity might be almost optimal, in particular, improving the factor k^4 in the second term of the query complexity seems difficult when k is large.

Several papers [Decelle et al., 2011, Chen et al., 2014b] suggested that, using non-rigorous but deep arguments from statistical physics, efficiently recovering the clusters in SBM(N, p, q, δ) is impossible if $\frac{p-q}{\sqrt{p}} = o(\frac{\sqrt{N}}{s})$, where s is the size of minimum cluster. Translating it to our case with $N = n$, $p = \frac{1}{2}+\frac{\delta}{2}$, $q = \frac{1}{2}-\frac{\delta}{2}$ and $s = \Omega(\frac{n}{k})$, it suggests that even if we query the whole graph (i.e., with $\Theta(n^2)$ queries), it is impossible to recover the clusters if $k = \omega(\delta\sqrt{n})$. On the other hand, suppose that there exists a polynomial time algorithm $\mathcal{B}$ that solves our problem with query complexity $O(kn/\delta^2 + k^{4-\varepsilon}/\delta^4)$ for any constant $\varepsilon > 0$, then it can recover the clusters in the corresponding SBM model by querying $o(n^2)$ pairs, for $k = \delta n^{\frac{1}{2}+\frac{\varepsilon}{10}} = \omega(\delta\sqrt{n})$, which seems impossible by the aforementioned evidence.

It will be very interesting to formally prove that the query complexity $O(k\cdot n\log n/\delta^2 + k^4\log^2 n/\delta^4)$ of the algorithm in Theorem 3 is almost optimal (up to a $\log^2 n$ factor) for any *polynomial time* algorithm, by assuming some standard hardness assumptions (e.g. finding a random clique is hard) in complexity theory. In fact, Mazumdar and Saha [Mazumdar and Saha, 2017a] also pointed it is impossible to push their query-optimal algorithm to be efficient unless there is an efficient algorithm finding hidden clique in random graphs .

2 Two Subroutines

We now introduce two subroutines, which will be used in our clustering algorithms later.

2.1 An algorithm for nearly balanced clustering in stochastic block model

For convenience of notation, we introduce the following. Fix any k clusters $V_1, \dots, V_k$ and a bias parameter $\delta \in [0, 1)$. The distribution $\mathcal{D}(V_1, \dots, V_k, \delta)$ samples a random graph as follows: for any two vertices u and v, we add an edge between them with probability $(1/2 + \delta/2)$ if u and v come from the same cluster V_i, and add an edge between them with probability $(1/2 - \delta/2)$ otherwise. The goal of the clustering algorithm is to recover the clusters $V_1, \dots, V_k$ though a random graph $G \sim \mathcal{D}(V_1, \dots, V_k, \delta)$.

We first note that the following result was implicitly shown in Vu [2018].

Theorem 4 (Vu [2018])**.** *Let $\delta \in [0, \frac{1}{2}]$ and $G \sim \mathcal{D}(V_1, \dots, V_k, \delta)$. Let $n = |V_1| + \cdots + |V_k|$. Suppose that the partition $V_1, \dots, V_k$ is b-balanced for some $b \in (0, 1]$. Then there exists an algorithm, denoted by* BALPARTITION*(G, k, δ, b), that recovers all the clusters $V_1, \dots, V_k$ of G in polynomial time with probability at least $1 - n^{-8}$, if the following condition holds,*

$$n \geq c_0 \frac{k^2}{b^2 \delta^2} \log n,$$

where $c_0 > 1000$ is some universal constant.

This theorem is slightly different from the original version of Vu [2018], and we present an explanation in Appendix B.1.

2.2 Growing a cluster from a biased set

All our algorithms will make use of a subroutine (Algorithm 1) for classifying vertices in V with the help of a *biased* set B, of which the majority belong to the same cluster. More formally, we give the following definition.

Definition 5. *Let $\eta \in [0, \frac{1}{2}]$. Let C be a true cluster, i.e., $C = V_i$ for some $i \in [k]$. A set of vertices B is called (η, C)-biased if $|B \cap C| \geq (1/2 + \eta) \cdot |B|$.*

Note that if $\eta = \frac{1}{2}$, then all the vertices in set B are contained in C, i.e., $B \subseteq C$. In this case, we all B a *sub-cluster* of C. We now describe this subroutine and state its performance guarantee.

Algorithm 1 BELONGTOCLUSTER(v, B): test if v belongs to a cluster C, given a (η, C)-biased set B

```
Query all pairs v, w for w ∈ B and let cnt be the number of + answers
if cnt ≥ |B|/2 then
    return Yes
else
    return No
end if
```

Lemma 6. *Let B be a set that is (η, C)-biased and have size at least $\frac{16 \log n}{\eta^2 \delta^2}$. Then with probability at least $1 - n^{-7}$,*

- *for all vertices* $v \in C$, BELONGTOCLUSTER(v, B) *returns* **Yes**;
- *for all vertices* $v \in V \setminus C$, BELONGTOCLUSTER(v, B) *returns* **No**.

Note that the above lemma says that by invoking BELONGTOCLUSTER(v, B) for any $v \in V$, we can identify all the cluster members in C with high probability.

