Correlation Clustering with Adaptive Similarity Queries

Marco Bressan, Nicolo Cesa-Bianchi, Andrea Paudice, Fabio Vitale

Correlation Clustering with Adaptive Similarity Queries

Marco Bressan
Department of Computer Science
University of Rome Sapienza

Nicolò Cesa-Bianchi
Department of Computer Science & DSRC
Università degli Studi di Milano

Andrea Paudice
Department of Computer Science
Università degli Studi di Milano & IIT

Fabio Vitale
Department of Computer Science
University of Lille & Inria

Abstract

In correlation clustering, we are given n objects together with a binary similarity score between each pair of them. The goal is to partition the objects into clusters so to minimise the disagreements with the scores. In this work we investigate correlation clustering as an active learning problem: each similarity score can be learned by making a query, and the goal is to minimise both the disagreements and the total number of queries. On the one hand, we describe simple active learning algorithms, which provably achieve an almost optimal trade-off while giving cluster recovery guarantees, and we test them on different datasets. On the other hand, we prove information-theoretical bounds on the number of queries necessary to guarantee a prescribed disagreement bound. These results give a rich characterization of the trade-off between queries and clustering error.

1 Introduction

Clustering is a central problem in unsupervised learning. A clustering problem is typically represented by a set of elements together with a notion of similarity (or dissimilarity) between them. When the elements are points in a metric space, dissimilarity can be measured via a distance function. In more general settings, when the elements to be clustered are members of an abstract set V, similarity is defined by an arbitrary symmetric function σ defined on pairs of distinct elements in V. Correlation Clustering (CC) [3] is a well-known special case where σ is a $\{-1, +1\}$-valued function establishing whether any two distinct elements of V are similar or not. The objective of CC is to cluster the points in V so to maximize the correlation with σ. More precisely, CC seeks a clustering minimizing the number of errors, where an error is given by any pair of elements having similarity -1 and belonging to the same cluster, or having similarity $+1$ and belonging to different clusters. Importantly, there are no a priori limitations on the number of clusters or their sizes: all partitions of V, including the trivial ones, are valid. Given V and σ, the error achieved by an optimal clustering is known as the *Correlation Clustering index*, denoted by OPT. A convenient way of representing σ is through a graph $G = (V, E)$ where $\{u, v\} \in E$ iff $\sigma(u, v) = +1$. Note that $\mathrm{OPT} = 0$ is equivalent to a perfectly clusterable graph (i.e., G is the union of disjoint cliques). Since its introduction, CC has attracted a lot of interest in the machine learning community, and has found numerous applications in entity resolution [16], image analysis [18], and social media analysis [24]. Known problems in data integration [13] and biology [4] can be cast into the framework of CC [25].

From a machine learning viewpoint, we are interested in settings when the similarity function σ is not available beforehand, and the algorithm must learn σ by querying for its value on pairs of objects. This setting is motivated by scenarios in which the similarity information is costly to obtain. For

Preprint. Under review.

example, in entity resolution, disambiguating between two entities may require invoking the user's help. Similarly, deciding if two documents are similar may require a complex computation, and possibly the interaction with human experts. In these active learning settings, the learner's goal is to trade the clustering error against the number of queries. Hence, the fundamental question is: how many queries are needed to achieve a specified clustering error? Or, in other terms, how close can we get to OPT, under a prescribed query budget Q?

1.1 Our Contributions

In this work we characterize the trade-off between the number Q of queries and the clustering error on n points. The table below here summarizes our bounds in the context of previous work. Running time and upper/lower bounds on the expected clustering error are expressed in terms of the number of queries Q, and all our upper bounds assume $Q = \Omega(n)$ while our lower bounds assume $Q = \mathcal{O}(n^2)$.

Running time	Expected clustering error	Reference
Q + LP solver + rounding	$3(\ln n + 1)\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$	[6]
Q	$3\mathrm{OPT} + \mathcal{O}(n^3/Q)$	Theorem 1 (see also [5])
Exponential	$\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$	Theorem 7
Exponential ($\mathrm{OPT} = 0$)	$\widetilde{\mathcal{O}}\big(n^3/Q\big)$	Theorem 7
Unrestricted ($\mathrm{OPT} = 0$)	$\Omega\big(n^2/\sqrt{Q}\big)$	Theorem 8
Unrestricted ($\mathrm{OPT} \gg 0$)	$\mathrm{OPT} + \Omega\big(n^3/Q\big)$	Theorem 9

Our first set of contributions is algorithmic. We take inspiration from an existing greedy algorithm, KwikCluster [2], that has expected error $3\mathrm{OPT}$ but a vacuous $\mathcal{O}(n^2)$ worst-case bound on the number of queries. We propose a variant of KwikCluster, called ACC, for which we prove several desirable properties. First, ACC achieves expected clustering error $3\mathrm{OPT} + \mathcal{O}(n^3/Q)$, where $Q = \Omega(n)$ is a deterministic bound on the number of queries. In particular, if ACC is run with $Q = \binom{n}{2}$, then it becomes exactly equivalent to KwikCluster. Second, ACC recovers adversarially perturbed latent clusters. More precisely, if the input contains a cluster C obtained from a clique by adversarially perturbing a fraction ε of its edges (internal to the clique or leaving the clique), then ACC returns a cluster $\widehat{C}$ such that $\mathbb{E}\big[|C \oplus \widehat{C}|\big] = \mathcal{O}\big(\varepsilon|C| + n^2/Q\big)$, where $\oplus$ denotes symmetric difference. This means that ACC recovers almost completely all perturbed clusters that are large enough to be "seen" with Q queries. We also show, under stronger assumptions, that via independent executions of ACC one can recover exactly all large clusters with high probability. Third, we show a variant of ACC, called ACCESS (for Early Stopping Strategy), that makes significantly less queries on some graphs. For example, when $\mathrm{OPT} = 0$ and there are $\Omega\big(n^3/Q\big)$ similar pairs, the expected number of queries made by ACCESS is only the square root of the queries made by ACC. In exchange, ACCESS makes at most Q queries in expectation rather than deterministically.

Our second set of contributions is a nearly complete information-theoretic characterization of the query vs. clustering error trade-off (thus, ignoring computational efficiency). Using VC theory, we prove that for all $Q = \Omega(n)$ the strategy of minimizing disagreements on a random subset of pairs achieves, with high probability, clustering error bounded by $\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$, which reduces to $\widetilde{\mathcal{O}}\big(n^3/Q\big)$ when $\mathrm{OPT} = 0$. The VC theory approach can be applied to any efficient approximation algorithm, too. The catch is that the approximation algorithm cannot ask the similarity of arbitrary pairs, but only of pairs included in the random sample of edges. The best known approximation factor in this case is $3(\ln n + 1)$ [14], which gives a clustering error bound of $3(\ln n + 1)\mathrm{OPT} + \mathcal{O}\big(n^{5/2}/\sqrt{Q}\big)$ with high probability. This was already observed in [6] albeit in a slightly different context.

We complement our upper bounds by developing two information-theoretic lower bounds; these lower bounds apply to any algorithm issuing $Q = \mathcal{O}(n^2)$ queries, possibly chosen in an adaptive way. For the general case, we show that any algorithm must suffer an expected clustering error of at least $\mathrm{OPT} + \Omega\big(n^3/Q\big)$. In particular, for $Q = \Theta(n^2)$ any algorithm still suffers an additive error of order n, and for $Q = \Omega(n)$ our algorithm ACC is essentially optimal in its additive error term. For the special case $\mathrm{OPT} = 0$, we show a lower bound $\Omega\big(n^2/\sqrt{Q}\big)$.

Finally, we evaluate our algorithms empirically on real-world and synthetic datasets.