Proof of Lemma 6. Let v be an arbitrary vertex. Let B_v denote the subset of vertices of B that belong to the same cluster as v. Query all the edges between v and B. Then the expected number of '$+$' neighbors of v is

$$\left(\frac{1}{2}+\frac{\delta}{2}\right)|B_v| + \left(\frac{1}{2}-\frac{\delta}{2}\right)|B \setminus B_v| = \left(\frac{1}{2}-\frac{\delta}{2}\right)|B| + \delta|B_v|$$

Let $\lambda = \frac{\eta\delta|B|}{2}$. Note that $\lambda^2/|B| \geq 4\log n$ as $|B| \geq \frac{16\log n}{\eta^2\delta^2}$. Recall that B is (η, C)-biased for some constant η and cluster C. We consider two cases.

- If $v \in C$, then $|B_v| \geq (\frac{1}{2}+\eta)|B|$ and the expected number of '$+$' neighbors of v is at least

  $$\left(\frac{1}{2}-\frac{\delta}{2}\right)|B| + \left(\frac{1}{2}+\eta\right)\delta|B| = \left(\frac{1}{2}+\eta\delta\right)|B|$$

  By Chernoff–Hoeffding bound (see Theorem 13), with probability at least $1 - e^{-2\lambda^2/|B|} \geq 1 - n^{-8}$, the number of '$+$' neighbors of v is at least

  $$\left(\frac{1}{2}+\eta\delta\right)|B| - \lambda = \left(\frac{1}{2}+\frac{1}{2}\eta\delta\right)|B| > \frac{1}{2}|B| \tag{1}$$

- if $v \in C'$ for some cluster $C' \neq C$, then $|B_v| \leq \left(\frac{1}{2}-\eta\right)|B|$, the expected number of '$+$' neighbors of v is at most

  $$\left(\frac{1}{2}-\delta\right)|B| + \left(\frac{1}{2}-\eta\right)\delta|B| = \left(\frac{1}{2}-\eta\delta\right)|B|$$

  By Chernoff–Hoeffding bound, with probability at least $1 - e^{-2\lambda^2/|B|} \geq 1 - n^{-8}$, the number of '$+$' neighbors of v is at most

  $$\left(\frac{1}{2}-\eta\delta\right)|B| + \lambda = \left(\frac{1}{2}-\frac{1}{2}\delta\eta\right)|B| < \frac{1}{2}|B|$$

Therefore, with probability at least $1 - n^{-7}$, for each vertex $v \in V$, it holds that

- if $v \in C$, then the number of $+$ neighbors is at least $\frac{1}{2}|B|$, and BELONGTOCLUSTER(v, B) returns **Yes**; and
- if $v \notin C$, then the number of $+$ neighbors is less than $\frac{1}{2}|B|$, and BELONGTOCLUSTER(v, B) returns **No**.

□

3 Clustering Nearly-Balanced Instances

In this section, we give our algorithm for clustering b-balanced instances, for any $b \in (0, 1]$. It simply first invokes the following Algorithm 2 and then Algorithm 3. It is built on the two subroutines BALPARTITION and BELONGTOCLUSTER introduced in Section 2.

Algorithm 2 BALANCEDCLUSTERING(V, k, δ, b): clustering for a b-balanced instance

1: Let $n = |V|$, $b' = b/2$ and c_0 be the constant from Theorem 4
2: Randomly sample a subset $T \subset V$ of size $|T| = \frac{400c_0k^2 \log n}{b^2\delta^2}$
3: Query all pairs $u, v \in T$ and let H_T be graph on vertex set T with only positive edges from the query answers
4: Apply BALPARTITION(H_T, k, δ, b') to obtain clusters $X_1, \ldots, X_k$

Algorithm 3 GLOBALGROW$(V, X_1, \ldots, X_k)$: from sub-clusters to clusters

1: Let $U = V$ and $n = |V|$
2: For each $1 \le i \le k$, find an arbitrary subset $X_i' \subseteq X_i$ of size $\frac{1600 \log n}{\delta^2}$
3: **for** each $i \in [k]$ **do**
4: let $C_i := \{v \in U :$ BELONGTOCLUSTER(v, X_i') returns $\mathbf{Yes}\}$
5: update $U \leftarrow U \setminus C_i$
6: **end for**
7: **return** $C_1, \cdots, C_k$

Now we provide the analysis of this algorithm, i.e., prove Theorem 3. In the following, we let T denote the sample set from BALANCEDCLUSTERING(V, k, δ, b). For each $i \in [k]$, let $T_i = T \cap V_i$ be the sub-clusters. We first show that, with high probability, the clusters $T_1, \ldots, T_k$ are balanced.

Lemma 7. *Let $V_1, \ldots, V_k$ be a family of b-balanced clusters. Then with probability at least $1 - n^{-7}$, $T_1, \ldots, T_k$ is b'-balanced.*

Proof. Since $V_1, \ldots, V_k$ is a family of b-balanced clusters, we have that $\mathbf{E}[|V_i \cap T|] \ge b \cdot |T|/k$. Notice that T is a uniform random subset. By the Chernoff bound, for each i, with probability at least $1 - n^{-8}$, $|T_i| \ge b' \cdot |T|/k$. The claim then follows by the union bound. □

Now we may assume that $(T_1, \ldots, T_k)$ is b'-balanced. Since the size of T is large, i.e., $|T| = \frac{400c_0k^2 \log n}{b^2\delta^2} = \frac{100 \cdot c_0 \cdot k^2 \log n}{b'^2\delta^2}$, we are able to recover the clusters in T by Theorem 4.

Lemma 8. *Suppose that the partition $T_1, \ldots, T_k$ of the sampled set T is b'-balanced. Let $X_1, \ldots, X_k$ be the output sets of* BALANCEDCLUSTERING*(V, k, δ, b). Then*

$$\Pr[X_1, \ldots, X_k \text{ is not a correct clustering of } H_T] \le |T|^{-8}$$

Proof. Note that by our choice of $|T|$ and that $b' = \frac{b}{2}$, we have $|T| \ge c_0 \frac{k^2}{b'^2\delta^2} \log n$. Then the correctness of Lemma 8 simply follows by Theorem 4. □

Now we are ready to prove Theorem 3.

Proof of Theorem 3. By Lemma 8, the output $X_1, \ldots, X_k$ is a correct clustering of H_T, with probability $1 - o_n(1)$. Conditioned on this, we know that each X_i is $(\frac{1}{2}, C)$-biased for some cluster C. This also implies that each $X'_i \subseteq X_i$ is $(\frac{1}{2}, C)$-biased. Thus, by invoking BELONGTOCLUSTER(v, X'_i) for all $v \in V$ and $i \leq k$ and by Lemma 6 with $\eta = 0.1 < \frac{1}{2}$, we can guarantee that the output $C_1, \ldots, C_k$ of GLOBAL-GROW$(V, X_1, \ldots, X_k)$ is a correct clustering with probability $1 - \Theta(|T|^{-8}) = 1 - o_n(1)$.

Note that we query all the pairs $u, v \in T$, which corresponds to $|T|^2$ queries. Note further that there are at most k clusters, each of which grows from a sub-cluster of size $\Theta(\frac{\log n}{\delta^2})$. In total, the query complexity of Algorithm 2 and 3 is upper bounded by $O(|T|^2 + k\frac{\log n}{\delta^2} \cdot n) = O(k^4 \cdot \log^2 n/(b^4\delta^4) + nk \cdot \log n/\delta^2)$. Since the running time of BALPARTITION is polynomial in $|T|, k, \delta, b$ and the running time for growing each of the clusters is linear in n, the total running time of our algorithm is polynomial (in n, k, δ, b). □

4 Clustering the General Instances

In the section, we give our algorithm for the general instances.

4.1 Existence of size-gap in unbalanced instances

We first focus on the unbalanced case, that is, the underlying clustering is not b-balanced, i.e., the size of the minimum cluster is less than $\frac{bn}{k}$. Let $V_1, \ldots, V_k$ be a family of clusters, and let $s_1, \ldots, s_k$ be the size of each cluster respectively. Without loss of generality, we assume that $s_1 \geq \cdots \geq s_k$. A useful observation is the following size-gap lemma. Roughly speaking, for any unbalanced clusters, there a threshold which separates large and small clusters.

Lemma 9 (size-gap)**.** *Let $b \in [0, \frac{1}{2}]$. If $s_k < \frac{bn}{k}$, then there exists $h < k$ such that*

- *$s_h \geq \frac{n}{k} - \frac{h \cdot b \cdot n}{k^2}$, and $s_{h+1} < \frac{n}{k} - \frac{(h+1) \cdot b \cdot n}{k^2}$.*

Hence the gap between s_h and s_{h+1} is at least $\frac{bn}{k^2}$.

Proof. Note that by averaging argument, it holds that $s_1 \geq \frac{n - s_k}{k-1} \geq \frac{(1-b/k)n}{k-1} > \frac{(1-b/k)n}{k} = \frac{n}{k} - \frac{bn}{k^2}$. This implies that the subset $I \subseteq [k]$ of indices i with $s_i \geq \frac{n}{k} - \frac{i \cdot bn}{k^2}$ is not empty. Let h be the largest i in the set I. Furthermore, since $s_k < \frac{bn}{k} \leq \frac{(1-b)n}{k} = \frac{n}{k} - \frac{k \cdot bn}{k^2}$ for any $b \leq \frac{1}{2}$, it must hold that $k \notin I$ and thus $h \leq k - 1$. The statement of the lemma then follows from the choice of h. □

4.2 Recovering sub-clusters from the sampled subgraph with known gap

From Lemma 9, we know that in the unbalanced case, there is a size-gap between two clusters V_h and V_{h+1}, for some index $h \leq k - 1$. In the following, we first present an algorithm under the assumption that the index h is known. Later, we show how to use this algorithm to deal with the general case.

Algorithm 4 GAPCLUSTERING(V, h, δ, b): clustering with known size-gap

1: Let $n = |V|$ and sample a set $T \subset U$ of size $t = \frac{8c_0k^4 \log n}{b^2 \cdot \delta^2}$
2: Query all pairs $u, v \in T$
3: Let $H_T = (T, E_T)$ be graph on vertex set T with only positive edges from the query answers
4: Remove all vertices in H_T with degree less than $d_h := \frac{t}{2} - \left(\frac{1}{2} - \frac{1}{k} + \frac{(h+1/2)b}{k^2}\right)\delta t$
5: Let T' be the set of remaining vertices and let the resulting graph be $H_{T'}$
6: Apply BALPARTITION$(H_{T'}, k, \delta, b'' := \frac{h}{2k})$ to find clusters $X_1, \ldots, X_h$

The crucial idea of the above algorithm is that we are able to show the Step 4 of Algorithm 4 removes all vertices sampled from small clusters in T. Hence the remaining graph T' becomes a nearly-balanced clustering instance, in which the sub-clusters correspond to large clusters $V_1, \ldots, V_h$. We have the following lemma regarding this algorithm.