2 Related work

Minimizing the correlation clustering error is APX-hard [8], and the best efficient algorithm found so far achieves $2.06\,\mathrm{OPT}$ [9]. This almost matches the best possible approximation factor 2 achievable via the natural LP relaxation of the problem [8]. A very simple and elegant algorithhm for approximating CC is KwikCluster [2]. At each round, KwikCluster draws a random pivot π_r from V, queries the similarities between π_r and every other node in V, and creates a cluster C containing π_r and all points u such that $\sigma(\pi_r, u) = +1$. The algorithm then recursively invokes itself on $V \setminus C$. On any instance of CC, KwikCluster achieves an expected error bounded by $3\mathrm{OPT}$. However, it is easy to see that KwikCluster makes $\Theta(n^2)$ queries in the worst case (e.g., if σ is the constant function -1). Our algorithms can be seen as a parsimonious version of KwikCluster whose goal is reducing the number of queries.

The work closest to ours is [5]. Their algorithm runs KwikCluster on a random subset of $1/(2\varepsilon)$ nodes and stores the set Π of resulting pivots. Then, each node $v \in V \setminus \Pi$ is assigned to the cluster identified by the pivot $\pi \in \Pi$ with smallest index and such that $\sigma(v, \pi) = +1$. If no such pivot is found, then v becomes a singleton cluster. According to [5, Lemma 4.1], the expected clustering error for this variant is $3\mathrm{OPT} + \mathcal{O}\big(\varepsilon n^2\big)$, which can be compared to our bound for ACC by setting $Q = n/\varepsilon$. On the other hand our algorithms are much simpler and significantly easier to analyze. This allows us to prove a set of additional properties, such as cluster recovery and instance-dependent query bounds. It is unclear whether these results are obtainable with the techniques of [5].

Another line of work attempts to circumvent computational hardness by using the more powerful same-cluster queries (SCQ). A same-cluster query tells whether any two given nodes are clustered together according to an optimal clustering or not. In [**?**] SCQs are used to design a FPTAS for a variant of CC with bounded number of clusters. In [**?**] SCQs are used to design algorithms for solving CC optimally by giving bounds on Q which depend on OPT. Unlike our setting, both works assume *all* $\binom{n}{2}$ similarities are known in advance. The work [21] considers the case in which there is a latent clustering with $\mathrm{OPT} = 0$. The algorithm can issue SCQs, however the oracle is noisy: each query is answered incorrectly with some probability, and the noise is persistent (repeated queries give the same noisy answer). The above setting is closely related to the stochastic block model (SBM), which is a well-studied model for cluster recovery [1, 19, 22]. However, few works investigate SBMs with pairwise queries [11]. Our setting is strictly harder because our oracle has a budget of OPT adversarially incorrect answers.

A different model is edge classification. Here the algorithm is given a graph $\mathcal{G}$ with hidden binary labels on the edges. The task is to predict the sign of all edges by querying as few labels as possible [6, 10, 12]. As before, the oracle can have a budget OPT of incorrect answers, or a latent clustering with $\mathrm{OPT} = 0$ is assumed and the oracle's answers are affected by persistent noise. Unlike correlation clustering, in edge classification the algorithm is not constrained to predict in agreement with a partition of the nodes. On the other hand, the algorithm cannot query arbitrary pairs of nodes in V, but only those that form an edge in $\mathcal{G}$.

Preliminaries and notation. We denote by $V \equiv \{1, \dots, n\}$ the set of input nodes, by $\mathcal{E} \equiv \binom{V}{2}$ the set of all pairs $\{u, v\}$ of distincts nodes in V, and by $\sigma : \mathcal{E} \to \{-1, +1\}$ the binary similarity function. A clustering $\mathcal{C}$ is a partition of V in disjoint clusters $C_i : i = 1, \dots, k$. Given $\mathcal{C}$ and σ, the set $\Gamma_{\mathcal{C}}$ of mistaken edges contains all pairs $\{u, v\}$ such that $\sigma(u, v) = -1$ and u, v belong to same cluster of $\mathcal{C}$ and all pairs $\{u, v\}$ such that $\sigma(u, v) = +1$ and u, v belong to different clusters of $\mathcal{C}$. The cost $\Delta_{\mathcal{C}}$ of $\mathcal{C}$ is $\big|\Gamma_{\mathcal{C}}\big|$. The correlation clustering index is $\mathrm{OPT} = \min_{\mathcal{C}} \Delta_{\mathcal{C}}$, where the minimum is over all clusterings $\mathcal{C}$. We often view V, σ as a graph $G = (V, E)$ where $\{u, v\} \in E$ is an edge if and only if $\sigma(u, v) = +1$. In this case, for any subset $U \subseteq V$ we let $G[U]$ be the subgraph of G induced by U, and for any $v \in V$ we let $\mathcal{N}_v$ be the neighbor set of v.

A triangle is any unordered triple $T = \{u, v, w\} \subseteq V$. We denote by $e = \{u, w\}$ a generic triangle edge; we write $e \subset T$ and $v \in T \setminus e$. We say T is a *bad triangle* if the labels $\sigma(u, v), \sigma(u, w), \sigma(v, w)$ are $\{+, +, -\}$ (the order is irrelevant). We denote by $\mathcal{T}$ the set of all bad triangles in V. It is easy to see that the number of edge-disjoint bad triangles is a lower bound on OPT.

Due to space limitations, here most of our results are stated without proof, or with a concise proof sketch; the full proofs can be found in the supplementary material.

3 The ACC algorithm

We introduce our active learning algorithm ACC (Active Correlation Clustering).

Algorithm 1 Invoked as $\mathrm{ACC}(V_1, 1)$ where $V_1 \equiv V$ and $r = 1$ is the index of the recursive call.

Parameters: Query rate function $f : \mathbb{N} \to \mathbb{N}$.

1: **if** $|V_r| = 0 \vee r > f(|V_1| - 1)$ **then** RETURN
2: Draw pivot π_r u.a.r. from V_r
3: $C_r \leftarrow \{\pi_r\}$ ▷ Create new cluster and add the pivot to it
4: Draw a random subset S_r of $f(|V_r| - 1)$ nodes from $V_r \setminus \{\pi_r\}$
5: **for** each $u \in S_r$ **do** query $\sigma(\pi_r, u)$
6: **if** $\exists\, u \in S_r$ such that $\sigma(\pi_r, u) = +1$ **then** ▷ Check if there is at least a positive edge
7: Query all remaining pairs (π_r, u) for $u \in V_r \setminus \big(\{\pi_r\} \cup S_r\big)$
8: $C_r \leftarrow C_r \cup \{u : \sigma(\pi_r, u) = +1\}$ ▷ Populate cluster based on queries
9: Output cluster C_r
10: $\mathrm{ACC}(V_r \setminus C_r, r + 1)$ ▷ Recursive call on the remaining nodes

ACC has the same recursive structure as KwikCluster. First, it starts with the full instance $V_1 = V$. Then, for each round $r = 1, 2, \ldots$ it selects a random pivot $\pi_r \in V_r$, queries the similarities between π_r and a subset of V_r, removes π_r and possibly other points from V_r, and proceeds on the remaining residual subset V_{r+1}. However, while KwikCluster queries $\sigma(\pi_r, u)$ for *all* $u \in V_r \setminus \{\pi_r\}$, ACC queries only $f(n_r) \leq n_r$ other nodes u (lines 4–5), where $n_r = |V_r| - 1$. Thus, while KwikCluster always finds all positive labels involving the pivot π_r, ACC can find them or not, with a probability that depends on f. The function f is called *query rate function* and dictates the tradeoff between the clustering cost Δ and the number of queries Q, as we prove below. Now, if any of the aforementioned $f(n_r)$ queries returns a positive label (line 6), then all the labels between π_r and the remaining $u \in V_r$ are queried and the algorithm operates as KwikCluster until the end of the recursive call; otherwise, the pivot becomes a singleton cluster which is removed from the set of nodes. Another important difference is that ACC deterministically stops after $f(n)$ recursive calls (line 1), declaring all remaining points as singleton clusters. The intuition is that with good probability the clusters not found within $f(n)$ rounds are small enough to be safely disregarded. Since the choice of f is delicate, we avoid trivialities by assuming f is positive, integral, and smooth enough. Formally:

Definition 1. *$f : \mathbb{N} \to \mathbb{N}$ is a* query rate function *if $f(1) = 1$ and $f(n) \leq f(n+1) \leq \big(1 + \frac{1}{n}\big) f(n)$ for all $n \in \mathbb{N}$. This implies $\frac{f(n+k)}{n+k} \leq \frac{f(n)}{n}$ for all $k \geq 1$.*

We can now state formally our bounds for ACC.

Theorem 1. *For any query rate function f and any labeling σ on n nodes, the expected cost $\mathbb{E}[\Delta_A]$ of the clustering output by* ACC *satisfies*

$$\mathbb{E}[\Delta_A] \leq 3\mathrm{OPT} + \frac{2e-1}{2(e-1)} \frac{n^2}{f(n)} + \frac{n}{2e} \,.$$

The number of queries made by ACC *is deterministically bounded as $Q \leq n f(n)$. In the special case $f(n) = n$ for all $n \in \mathbb{N}$,* ACC *reduces to $KwikCluster$ and achieves $\mathbb{E}[\Delta_A] \leq 3\mathrm{OPT}$ with $Q \leq n^2$.*

Note that Theorem 1 gives an upper bound on the error achievable when using Q queries: since $Q = nf(n)$, the expected error is at most $3\mathrm{OPT} + \mathcal{O}(n^3/Q)$. Furthermore, as one expects, if the learner is allowed to ask for all edge signs, then the *exact* bound of $KwikCluster$ is recovered (note that the first formula in Theorem 1 clearly does not take into account the special case when $f(n) = n$, which is considered in the last part of the statement).

Proof sketch. Look at a generic round r, and consider a pair of points $\{u, w\} \in V_r$. The essence is that ACC can misclassify $\{u, w\}$ in one of two ways. First, if $\sigma(u, w) = -1$, ACC can choose as pivot π_r a node v such that $\sigma(v, u) = \sigma(v, w) = +1$. In this case, if the condition on line 6 holds, then ACC will cluster v together with u and w, thus mistaking $\{u, w\}$. If instead $\sigma(u, w) = +1$,

then ACC could mistake $\{u, w\}$ by pivoting on a node v such that $\sigma(v, u) = +1$ and $\sigma(v, w) = -1$, and clustering together only v and u. Crucially, both cases imply the existence of a bad triangle $T = \{u, w, v\}$. We charge each such mistake to exactly one bad triangle T, so that no triangle is charged twice. The expected number of mistakes can then be bound by 3OPT using the packing argument of [2] for KwikCluster. Second, if $\sigma(u, w) = +1$ then ACC could choose one of them, say u, as pivot π_r, and assign it to a singleton cluster. This means the condition on line 6 fails. We can then bound the number of such mistakes as follows. Suppose π_r has $cn/f(n)$ positive labels towards V_r for some $c \geq 0$. Loosely speaking, we show that the check of line 6 fails with probability e^{-c}, in which case $cn/f(n)$ mistakes are added. In expectation, this gives $cne^{-c}/f(n) = \mathcal{O}\big(n/f(n)\big)$ mistakes. Over all $f(n) \leq n$ rounds, this gives an overall $\mathcal{O}\big(n^2/f(n)\big)$. (The actual proof has to take into account that all the quantities involved here are not constants, but random variables).

3.1 ACC with Early Stopping Strategy

We can refine our algorithm ACC so that, in some cases, it takes advantage of the structure of the input to reduce significantly the expected number of queries. To this end we see the input as a graph G with edges corresponding to positive labels (see above). Suppose then G contains a sufficiently small number $\mathcal{O}(n^2/f(n))$ of edges. Since ACC deterministically performs $f(n-1)$ rounds, it could make $Q = \Theta(f(n)^2)$ queries. However, with just $Q = f(n)$ queries one could *detect* that G contains $\mathcal{O}(n^2/f(n))$ edges, and immediately return the trivial clustering formed by all singletons. The expected error would obviously be at most $\mathrm{OPT} + \mathcal{O}(n^2/f(n))$, i.e. the same of Theorem 1. More generally, at each round r with $f(n_r)$ queries one can check if the residual graph contains at least $n^2/f(n)$ edges; if the test fails, declaring all nodes in V_r as singletons gives expected additional error $\mathcal{O}(n^2/f(n))$. The resulting algorithm is a variant of ACC that we call ACCESS (ACC with Early Stopping Strategy). The pseudocode can be found in the supplementary material.

First, we show ACCESS gives guarantees virtually identical to ACC (only, with Q in expectation). Formally:

Theorem 2. *For any query rate function f and any labeling σ on n nodes, the expected cost $\mathbb{E}[\Delta_A]$ of the clustering output by* ACCESS *satisfies*

$$\mathbb{E}[\Delta_A] \leq 3\mathrm{OPT} + \frac{n^2}{ef(n)} + \frac{n}{2e}\,.$$

Moreover, the expected number of queries performed by ACCESS *is $\mathbb{E}[Q] \leq n(2f(n)+1)$.*

Theorem 2 reassures us that ACCESS is no worse than ACC. In fact, if most edges of G belong to relatively large clusters (namely, all but $O(n^2/f(n))$ edges), then we can show ACCESS uses much fewer queries than ACC (in a nutshell, ACCESS quickly finds all large clusters and then quits). The following theorem captures the essence. For simplicity we assume $\mathrm{OPT} = 0$, i.e. G is a disjoint union of cliques.

Theorem 3. *Suppose $\mathrm{OPT} = 0$ so G is a union of disjoint cliques. Let $C_1, \ldots, C_\ell$ be the cliques of G in nondecreasing order of size. Let i' be the smallest i such that $\sum_{j=1}^{i} |E_{C_j}| = \Omega(n^2/f(n))$, and let $h(n) = |C_{i'}|$. Then* ACCESS *makes in expectation $\mathbb{E}[Q] = \mathcal{O}\big(n^2 \lg(n)/h(n)\big)$ queries.*

As an example, say $f(n) = \sqrt{n}$ and G contains $n^{1/3}$ cliques of $n^{2/3}$ nodes each. Then for ACC Theorem 1 gives $Q \leq nf(n) = \mathcal{O}(n^{3/2})$, while for ACCESS Theorem 3 gives $\mathbb{E}[Q] = \mathcal{O}(n^{4/3}\lg(n))$.

4 Cluster recovery

In the previous section we gave bounds on $\mathbb{E}[\Delta]$, the expected *total* cost of the clustering. However, in applications such as community detection and alike, the primary objective is recovering accurately the latent clusters of the graph, the sets of nodes that are "close" to cliques. This is usually referred to as *cluster recovery*. For this problem, an algorithm that outputs a good approximation $\widehat{C}$ of every latent cluster C is preferrable to an algorithm that minimizes $\mathbb{E}[\Delta]$ globally. In this section we show that ACC natively outputs clusters that are close to the latent clusters in the graph, thus acting as a cluster recovery tool. We also show that, for a certain type of latent clusters, one can amplify the accuracy of ACC via independent executions and recover all clusters exactly with high probability.

To capture the notion of "latent cluster", we introduce the concept of $(1-\varepsilon)$-*knit* set. As usual, we view V, σ as a graph $G = (V, E)$ with $e \in E$ iff $\sigma(e) = +1$. Let E_C be the edges in the subgraph induced by $C \subseteq V$ and $\mathrm{cut}(C, \overline{C})$ be the edges between C and $\overline{C} = V \setminus C$.

Definition 2. *A subset $C \subseteq V$ is* $(1-\varepsilon)$-knit *if* $\big|E_C\big| \geq (1-\varepsilon)\binom{|C|}{2}$ *and* $\big|\mathrm{cut}(C, \overline{C})\big| \leq \varepsilon\binom{|C|}{2}$.