Lemma 10. *Let $b \in [0, \frac{1}{2}]$. Suppose that $s_h \geq \frac{n}{k} - \frac{h \cdot b \cdot n}{k^2}$, and $s_{h+1} < \frac{n}{k} - \frac{(h+1) \cdot b \cdot n}{k^2}$. Then with probability $1 - O(k^{-24} \log^{-8} n)$, the algorithm* GAPCLUSTERING*(V, h, δ, b) successfully recover all the sub-clusters from the sampled set T, which correspond to true clusters $V_1, \ldots, V_h$.*

Proof. Let $T_i = V_i \cap T$, where T is the sample set with t vertices from the algorithm. Let $\lambda_1 = \frac{bt}{4k^2}$. Note that $\lambda_1^2/t \geq 4 \log n$ by our setting $t = \frac{8c_0k^4 \log n}{b^2\delta^2}$.

We first note that (over the randomness of sampling the vertex set T)

- for any $i \leq h$, it holds that $\mathbf{E}[|T_i|] \geq (\frac{1}{k} - \frac{hb}{k^2})t$. Thus, by Chernoff–Hoeffding bound (Theorem 13), with probability at least $1 - e^{-2\lambda_1^2/t} \geq 1 - n^{-8}$,

$$|T_i| \geq \left(\frac{1}{k} - \frac{hb}{k^2}\right) t - \lambda_1 = \left(\frac{1}{k} - \frac{(h + 1/4)b}{k^2}\right) t \tag{2}$$

- for any $i > h$, it holds that $\mathbf{E}[|T_i|] < (\frac{1}{k} - \frac{(h+1)b}{k^2})t = (\frac{1}{k} - \frac{hb}{k^2})t - \frac{bt}{k^2}$. Thus, with probability at least $1 - e^{-2\lambda_1^2/t} \geq 1 - n^{-8}$,

$$|T_i| < \left(\frac{1}{k} - \frac{hb}{k^2}\right) t - \frac{bt}{k^2} + \lambda_1 \leq \left(\frac{1}{k} - \frac{(h + 3/4)b}{k^2}\right) t \tag{3}$$

In the following, we assume the inequalities (2) and (3) hold for all $i \leq k$, which occur with probability at least $1 - n^{-7}$ by the union bound.

Now we analyze the vertex degrees of vertices in the queried graph H_T. We first note that for any $v \in T_i$, its expected degree is

$$\left(\frac{1}{2} + \frac{\delta}{2}\right)|T_i| + \left(\frac{1}{2} - \frac{\delta}{2}\right)|T \setminus T_i| = \left(\frac{1}{2} - \frac{\delta}{2}\right)|T| + \delta|T_i|$$

Let $\lambda_2 = \frac{bt\delta}{4k^2}$. Note that $\lambda_1^2/t \geq 4 \log n$ by our setting. Now we have that

- for any $i \leq h$ and vertex $v \in T_i$, then its expected degree is at least

$$\left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta b_h \geq \left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta \cdot \left(\frac{1}{k} - \frac{(h + 1/4)b}{k^2}\right) t.$$

Thus, over the randomness of querying the oracle regarding vertices in T, with probability at least $1 - e^{-2\lambda_2^2/t} \geq 1 - n^{-8}$, the degree of v is at least

$$\left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta \cdot \left(\frac{1}{k} - \frac{(h+1/4)b}{k^2}\right) t - \lambda_2 = \left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta \cdot \left(\frac{1}{k} - \frac{(h+1/2)b}{k^2}\right) t$$

- for any $i > h$ and vertex $v \in T_i$, its expected degree is less than

$$\left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta |T_i| \leq \left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta \cdot \left(\frac{1}{k} - \frac{(h+3/4)b}{k^2}\right) t$$

Thus, with probability at least $1 - e^{-2\lambda_2^2/t} \geq 1 - n^{-8}$, the degree of v is less than

$$\left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta \cdot \left(\frac{1}{k} - \frac{(h+3/4)b}{k^2}\right) t + \lambda_2 = \left(\frac{1}{2} - \frac{\delta}{2}\right) t + \delta \cdot \left(\frac{1}{k} - \frac{(h+1/2)b}{k^2}\right) t$$

Let $d_h := (\frac{1}{2} - \frac{\delta}{2})t + \delta \cdot \left(\frac{1}{k} - \frac{(h+1/2)b}{k^2}\right) t = \frac{t}{2} - \left(\frac{1}{2} - \frac{1}{k} + \frac{(h+1/2)b}{k^2}\right) \delta t$. That is, with probability at least $1 - n^{-7}$, all vertices in $T_1, \ldots, T_h$ have degree at least d_h, and all vertices in $T_{h+1}, \ldots, T_k$ have degree less than d_h. Then by the description of the algorithm, $T' = \cup_{i \leq h} T_i$.