Suppose now we have a cluster $\widehat{C}$ as "estimate" of C. We quantify the distance between C and $\widehat{C}$ as the cardinality of their symmetric difference, $\big|\widehat{C} \oplus C\big| = \big|\widehat{C} \setminus C\big| + \big|C \setminus \widehat{C}\big|$. The goal is to obtain, for each $(1-\varepsilon)$-knit set C in the graph, a cluster $\widehat{C}$ with $|\widehat{C} \oplus C| = \mathcal{O}(\varepsilon|C|)$ for some small ε. We prove ACC does exactly this. Clearly, we must accept that if C is too small, i.e. $|C| = o(n/f(n))$, then ACC will miss C entirely. But, for $|C| = \Omega(n/f(n))$, we can prove $\mathbb{E}[|\widehat{C} \oplus C|] = \mathcal{O}(\varepsilon|C|)$. We point out that the property of being $(1-\varepsilon)$-knit is rather weak for an algorithm, like ACC, that is completely oblivious to the global topology of the cluster — all what ACC tries to do is to blindly cluster together all the neighbors of the current pivot. In fact, consider a set C formed by two disjoint cliques of equal size. This set would be close to $1/2$-knit, and yet ACC would never produce a single cluster $\widehat{C}$ corresponding to C. Things can only worsen if we consider also the edges in $\mathrm{cut}(C, \overline{C})$, which can lead ACC to assign the nodes of C to several different clusters when pivoting on $\overline{C}$. Hence it is not obvious that a $(1-\varepsilon)$-knit set C can be efficiently recovered by ACC.

Note that this task can be seen as an *adversarial* cluster recovery problem. Initially, we start with a disjoint union of cliques, so that $\mathrm{OPT} = 0$. Then, an adversary flips the signs of some of the edges of the graph. The goal is to retrieve every original clique that has not been perturbed excessively. Note that we put no restriction on how the adversary can flip edges; therefore, this adversarial setting subsumes constrained adversaries. For example, it subsumes the high-probability regime of the stochastic block model [17] where edges are flipped according to some distribution.

We can now state our main cluster recovery bound for ACC.

Theorem 4. *For every $C \subseteq V$ that is $(1-\varepsilon)$-knit,* ACC *outputs a cluster $\widehat{C}$ such that* $\mathbb{E}\big[|C \oplus \widehat{C}|\big] \leq 3\varepsilon|C| + \min\big\{\frac{2n}{f(n)}, \big(1 - \frac{f(n)}{n}\big)|C|\big\} + |C|e^{-|C|f(n)/5n}$.

The $\min$ in the bound captures two different regimes: when $f(n)$ is very close to n, then $\mathbb{E}\big[|C \oplus \widehat{C}|\big] = \mathcal{O}(\varepsilon|C|)$ independently of the size of C, but when $f(n) \ll n$ we need $|C| = \Omega(n/f(n))$, i.e., $|C|$ must be large enough to be found by ACC.

4.1 Exact cluster recovery via amplification

For certain latent clusters, one can get recovery guarantees significantly stronger than the ones given natively by ACC (see Theorem 4). We start by introducing *strongly* $(1-\varepsilon)$-*knit* sets (also known as quasi-cliques). Recall that $\mathcal{N}_v$ is the neighbor set of v in the graph G induced by the positive labels.

Definition 3. *A subset $C \subseteq V$ is* strongly $(1-\varepsilon)$-knit *if, for every $v \in C$, we have $\mathcal{N}_v \subseteq C$ and $|\mathcal{N}_v| \geq (1-\varepsilon)(|C| - 1)$.*

We remark that ACC alone does not give better guarantees on strongly $(1-\varepsilon)$-knit subsets than on $(1-\varepsilon)$-knit subsets. Suppose for example that $|\mathcal{N}_v| = (1-\varepsilon)(|C| - 1)$ for all $v \in C$. Then C is strongly $(1-\varepsilon)$-knit, and yet when pivoting on any $v \in C$ ACC will inevitably produce a cluster $\widehat{C}$ with $|\widehat{C} \oplus C| \geq \varepsilon|C|$, since the pivot has edges to less than $(1-\varepsilon)|C|$ other nodes of C.

To bypass this limitation, we run ACC several times to amplify the probability that every node in C is found. Recall that $V = [n]$. Then, we define the id of a cluster $\widehat{C}$ as the smallest node of $\widehat{C}$. The min-tagging rule is the following: when forming $\widehat{C}$, use its id to tag all of its nodes. Therefore, if $u_{\widehat{C}} = \min\{u \in \widehat{C}\}$ is the id of $\widehat{C}$, we will set $\mathrm{id}(v) = u_{\widehat{C}}$ for every $v \in \widehat{C}$. Consider now the following algorithm, called ACR (Amplified Cluster Recovery). First, ACR performs K independent runs of ACC on input V, using the min-tagging rule on each run. In this way, for each $v \in V$ we obtain K tags $\mathrm{id}_1(v), \ldots, \mathrm{id}_K(v)$, one for each run. Thereafter, for each $v \in V$ we select the tag that v has received most often, breaking ties arbitrarily. Finally, nodes with the same tag are clustered together. One can prove that, with high probability, this clustering contains all strongly $(1-\varepsilon)$-knit sets. In other words, ACR with high probability recovers all such latent clusters *exactly*. Formally, we prove:

Theorem 5. *Let $\varepsilon \leq \frac{1}{10}$ and fix $p > 0$. If* ACR *is run with $K = 48 \ln \frac{n}{p}$, then the following holds with probability at least $1 - p$: for every strongly $(1-\varepsilon)$-knit C with $|C| > 10\frac{n}{f(n)}$, the algorithm outputs a cluster $\widehat{C}$ such that $\widehat{C} = C$.*

It is not immediately clear that one can extend this result by relaxing the notion of strongly $(1-\varepsilon)$-knit set so to allow for edges between C and the rest of the graph. We just notice that, in that case, every node $v \in C$ could have a neighbor $x_v \in V \setminus C$ that is smaller than every node of C. In this case, when pivoting on v ACC would tag v with x rather than with u_C, disrupting ACR.

5 A fully additive scheme

In this section, we introduce a(n inefficient) fully additive approximation algorithm achieving cost $\mathrm{OPT} + n^2\varepsilon$ in high probability using order of $\frac{n}{\varepsilon^2}$ queries. When $\mathrm{OPT} = 0$, $Q = \frac{n}{\varepsilon} \ln \frac{1}{\varepsilon}$ suffices. Our algorithm combines uniform sampling with empirical risk minimization and is analyzed using VC theory.

First, note that CC can be formulated as an agnostic binary classification problem with binary classifiers $h_{\mathcal{C}} : \mathcal{E} \to \{-1, +1\}$ associated with each clustering $\mathcal{C}$ of V (recall that $\mathcal{E}$ denotes the set of all pairs $\{u, v\}$ of distinct elements $u, v \in V$), and we assume $h_{\mathcal{C}}(u, v) = +1$ iff u and v belong to the same cluster of $\mathcal{C}$. Let $\mathcal{H}_n$ be the set of all such $h_{\mathcal{C}}$. The risk of a classifier $h_{\mathcal{C}}$ with respect to the uniform distribution over $\mathcal{E}$ is $\mathbb{P}(h_{\mathcal{C}}(e) \neq \sigma(e))$ where e is drawn u.a.r. from $\mathcal{E}$. It is easy to see that the risk of any classifier $h_{\mathcal{C}}$ is directly related to $\Delta_{\mathcal{C}}$, $\mathbb{P}\big(h_{\mathcal{C}}(e) \neq \sigma(e)\big) = \Delta_{\mathcal{C}} / \binom{n}{2}$. Hence, in particular, $\mathrm{OPT} = \binom{n}{2} \min_{h \in \mathcal{H}_n} \mathbb{P}\big(h(e) \neq \sigma(e)\big)$. Now, it is well known —see, e.g., [23, Theorem 6.8]— that we can minimize the risk to whithin an additive term of ε using the following procedure: query $\mathcal{O}\big(d/\varepsilon^2\big)$ edges drawn u.a.r. from $\mathcal{E}$, where d is the VC dimension of $\mathcal{H}_n$, and find the clustering $\mathcal{C}$ such that $h_{\mathcal{C}}$ makes the fewest mistakes on the sample. If there is $h^* \in \mathcal{H}_n$ with zero risk, then $\mathcal{O}\big((d/\varepsilon)\ln(1/\varepsilon)\big)$ random queries suffice. A trivial upper bound on the VC dimension of $\mathcal{H}_n$ is $\log_2 |\mathcal{H}_n| = \mathcal{O}\big(n \ln n\big)$. The next result gives the exact value.

Theorem 6. *The VC dimension of the class $\mathcal{H}_n$ of all partitions of n elements is $n - 1$.*

Proof. Let d be the VC dimension of $\mathcal{H}_n$. We view an instance of CC as the complete graph K_n with edges labelled by σ. Let T be any spanning tree of K_n. For any labeling σ, we can find a clustering of V such that h perfectly classifies the edges of T: simply remove the edges with label -1 in T and consider the clusters formed by the resulting connected components. Hence $d \geq n - 1$ because any spanning tree has exactly $n - 1$ edges. On the other hand, any set of n edges must contain at least a cycle. It is easy to see that no clustering $\mathcal{C}$ makes $h_{\mathcal{C}}$ consistent with the labeling σ that gives positive labels to all edges in the cycle but one. Hence $d < n$. □

An immediate consequence of the above is the following.

Theorem 7. *There exists a randomized algorithm A that, for all $0 < \varepsilon < 1$, finds a clustering $\mathcal{C}$ satisfying $\Delta_{\mathcal{C}} \leq \mathrm{OPT} + \mathcal{O}\big(n^2\varepsilon\big)$ with high probability while using $Q = \mathcal{O}\big(\frac{n}{\varepsilon^2}\big)$ queries. Moreover, if $\mathrm{OPT} = 0$, then $Q = \mathcal{O}\big(\frac{n}{\varepsilon} \ln \frac{1}{\varepsilon}\big)$ queries are enough to find a clustering $\mathcal{C}$ satisfying $\Delta_{\mathcal{C}} = \mathcal{O}\big(n^2\varepsilon\big)$.*

6 Lower bounds

In this section we give two lower bounds on the expected clustering error of any (possibly randomized) algorithm. The first bound holds for $\mathrm{OPT} = 0$, and applies to algorithms using a deterministically bounded number of queries. This bound is based on a construction from [7, Lemma 11] and related to kernel-based learning.

Theorem 8. *For any $\varepsilon > 0$ such that $\frac{1}{\varepsilon}$ is an even integer, and for every (possibly randomized) learning algorithm asking fewer than $\frac{1}{50\varepsilon^2}$ queries with probability 1, there exists a labeling σ on $n \geq \frac{16}{\varepsilon} \ln \frac{1}{\varepsilon}$ nodes such that $\mathrm{OPT} = 0$ and the expected cost of the algorithm is at least $\frac{n^2\varepsilon}{8}$.*

Our second bound relaxed the assumption on OPT. It uses essentially the same construction of [5, Lemma 6.1], giving asymptotically the same guarantees. However, the bound of [5] applies only to

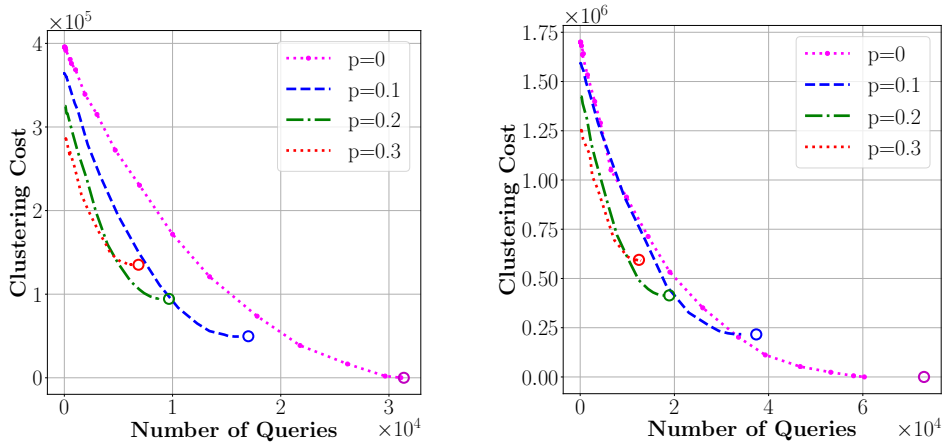
a very restricted class of algorithms: namely, those where the number q_v of queries involving any specific node $v \in V$ is deterministically bounded. This rules out a vast class of algorithms, including KwikCluster, ACC, and ACCESS, where the number of queries involving a node is a function of the random choices of the algorithm. Our lower bound is instead fully general: it holds unconditionally for *any* randomized algorithm, with no restriction on what or how many pairs of points are queried.

Theorem 9. *Choose any function $\varepsilon = \varepsilon(n)$ such that $\Omega\left(\frac{1}{n}\right) \leq \varepsilon \leq \frac{1}{2}$ and $\frac{1}{\varepsilon} \in \mathbb{N}$. For every (possibly randomized) learning algorithm and any $n_0 > 0$ there exists a labeling σ on $n \geq n_0$ nodes such that the algorithm has expected error $\mathbb{E}[\Delta] \geq \mathrm{OPT} + \frac{n^2\varepsilon}{80}$ whenever its expected number of queries satisfies $\mathbb{E}[Q] < \frac{n}{80\,\varepsilon}$.*

In fact, the bound of Theorem 9 can be put in a more general form: for any constant $c \geq 1$, the expected error is at least $c \cdot \mathrm{OPT} + A(c)$ where $A(c) = \Omega(n^2\varepsilon)$ is an additive term with constant factors depending on c (see the proof). Thus, our algorithms ACC and ACCESS are essentially optimal in the sense that, for $c = 3$, they guarantee an optimal additive error up to constant factors.

7 Experiments

We tested ACC on six datasets from [21, 20]. Four of these datasets are obtained from real-world data and the remaining two are synthetic. In Figure 2 we show our results for one real-world dataset (cora, with 1879 nodes and 191 clusters) and one synthetic dataset (skew, with 900 nodes and 30 clusters). Similar results for the remaining four datasets can be found in the supplementary material. Every dataset provides a ground-truth partitioning of nodes with $\mathrm{OPT} = 0$. To test the algorithm for $\mathrm{OPT} > 0$, we perturbed the dataset by flipping the label of each edge indipendently with probability p (so the results for $p = 0$ refer to the original dataset with $\mathrm{OPT} = 0$).



(a) skew. (b) cora.

Figure 1: Clustering cost vs. number of queries. The curves show the average value of Δ. The circular outliers mark the performance of KwikCluster.

Figure 2 shows the measured clustering cost Δ against the number of queries Q performed by ACC. For each value of $p \in \{0, 0.1, 0.2, 0.3\}$, each curve in the plot is obtained by setting the query rate $f(n)$ to n^α for 20 distinct values of $\alpha \in [0, 3/4]$. For each value of α we ran ACC fifty times. The curve shows the average value of Δ (standard deviations, which are small, are omitted to avoid cluttering the figure). The performance of KwikCluster is shown by the circular marker. On both datasets, the error of ACC shows a nice sublinear drop as the number of queries increases, quickly approaching the performance of KwikCluster. Ignoring lower order terms, Theorem 1 gives an expected cost bounded by about $3.8n^3/Q$ for the case $\mathrm{OPT} = 0$ (recall that OPT is unknown). Placing this curve in our plots, shows that ACC is a factor of two or three better than the theoretical bound (which is not shown in Figure 2 due to scaling issues).

Acknowledgments

The authors gratefully acknowledge partial support by the Google Focused Award "Algorithms and Learning for AI" (ALL4AI). Marco Bressan and Fabio Vitale are also supported in part by the ERC Starting Grant DMAP 680153 and by the "Dipartimenti di Eccellenza 2018-2022" grant awarded to the Department of Computer Science of the Sapienza University of Rome. Nicolò Cesa-Bianchi is also supported by the MIUR PRIN grant *Algorithms, Games, and Digital Markets* (ALGADIMAR)

References

[1] Emmanuel Abbe and Colin Sandon. Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 670–688. IEEE, 2015.