Now we note that $H_{T'} \sim \mathcal{D}(T_1, \ldots, T_h, \delta)$, and that the number of clusters in $H_{T'}$ is h. Now we apply BALPARTITION($H_{T'}, h, \delta, b''$) on $H_{T'}$. Recall that we have chosen $t = \frac{8c_0 k^4 \log n}{b^2 \delta^2}$. Note that we only need to consider the case that $t \leq n$ (as otherwise, we can simply query the whole graph). Now we note that

$$t \geq |T'| \geq \sum_{i=1}^{h} |T_i| \geq h \cdot \left(\frac{1}{k} - \frac{(h+1/4)b}{k^2}\right) t \geq \frac{ht}{2k}$$

Furthermore, we know for each $i \leq h$,

$$|T_i| \geq \left(\frac{1}{k} - \frac{(h+1/4)b}{k^2}\right) t \geq \frac{t}{2k} \geq \frac{|T'|}{2k} = \frac{h}{2k} \cdot \frac{|T'|}{h}.$$

Thus, if we set $b'' = \frac{h}{2k}$, then the partition $T_1, \cdots, T_h$ is b''-balanced. Note that $|T'| \geq \frac{ht}{2k} \geq \frac{4c_0 k^3 \log n}{b^2 \delta^2}$. Thus,

$$\log |T'| \leq \log t \leq \log n$$

$$\frac{|T'|}{\log |T'|} \geq \frac{4c_0 k^2}{h^2} \cdot \frac{h^2}{\delta^2} \cdot \frac{\log n}{\log t} \geq \frac{c_0}{b''^2} \cdot \frac{h^2}{\delta^2}$$

Thus by Theorem 4, the algorithm BALPARTITION($H_{T'}, h, \delta, b''$) successfully recover all the clusters $T_1, \ldots, T_h$ with probability at least $1 - |T'|^{-8} \geq 1 - O((b\delta)^{16} k^{-24} \log^{-8} n)$.

□

4.3 Finding a good index h

In the previous section, we presented an algorithm for finding clusters assuming that the index h that corresponds to the size-gap is known, and we have shown that the algorithm GAPCLUSTERING(V, h, δ, b) outputs h sub-clusters from the sampled set T. However, in the general case, we do not know this index h. To handle

this issue, we enumerate all possible candidates h for $1 \le h \le k$, and use a subroutine to test if the current candidate h is 'right' or not, which in turn makes use of a procedure for testing the bias of a given set.

We first describe the algorithm for testing the bias of a set. Its performance is guaranteed in Lemma 11.

Algorithm 5 TESTBIAS(n, B, η): test if a set B is (η, C)-biased for some cluster C

1: **for** $i = 1, \cdots, \frac{16k \cdot \log n}{b}$ **do**
2: Randomly sample a vertex v_i and query all the pairs v_i, u for $u \in B$
3: **if** the number of '+' neigbhors of v_i in B is at least $(\frac{1}{2} + \frac{1}{2}\eta\delta)|B|$ **then**
4: **return Yes**
5: **end if**
6: **end for**
7: **return No**

Lemma 11. *Let B be a vertex set of size at least $\frac{64 \log n}{\eta^2 \delta^2}$. There exists one algorithm* TESTBIAS*(n, B, η) that with probability at least $1 - n^{-7}$,*

- *accepts B, if B is (η, C)-biased for some cluster C of size at least $\frac{bn}{k}$, i.e., $|B \cap C| \ge (1/2 + \eta) \cdot |B|$*
- *rejects B, if B is not $(\frac{\eta}{4}, C)$-biased for any C, i.e., for any C, $|B \cap C| < (1/2 + \frac{\eta}{4}) \cdot |B|$.*

Proof. We first consider the case that B is (η, C)-biased for some cluster C of size at least $\frac{bn}{k}$. Note that with probability at least $1 - n^{-8}$, one of the sampled $\frac{16k \log n}{b}$ vertices will belong to C, as $|C| \ge \frac{bn}{k}$.

Furthermore, by the same calculations as the inequality (1) in the proof of Lemma 6, we know that with high probability, the $+$ neighbors of v is at least $(\frac{1}{2} + \frac{1}{2}\eta\delta)|B|$, then TESTBIAS$(n, B, \eta)$ will return **Yes**.

Now suppose that B is not $(\frac{\eta}{4}, C)$-biased for any C. For any vertex $v \in V$, let B_v be the set of vertices in B in the same cluster as v. Then $|B_v| < (\frac{1}{2} + \frac{\eta}{4})|B|$. The expected number of '+' neighbors of v is

$$\left(\frac{1}{2} - \frac{\delta}{2}\right)|B| + \delta|B_v| \le \left(\frac{1}{2} - \frac{\delta}{2}\right)|B| + \delta\left(\frac{1}{2} + \frac{\eta}{4}\right)|B| = \left(\frac{1}{2} + \frac{\eta\delta}{4}\right)|B|$$

Let $\lambda = \frac{\eta\delta|B|}{4}$. Note that $\lambda^2/|B| \ge 4 \log n$ as $|B| \ge \frac{64 \log n}{\eta^2\delta^2}$. By Chernoff–Hoeffding bound, with probability at least $1 - e^{-2t^2/|B|} \ge 1 - n^{-8}$, the number of $+$ neighbors of v is less than $(\frac{1}{2} + \frac{\eta\delta}{4})|B| + \lambda = (\frac{1}{2} + \frac{\eta\delta}{2})|B|$. In this case, the TESTBIAS(n, B, η) will return **No**. □

Now we describe our idea for finding a good index h and the corresponding sub-clusters. For each $h \in [k]$, we first "pretend" that the gap is h, and invoke GAPCLUSTERING(V, h, δ, b) to find h different sets $X_1, \cdots, X_h$ (or invoke BALANCEDCLUSTERING(V, h, δ, b) if $h = k$). Then we select sufficiently large subsets $X_i' \subset X_i$, $1 \le i \le h$, and test if all of the sets X_i' are sub-clusters by invoking a subroutine TESTBIAS(n, X_i', η). If so, we say the corresponding index h is accepted, and the algorithm outputs the sets $X_1', \ldots, X_h'$.