[2] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *J. ACM*, 55(5):23:1–23:27, 2008.

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

[4] Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of computational biology*, 6(3-4):281–297, 1999.

[5] Francesco Bonchi, David García-Soriano, and Konstantin Kutzkov. Local correlation clustering. *arXiv preprint arXiv:1312.5105*, 2013.

[6] Nicolo Cesa-Bianchi, Claudio Gentile, Fabio Vitale, and Giovanni Zappella. A correlation clustering approach to link classification in signed networks. In *Annual Conference on Learning Theory*, volume 23, pages 34–1. Microtome, 2012.

[7] Nicolo Cesa-Bianchi, Yishay Mansour, and Ohad Shamir. On the complexity of learning with kernels. In *Conference on Learning Theory*, pages 297–325, 2015.

[8] Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *Journal of Computer and System Sciences*, 71(3):360–383, 2005.

[9] Shuchi Chawla, Konstantin Makarychev, Tselil Schramm, and Grigory Yaroslavtsev. Near optimal lp rounding algorithm for correlationclustering on complete and complete k-partite graphs. In *Proceedings of the Forty-seventh Annual ACM Symposium on Theory of Computing*, STOC '15, pages 219–228, New York, NY, USA, 2015. ACM.

[10] 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, 2014.

[11] Yuxin Chen, Govinda Kamath, Changho Suh, and David Tse. Community recovery in graphs with locality. In *International Conference on Machine Learning*, pages 689–698, 2016.

[12] Kai-Yang Chiang, Cho-Jui Hsieh, Nagarajan Natarajan, Inderjit S Dhillon, and Ambuj Tewari. Prediction and clustering in signed networks: a local to global perspective. *The Journal of Machine Learning Research*, 15(1):1177–1213, 2014.

[13] William W Cohen and Jacob Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 475–480. ACM, 2002.

[14] Erik D Demaine, Dotan Emanuel, Amos Fiat, and Nicole Immorlica. Correlation clustering in general weighted graphs. *Theoretical Computer Science*, 361(2-3):172–187, 2006.

[15] Devdatt Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[16] Lise Getoor and Ashwin Machanavajjhala. Entity resolution: theory, practice & open challenges. *Proceedings of the VLDB Endowment*, 5(12):2018–2019, 2012.

[17] Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109 – 137, 1983. ISSN 0378-8733. doi: https://doi.org/10.1016/0378-8733(83)90021-7. URL http://www.sciencedirect.com/science/article/pii/0378873383900217.

[18] Sungwoong Kim, Sebastian Nowozin, Pushmeet Kohli, and Chang D Yoo. Higher-order correlation clustering for image segmentation. In *Advances in neural information processing systems*, pages 1530–1538, 2011.

[19] Laurent Massoulié. Community detection thresholds and the weak ramanujan property. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 694–703. ACM, 2014.

[20] Arya Mazumdar and Barna Saha. Query complexity of clustering with side information. In *Advances in Neural Information Processing Systems 30*. 2017.

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

[22] Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *Combinatorica*, 38(3):665–708, 2018.

[23] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, New York, NY, USA, 2014. ISBN 1107057132, 9781107057135.

[24] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. A survey of signed network mining in social media. *ACM Computing Surveys (CSUR)*, 49(3):42, 2016.

[25] Anthony Wirth. Correlation clustering. In Claude Sammut and Geoffrey I Webb, editors, *Encyclopedia of machine learning and data mining*, pages 227–231. Springer US, 2010.

A Probability bounds

We give Chernoff-type probability bounds can be found in e.g. [15] and that we repeatedly use in our proofs. Let $X_1, \dots, X_n$ be binary random variables. We say that $X_1, \dots, X_n$ are non-positively correlated if for all $I \subseteq \{1, \dots, n\}$ we have:

$$\mathbb{P}[\forall i \in I : X_i = 0] \leq \prod_{i \in I} \mathbb{P}[X_i = 0] \quad \text{and} \quad \mathbb{P}[\forall i \in I : X_i = 1] \leq \prod_{i \in I} \mathbb{P}[X_i = 1] \tag{1}$$

The following holds:

Lemma 1. *Let $X_1, \dots, X_n$ be independent or, more generally, non-positively correlated binary random variables. Let $a_1, \dots, a_n \in [0, 1]$ and $X = \sum_{i=1}^{n} a_i X_i$. Then, for any $\delta > 0$, we have:*

$$\mathbb{P}[X < (1 - \delta)\mathbb{E}[X]] < e^{-\frac{\delta^2}{2}\mathbb{E}[X]} \tag{2}$$

$$\mathbb{P}[X > (1 + \delta)\mathbb{E}[X]] < e^{-\frac{\delta^2}{2+\delta}\mathbb{E}[X]} \tag{3}$$

B Supplementary Material for Section 3

B.1 Pseudocode of ACC

Algorithm 2 Invoked as $\mathrm{ACC}(V_1, 1)$ where $V_1 \equiv V$ and $r = 1$ is the index of the recursive call.

Parameters: Query rate function $f : \mathbb{N} \to \mathbb{N}$.

1: **if** $|V_r| = 0 \vee r > f(|V_1| - 1)$ **then** RETURN
2: Draw pivot π_r u.a.r. from V_r
3: $C_r \leftarrow \{\pi_r\}$ ▷ Create new cluster and add the pivot to it
4: Draw a random subset S_r of $f(|V_r| - 1)$ nodes from $V_r \setminus \{\pi_r\}$
5: **for** each $u \in S_r$ **do** query $\sigma(\pi_r, u)$
6: **if** $\exists\, u \in S_r$ such that $\sigma(\pi_r, u) = +1$ **then** ▷ Check if there is at least an edge
7: Query all remaining pairs (π_r, u) for $u \in V_r \setminus \big(\{\pi_r\} \cup S_r\big)$
8: $C_r \leftarrow C_r \cup \{u : \sigma(\pi_r, u) = +1\}$ ▷ Populate cluster based on queries
9: Output cluster C_r
10: $\mathrm{ACC}(V_r \setminus C_r, r + 1)$ ▷ Recursive call on the remaining nodes

B.2 Proof of Theorem 1

We refer to the pseudocode of ACC (Algorithm 2). We use V_r to denote the set of remaining nodes at the beginning of the r-th recursive call. Hence $V_1 = V$. If the condition in the **if** statement on line 6 is not true, then C_r is a singleton cluster. We denote by V_{sing} the set nodes that are output as singleton clusters.

Let Γ_A be the set of mistaken edges for the clustering output by ACC and let $\Delta_A = \big|\Gamma_A\big|$ be the cost of this clustering. Note that, in any recursive call, ACC misclassifies an edge $e = \{u, w\}$ if and only if e is part of a bad triangle whose third node v is chosen as pivot and does not become a singleton cluster, or if $\sigma(e) = +1$ and at least one of u, w becomes a singleton cluster. More formally, ACC misclassifies an edge $e = \{u, w\}$ if and only if one of the following three disjoint events holds:

$B_1(e)$: There exists $r \leq f(n-1)$ and a bad triangle $T \equiv \{u, v, w\} \subseteq V_r$ such that $\pi_r = v$ and $v \notin V_{\mathrm{sing}}$.
$B_2(e)$: There exists $r \leq f(n-1)$ such that $u, w \in V_r$ with $\sigma(u, w) = +1$ and $\pi_r \in \{u, w\} \cap V_{\mathrm{sing}}$.
$B_3(e)$: The algorithm stops after $f(n-1)$ calls without removing neither u nor w, and $\sigma(u, w) = +1$.

Therefore the indicator variable for the event "e is mistaken" is:

$$\mathbb{I}\{e \in \Gamma_A\} = \mathbb{I}\{B_1(e)\} + \mathbb{I}\{B_2(e)\} + \mathbb{I}\{B_3(e)\}$$

The expected cost of the clustering is therefore:

$$\mathbb{E}[\Delta_A] = \sum_{e\in\mathcal{E}} \mathbb{P}(B_1(e)) + \sum_{e\in\mathcal{E}} \mathbb{P}(B_2(e)) + \sum_{e\in\mathcal{E}} \mathbb{P}(B_3(e)) \tag{4}$$

We proceed to bound the three terms separately.

Bounding $\sum_{e\in\mathcal{E}} \mathbb{P}(B_1(e))$. Fix an arbitrary edge $e = \{u, w\}$. Note that, if $B_1(e)$ occurs, then T is unique, i.e. exactly one bad triangle T in V satisfies the definition of $B_1(e)$. Each occurrence of $B_1(e)$ can thus be charged to a single bad triangle T. We may thus write

$$\begin{aligned}
\sum_{e\in\mathcal{E}} \mathbb{I}\{B_1(e)\} &= \sum_{e\in\mathcal{E}} \mathbb{I}\{(\exists\, r)(\exists\, T\in\mathcal{T}) \,:\, T\subseteq V_r \,\wedge\, e\subset T \,\wedge\, \pi_r\in T\setminus e \,\wedge\, \pi_r\notin V_{\text{sing}}\} \\
&= \sum_{T\in\mathcal{T}} \mathbb{I}\{(\exists\, r) \,:\, T\subseteq V_r \,\wedge\, \pi_r\in T \,\wedge\, \pi_r\notin V_{\text{sing}}\} \\
&\le \sum_{T\in\mathcal{T}} \mathbb{I}\{A_T\}
\end{aligned}$$

where $A_T \equiv \big\{(\exists\, r) \,:\, T\subseteq V_r \,\wedge\, \pi_r\in T\big\}$. Let us then bound $\sum_{T\in\mathcal{T}} \mathbb{P}(A_T)$. Let $\mathcal{T}(e) \equiv \{T'\in\mathcal{T} \,:\, e\in T'\}$. We use the following fact extracted from the proof of [2, Theorem 6.1]. If $\{\beta_T \ge 0 \,:\, T\in\mathcal{T}\}$ is a set of weights on the bad triangles such that $\sum_{T\in\mathcal{T}(e)} \beta_T \le 1$ for all $e\in\mathcal{E}$, then $\sum_{T\in\mathcal{T}} \beta_T \le \text{OPT}$. Given $e\in\mathcal{E}$ and $T\in\mathcal{T}$, let $F_T(e)$ be the event corresponding to T being the first triangle in the set $\mathcal{T}(e)$ such that $T\in V_r$ and $\pi_r\in T\setminus e$ for some r. Now if $F_T(e)$ holds then A_T holds and no other $A_{T'}$ for $T'\in\mathcal{T}(e)\setminus\{T\}$ holds. Therefore

$$\sum_{T\in\mathcal{T}(e)} \mathbb{I}\{A_T \,\wedge\, F_T(e)\} = 1\,.$$

If A_T holds for some r_0, then it cannot hold for any other $r > r_0$ because $\pi_{r_0}\in T$ implies that for all $r > r_0$ we have $\pi_{r_0}\notin V_r$ implying $T\not\subseteq V_r$. Hence, given that A_T holds for r_0, if $F_T(e)$ holds too, then it holds for the same r_0 by construction. This implies that $\mathbb{P}\big(F_T(e)\mid A_T\big) = \frac{1}{3}$ because ACC chooses the pivot u.a.r. from the nodes in V_{r_0}. Thus, for each $e\in E$ we can write

$$1 = \sum_{T\in\mathcal{T}(e)} \mathbb{P}\big(A_T \,\wedge\, F_T(e)\big) = \sum_{T\in\mathcal{T}(e)} \mathbb{P}\big(F_T(e)\mid A_T\big)\mathbb{P}(A_T) = \sum_{T\in\mathcal{T}(e)} \frac{1}{3}\mathbb{P}(A_T)\,. \tag{5}$$

Choosing $\beta_T = \frac{1}{3}\mathbb{P}(A_T)$ we get $\sum_{T\in\mathcal{T}} \mathbb{P}(A_T) \le 3\text{OPT}$.

In the proof of KwikCluster, the condition $\sum_{T\in\mathcal{T}(e)} \beta_T \le 1$ was ensured by considering events $G_T(e) = A_T \,\wedge\, e\in\Gamma_A$. Indeed, in KwikCluster the events $\{G_T(e) \,:\, T\in\mathcal{T}(e)\}$ are disjoint, because $G_T(e)$ holds iff T is the first and only triangle in $\mathcal{T}(e)$ whose node opposite to e is chosen as pivot. For ACC this is not true because a pivot can become a singleton cluster, which does not cause $e\in\Gamma_A$ necessarily to hold.

Bounding $\sum_{e\in\mathcal{E}} \mathbb{P}(B_2(e))$. For any $u\in V_r$, let $d_r^+(u) = \big|\{v\in V_r \,:\, \sigma(u,v) = +1\}\big|$. We have:

$$\sum_{e\in\mathcal{E}} \mathbb{I}\{B_2(e)\} = \frac{1}{2}\sum_{u\in V} \sum_{r=1}^{f(n-1)} \mathbb{I}\{\pi_r = u \,\wedge\, \pi_r\in V_{\text{sing}}\}\, d_r^+(u)\,.$$

Taking expectations with respect to the randomization of ACC,

$$\begin{aligned}
\sum_{e\in\mathcal{E}} \mathbb{P}\big(B_2(e)\big) &= \frac{1}{2}\sum_{u\in V} \sum_{r=1}^{f(n-1)} \mathbb{E}\Big[\mathbb{I}\{\pi_r = u \,\wedge\, \pi_r\in V_{\text{sing}}\}\, d_r^+(u)\Big] \\
&= \frac{1}{2}\sum_{u\in V} \sum_{r=1}^{f(n-1)} \mathbb{E}\Big[\mathbb{I}\{\pi_r\in V_{\text{sing}}\}\, d_r^+(u)\,\Big|\,\pi_r = u\Big]\mathbb{P}(\pi_r = u)
\end{aligned}$$

For any round r, let H_{r-1} be the sequence of random draws made by the algorithm before round r. Then $\mathbb{P}\big(\pi_r \in V_{\text{sing}} \,\big|\, \pi_r = u,\, H_{r-1}\big) d_r^+(u) = 0$ if either $d_r^+(u) = 0$, or $d_r^+(u) \geq 1$ and $d_r^-(u) < f(n_r)$. Otherwise,

$$\mathbb{P}\big(\pi_r \in V_{\text{sing}} \,\big|\, \pi_r = u,\, H_{r-1}\big) = \prod_{j=0}^{f(n_r)-1} \frac{d_r^-(u) - j}{n_r - j} \leq \left(\frac{d_r^-(u)}{n_r}\right)^{f(n_r)} = \left(1 - \frac{d_r^+(u)}{n_r}\right)^{f(n_r)} \tag{6}$$

where the inequality holds because $d_r^-(u) \leq n_r$. Therefore, when $d_r^+(u) \geq 1$ and $d_r^-(u) \geq f(n_r)$,

$$\begin{aligned}
\mathbb{E}\Big[\mathbb{I}\left\{\pi_r \in V_{\text{sing}}\right\} d_r^+(u) \,\Big|\, \pi_r = u,\, H_{r-1}\Big] &= \mathbb{P}\big(\pi_r \in V_{\text{sing}} \,\big|\, \pi_r = u,\, H_{r-1}\big) d_r^+(u) \\
&= \left(1 - \frac{d_r^+(u)}{n_r}\right)^{f(n_r)} d_r^+(u) \\
&= \left(1 - \frac{d_r^+(u)}{n_r}\right)^{f(n_r)} d_r^+(u) \\
&\leq \exp\left(-\frac{d_r^+(u) f(n_r)}{n_r}\right) d_r^+(u) \\
&\leq \max_{z>0} \exp\left(-\frac{z\, f(n_r)}{n_r}\right) z \\
&\leq \frac{n_r}{e f(n_r)} \,.
\end{aligned}$$

Combining with the above, this implies

$$\sum_{e \in \mathcal{E}} \mathbb{P}\big(B_2(e)\big) \leq \frac{1}{2e} \sum_{r=1}^{f(n-1)} \mathbb{E}\left[\frac{n_r}{f(n_r)}\right] \leq \frac{1}{2e} \sum_{r=1}^{f(n-1)} \frac{n}{f(n)} < \frac{n}{2e}$$

where we used the facts that $n_r \leq n$ and the properties of f.