Algorithm 6 ENUMERATEINDEX(V, k, δ, b, η): find a good index h and the corresponding sub-clusters

Let $n = |V|$
for $h = k, \ldots, 1$ **do**
  if $h == k$ **then**
    Invoke BALANCEDCLUSTERING(V, h, δ, b) to find h clusters $X_1, \ldots, X_h$
  else
    Invoke GAPCLUSTERING(V, h, δ, b) to find h clusters $X_1, \ldots, X_h$
  end if
  For each $i \leq h$, let X'_j be an arbitrary subset of X_j of size $\frac{256 \log n}{\eta^2 \delta^2}$
  if for all $i \leq h$, TESTBIAS(n, X'_i, η) returns **Yes then**
    return $X'_1, \ldots, X'_h$
  end if
end for
return Fail.

We have the following lemma regarding the performance guarantee of the above algorithm.

Lemma 12. *Let $\eta^2/b \geq 64/c_0$, where c_0 is the constant from Theorem 4. It holds that with probability at least $1 - n^{-6}$,*

- *there exists an index $h \in [k]$ such that* ENUMERATEINDEX*(V, k, δ, b, η) will output h sets $X'_1, \ldots, X'_h$;*
- *if $X'_1, \ldots, X'_h$ are the sets output by* ENUMERATEINDEX*(V, k, δ, b, η), then each of them is $(\eta/4, C)$-biased for some cluster C.*

Proof. If the instance is b-balanced, then we let $h = k$, and by Lemma 8, BALANCEDCLUSTERING(V, h, δ, b) outputs all the sub-clusters $X_1, \ldots, X_h$ from the sample set T. If the instance is not b-balanced, then by Lemma 10, there exists an index $h \in [1, k-1]$ that corresponds to size-gap, and thus all the output sets X_i by GAPCLUSTERING(V, h, δ, b) are sub-clusters. In both cases, we know that X_i's are $(\frac{1}{2}, C)$-biased for some cluster C. Now by the previous argument, we can guarantee that each of the set X_i has size at least $\frac{200 c_0 k \log n}{b\delta^2}$ (in case that $h = k$) or $\frac{4 c_0 k^3 \log n}{b^2 \delta^2}$ (in case that $h \leq k-1$), and thus larger than $\frac{256 \log n}{\eta^2 \delta^2}$, as $\eta^2/b \geq 64/c_0$ by assumption. Therefore, we can find subsets X'_i, $1 \leq i \leq h$ of size $\frac{256 \log n}{\eta^2}$ that are $(\frac{1}{2}, C)$-biased for some cluster C. Thus, by Lemma 11, for all $i \leq h$, TESTBIAS(n, X'_i, η) will be accepted with high probability.

Now we prove the second item of the lemma. Let h be an index such that $1 \leq h \leq k$. Let $X'_1, \ldots, X'_h$ be the sets corresponding to Step 8 of the algorithm ENUMERATEINDEX. Let $\mathcal{E}_h$ denote the event that there exists one of the sets X'_i, $1 \leq i \leq h$ is not $(\frac{\eta}{4}, C)$-biased for any C. For any h such that $\mathcal{E}_h$ holds, we know that with probability at least $1 - n^{-7}$, one of tests TESTBIAS(n, X'_i, η) will return **No** and thus h will not be accepted. Therefore, we can assume that for any $h \leq k$ such that $\mathcal{E}_h$ holds, h will be rejected, which happens with probability at least $1 - n^{-6}$. Furthermore, under this assumption, we have that if h is accepted, then $\mathcal{E}_h$ does not hold, i.e., all the sets X'_i, $1 \leq i \leq h$ are $(\frac{\eta}{4}, C)$-biased for some cluster C. □

4.4 The final algorithm

Our algorithm is outlined as follows.

- Initialize $U = V$ and suppose the number of clusters in the current graph $G[U]$ is k_c, which equals k at very beginning. Repeat the following until U has small enough size or $k_c \leq 1$.

- Use ENUMERATEINDEX$(U, k_c, \delta, b, \eta)$ to find h sets $X'_1, \ldots, X'_h$, for some $h \le k_c$.
- Grow the found sets $X'_1, \ldots, X'_h$ to find the clusters $C_1, \ldots, C_h$.
- Update k_c to be $k_c - h$, and remove all the clustered vertices from U.

• Output all the found clusters C_i's.

The psuedocode of the algorithm is as follows.

Algorithm 7 NOISYCLUSTERING(V, k, δ): the final clustering algorithm

Let $U = V$; let $k_c = k$ be the number of clusters in current graph; let $j = 0$ be the number of clusters found so far; let c_0 be the universal constant from Theorem 4; let $b = \eta = 0.1$
while $|U| \ge \frac{40000 c_0 k^4 \log n}{\delta^2}$ and $k_c \ge 2$ **do**
  Invoke ENUMERATEINDEX$(U, k_c, \delta, b, \eta)$ and let $X'_1, \ldots, X'_h$ denote the output h sets.
  for each $i \in [h]$ **do**
    $C_{j+i} \leftarrow \{v \in U :$ BELONGTOCLUSTER(v, X'_i) returns **Yes**$\}$
    $U \leftarrow U \setminus C_{j+i}$
  end for
  $j \leftarrow j + h$
  $k_c \leftarrow k_c - h$
end while
return all the clusters C_i's

Proof of Theorem 1. Since we have set $b = \eta = 0.1$, it holds that $\eta^2/b \ge 64/c_0$ as $c_0 \ge 1000$ by Theorem 4. By Lemma 12, we know Algorithm 7 will output $X'_1, \ldots, X'_h$ for some $h \le k_c$, and each of these sets is $(\frac{\eta}{4}, C)$-biased for some cluster C. Then by Lemma 6, we can grow each X'_i to get the true cluster C. Note that at least one cluster will be found in each iteration, and the error probability in each iteration is at most $o_n(1)/k$ (by Lemma 8 and 10). The final algorithm thus succeeds with probability $1 - o_n(1)$ as there are at most k iterations. The correctness of the algorithm then follows from the fact that the algorithm stops when all the k clusters have been identified or the size of the remaining graph becomes smaller than $\frac{40000 c_0 k^4 \log n}{\delta^2}$.

Now we bound the query complexity of the algorithm. Note that there are at most k iterations. In each iteration, we invoke ENUMERATEINDEX to try all k possible values of h. For each h, we will sample at most $t = \frac{400 c_0 k^4 \log n}{b^2 \delta^2} = \frac{40000 c_0 k^4 \log n}{\delta^2}$ vertices and query the induced subgraph by making t^2 queries for finding biased sets. To test the bias of each candidate set X'_i (i.e., invoke TESTBIAS(n, X'_i, η)), we only need to sample $\Theta(\frac{k \log n}{b})$ vertices and make $O(\frac{k \log n}{b} \cdot \frac{\log n}{\eta^2 \delta^2})$ queries. For the accepted index h, i.e., ENUMERATEINDEX outputs h sets $X_1, \ldots, X_H$, we will make use of the subsets $X'_1, \ldots, X'_h$ to grow the clusters, and growing any set X'_i to the true cluster requires at most $\frac{256 \log n}{\eta^2 \delta^2} n$ queries. Finally, we note that there can be at most k subsets X'_i throughout the whole procedure that we will use to grow the clusters. Thus, the total query complexity is $O(k^2 t^2 + kn \log n/\delta^2) = O(\frac{k^{10} \log^2 n}{\delta^4} + \frac{nk \log n}{\delta^2})$.

Regarding the running time, we let $T(t, k, \delta) = \text{poly}(t, k, 1/\delta)$ denote the running time of BALPARTITION (in Theorem 4) on a set of size t. The running time for TESTBIAS(n, T_i, η) is proportional to the size T_i and the running time of using BELONGTOCLUSTER to identify each cluster is at most tn. Thus, the total running time is $O(k^2 T(t, k, \delta) + kn \log n/\delta^2) = O((\frac{k \log n}{\delta})^C + \frac{nk \log n}{\delta^2})$, for some constant $C > 0$. □

References

Emmanuel Abbe. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research*, 18(1):6446–6531, 2017.

Nikhil Bansal, Avrim Blum, and Shuchi Chawla. Correlation clustering. *Machine learning*, 56(1-3):89–113, 2004.

Michael J Brzozowski, Tad Hogg, and Gabor Szabo. Friends and foes: ideological social networking. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 817–820, 2008.

Moira Burke and Robert Kraut. Mopping up: modeling wikipedia promotion decisions. In *Proceedings of the 2008 ACM conference on Computer supported cooperative work*, pages 27–36, 2008.

Nicolo Cesa-Bianchi, Claudio Gentile, Fabio Vitale, and Giovanni Zappella. A correlation clustering approach to link classification in signed networks. In *Conference on Learning Theory*, pages 34–1. JMLR Workshop and Conference Proceedings, 2012.

Yudong Chen, Ali Jalali, Sujay Sanghavi, and Huan Xu. Clustering partially observed graphs via convex optimization. *The Journal of Machine Learning Research*, 15(1):2213–2238, 2014a.

Yudong Chen, Sujay Sanghavi, and Huan Xu. Improved graph clustering. *IEEE Transactions on Information Theory*, 60(10):6440–6455, 2014b.

Nilesh Dalvi, Anirban Dasgupta, Ravi Kumar, and Vibhor Rastogi. Aggregating crowdsourced binary ratings. In *Proceedings of the 22nd international conference on World Wide Web*, pages 285–294, 2013.

Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6): 066106, 2011.

Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

Ivan P Fellegi and Alan B Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64(328):1183–1210, 1969.

Chaitanya Gokhale, Sanjib Das, AnHai Doan, Jeffrey F Naughton, Narasimhan Rampalli, Jude Shavlik, and Xiaojin Zhu. Corleone: hands-off crowdsourcing for entity matching. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 601–612, 2014.

Sally A Goldman, Michael J Kearns, and Robert E Schapire. Exact identification of circuits using fixed points of amplification functions. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*, pages 193–202. IEEE, 1990.

David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In *Advances in neural information processing systems*, pages 1953–1961, 2011.

Kasper Green Larsen, Michael Mitzenmacher, and Charalampos Tsourakakis. Clustering with a faulty oracle. In *Proceedings of The Web Conference 2020*, pages 2831–2834, 2020.

Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proceedings of the 19th international conference on World wide web*, pages 641–650, 2010.

Claire Mathieu and Warren Schudy. Correlation clustering with noisy input. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 712–728. SIAM, 2010.

Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems*, pages 5788–5799, 2017a.

Arya Mazumdar and Barna Saha. A theoretical analysis of first heuristics of crowdsourced entity resolution. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 970–976, 2017b.

Frank McSherry. Spectral partitioning of random graphs. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 529–537. IEEE, 2001.

Michael Mitzenmacher and Charalampos E Tsourakakis. Predicting signed edges with $o(n^{1+o(1)} \log n$ queries. *arXiv preprint arXiv:1609.00750*, 2016.

Norases Vesdapunt, Kedar Bellare, and Nilesh Dalvi. Crowdsourcing algorithms for entity resolution. *Proceedings of the VLDB Endowment*, 7(12):1071–1082, 2014.

Van Vu. A simple svd algorithm for finding hidden partitions. *Combinatorics, Probability and Computing*, 27(1):124–140, 2018.

Jiannan Wang, Tim Kraska, Michael J Franklin, and Jianhua Feng. Crowder: crowdsourcing entity resolution. *Proceedings of the VLDB Endowment*, 5(11):1483–1494, 2012.

Appendix

A Preliminaries

We will make use of the following Chernoff–Hoeffding bound (see Theorem 1.1 in Dubhashi and Panconesi [2009]).

Theorem 13 (The Chernoff–Hoeffding bound)**.** *Let $t \geq 1$. Let $X := \sum_{1 \leq i \leq t} X_i$, where $X_i, 1 \leq i \leq t$, are independently distributed in $[0, 1]$. Then for all $\lambda > 0$,*

$$\Pr[X > \mathbf{E}[X] + \lambda], \Pr[X < \mathbf{E}[X] - \lambda] \leq e^{-2\lambda^2/t}.$$

B Deferred Proofs from Section 2

B.1 Proof of Theorem 4

We use $G \sim \text{SBM}(N, k, p, q)$ to denote that the graph G is generated from the SBM(N, k, p, q) model. Let C_u be the cluster that contains u, for any $u \in V$. The following was shown by Vu [2018].

Theorem 14 (Theorem 1.2 in Vu [2018]). *Let $G \sim SBM(N, k, p, q)$. Let s be the size of the minimum cluster. There exists a universal constant $c_1 > 20$ such that the following holds. Assume that*

$$\sigma := \sqrt{\max\{p(1-p), q(1-q)\}} \geq c_1 \log N/N, s \geq c_1 \log N, \text{ and } k = o((N/\log N)^{1/2}).$$

Suppose further that for any u, v that belong to two different clusters

$$\sqrt{|C_u| + |C_v|}(p-q) \geq c_1 \left(\sigma \sqrt{\frac{N}{s}} + \sqrt{\log N} \right).$$

Then there exists a polynomial time algorithm $\mathcal{A}$ that recovers all the clusters $V_1, \cdots, V_k$ of G, with probability at least $1 - N^{-8}$.

Now we show that Theorem 4 can be derived the above theorem.

of Theorem 4. Note that to recover the clusters of $G \sim \mathcal{D}(V_1, \ldots, V_k, \delta)$, it suffices to consider the SBM(N, k, p, q) model with $N = n$, k and $p = \frac{1}{2} + \frac{\delta}{2}$ and $q = \frac{1}{2} - \frac{\delta}{2}$. Furthermore, since the corresponding partition is b-balanced, the size of the smallest cluster is $s \geq \frac{bn}{k}$. Let $c_0 = 8c_1^2$, where c_1 is the universal constant from Theorem 14.

Now we claim that the precondition of Theorem 14 is satisfied. By the assumption that $n \geq c_0(k^2 \log n)/(b^2\delta^2)$, it hols that $k = o((N/\log N)^{1/2})$ and $s \geq \frac{bN}{k} \geq c_1 \log N$. Note further that

$$\sigma = \sqrt{(\frac{1}{2} + \frac{\delta}{2})(\frac{1}{2} - \frac{\delta}{2})} = \sqrt{\frac{1}{4} - \frac{\delta^2}{4}} \in [\frac{\sqrt{3}}{4}, \frac{1}{2}] \qquad \Longrightarrow \qquad \sigma \geq c_1 \log N/N$$

where we used the assumption that $\delta \leq \frac{1}{2}$ and that n is sufficiently large.

Furthermore, for any two different clusters, we have $|C_u| + |C_v| \geq \frac{2bn}{k}$. Note that

$$p - q = \delta, \qquad \Longrightarrow \qquad \sqrt{|C_u| + |C_v|}(p-q) \geq \delta \sqrt{\frac{2bn}{k}},$$

$$\sqrt{\frac{N}{s}} \leq \sqrt{\frac{k}{b}}, \qquad \sqrt{\log N} = \sqrt{\log n} \qquad \Longrightarrow \qquad \sigma \sqrt{\frac{N}{s}} + \sqrt{\log N} \leq \frac{1}{2}\sqrt{\frac{k}{b}} + \sqrt{\log n}$$

Then by the precondition that

$$n \geq c_0 \frac{k^2}{b^2\delta^2} \log n \geq c_1^2 \left(\frac{k^2}{4b^2\delta^2} + \frac{k \log n}{b\delta^2} \right)$$

we have that

$$\delta^2 \frac{2bn}{k} \geq 2c_1^2 (\frac{k}{4b} + \log n) \Longrightarrow \delta \sqrt{\frac{2bn}{k}} \geq c_1 \left(\frac{1}{2}\sqrt{\frac{k}{b}} + \sqrt{\log n} \right),$$

where we used the inequality $2x^2 + 2y^2 \geq (x+y)^2$. Thus,

$$\sqrt{|C_u| + |C_v|}(p-q) \geq c_1 \left(\sigma \sqrt{\frac{N}{s}} + \sqrt{\log N} \right).$$

Therefore, by Theorem 14, with probability at least $1 - n^{-8}$, we can recover all the clusters $V_1, \ldots, V_k$ in polynomial time.

□