Bounding $\sum_{e \in \mathcal{E}} \mathbb{P}(B_3(e))$. Let V_{fin} be the remaining vertices in V_r after the algorithm stops and assume $|V_{\text{fin}}| > 1$ (so that there is at least a query left). Let $n_{\text{fin}} = |V_{\text{fin}}| - 1$ and, for any $u \in V_{\text{fin}}$, let $d_{\text{fin}}^+(u) = \big|\left\{v \in V_{\text{fin}} \,:\, \sigma(u,v) = +1\right\}\big|$. In what follows, we conventionally assume $V_r \equiv V_{\text{fin}}$ for any $r > f(n-1)$, and similarly for n_{fin} and d_{fin}^+. We have

$$\sum_{e \in \mathcal{E}} \mathbb{I}\left\{B_3(e)\right\} = \frac{1}{2} \sum_{u \in V_{\text{fin}}} d_{\text{fin}}^+(u) \leq \frac{1}{2} \left(\sum_{u \in V_{\text{fin}}} \frac{n_{\text{fin}}}{f(n_{\text{fin}})} + \sum_{u \in V_{\text{fin}}} \mathbb{I}\left\{ d_{\text{fin}}^+(u) > \frac{n_{\text{fin}}}{f(n_{\text{fin}})} \right\} d_{\text{fin}}^+(u) \right).$$

Fix some $r \leq f(n-1)$. Given any vertex $v \in V_r$ with $d_r^+(v) \geq \frac{n_r}{f(n_r)}$, let $E_r(v)$ be the event when at round r, ACC queries $\sigma(v,u)$ for all $u \in V_r \setminus \{v\}$. Introduce the notation $S_r = \sum_{u \in V_r} \mathbb{I}\left\{ d_r^+(u) > \frac{n_r}{f(n_r)} \right\} d_r^+(u)$ with $S_r = S_{\text{fin}}$ for all $r > f(n)$, and let $\delta_r = n_r - n_{r+1}$ be the number of nodes that are removed from V_r at the end of the r-th recursive call. Then

$$\delta_r \geq \mathbb{I}\left\{E_r(\pi_r)\right\} d_r^+(\pi_r) \geq \mathbb{I}\left\{ d_r^+(\pi_r) > \frac{n_r}{f(n_r)} \right\} \mathbb{I}\left\{E_r(\pi_r)\right\} d_r^+(\pi_r)$$

and

$$\mathbb{E}[\delta_r \mid H_{r-1}] \geq \sum_{v \in V_r} \mathbb{I}\left\{ d_r^+(v) > \frac{n_r}{f(n_r)} \right\} \mathbb{P}\big(E_r(v) \mid \pi_r = v,\, H_{r-1}\big) \mathbb{P}(\pi_r = v \mid H_{r-1}) d_r^+(v) \,.$$

Using the same argument as the one we used to bound (6),

$$\mathbb{P}\big(E_r(v) \mid \pi_r = v,\, H_{r-1}\big) \geq 1 - \left(1 - \frac{d_r^+(v)}{n_r}\right)^{f(n_r)} \geq 1 - \left(1 - \frac{1}{f(n_r)}\right)^{f(n_r)} \geq 1 - \frac{1}{e}$$

and $\mathbb{P}(\pi_r = v \mid H_{r-1}) = \frac{1}{n_r+1}$ for any $v \in V_r$, we may write

$$\mathbb{E}[\delta_r \mid H_{r-1}] \geq \left(1 - \frac{1}{e}\right) \frac{\mathbb{E}[S_r \mid H_{r-1}]}{n_r + 1} \geq \left(1 - \frac{1}{e}\right) \frac{\mathbb{E}[S_r \mid H_{r-1}]}{n} \,.$$

Observe now that $\sum_{r=1}^{f(n-1)} \delta_r \leq n_1 - n_{\text{fin}} \leq n-1$ and S_r is monotonically nonincreasing in r. Thus

$$n-1 \geq \sum_{r=1}^{f(n-1)} \mathbb{E}[\delta_r] \geq \frac{1}{n}\left(1-\frac{1}{e}\right)\sum_{r=1}^{f(n)} \mathbb{E}[S_r] \geq \frac{f(n-1)}{n}\left(1-\frac{1}{e}\right)\mathbb{E}[S_{\text{fin}}]$$

which implies $\mathbb{E}[S_{\text{fin}}] \leq \left(\frac{e}{e-1}\right)\frac{n(n-1)}{f(n-1)} \leq \left(\frac{e}{e-1}\right)\frac{n^2}{f(n)}$. So we have

$$\sum_{e\in\mathcal{E}} \mathbb{P}\big(B_3(e)\big) \leq \frac{1}{2}\left(\sum_{u\in V_{\text{fin}}} \mathbb{E}\left[\frac{n_{\text{fin}}}{f(n_{\text{fin}})}\right] + \mathbb{E}[S_{\text{fin}}]\right) \leq \frac{1}{2}\left(\frac{n^2}{f(n)} + \frac{e}{e-1}\frac{n^2}{f(n)}\right)$$

as claimed.

Bounding the number of queries. In round r, ACC asks $n_r \leq n$ queries if $\pi_r \notin V_{\text{sing}}$ and $f(n_r) \leq f(n)$ queries otherwise. Since the number of rounds is at most $f(n)$, the overall number of queries is at most $\max\big\{n, f(n)\big\}f(n) \leq nf(n)$.

KwikCluster as special case. When $f(r) = r$ for all r, ACC issues all queries $\sigma(\pi_r, u)$ for $u \in V_r \setminus \{\pi_r\}$ in each round r, and builds a cluster just like KwikCluster would. At the end of $f(n) = n = |V| - 1$ rounds, there can be at most a single node left, which is then declared a singleton cluster. Hence, ACC and KwikCluster behaves identically for any sequence of pivot draws. Moreover, it is easy to check that the events $B_2(e)$ and $B_3(e)$ can never occur when $f(n) = n$. Therefore, the only contribution to Δ_A is $\sum_{T\in\mathcal{T}} A_T$ which is bounded by 3OPT for any choice of f.

B.3 Pseudocode of ACCESS

Algorithm 3 Invoked as ACCESS$(V_1, 1)$ where $V_1 \equiv V$ and $r = 1$ is the index of the recursive call.

Parameters: Query rate function $f : \mathbb{N} \to \mathbb{N}$.

Sample the labels of $\binom{|V_r|}{2} f(n)/n^2$ edges chosen u.a.r. from $\binom{V_r}{2}$
if no label is positive **then**
    STOP and declare every $v \in V_r$ as singleton
Draw pivot π_r u.a.r. from V_r
$C_r \leftarrow \{\pi_r\}$ ▷ Create new cluster and add the pivot to it
Draw a random subset S_r of $f(|V_r| - 1)$ nodes from $V_r \setminus \{\pi_r\}$
for each $u \in S_r$ **do** query $\sigma(\pi_r, u)$
if $\exists\, u \in S_r$ such that $\sigma(\pi_r, u) = +1$ **then** ▷ Check if there is at least an edge
    Query all remaining pairs (π_r, u) for $u \in V_r \setminus \big(\{\pi_r\} \cup S_r\big)$
    $C_r \leftarrow C_r \cup \{u : \sigma(\pi_r, u) = +1\}$ ▷ Populate cluster based on queries
Output cluster C_r
ACCESS$(V_r \setminus C_r, r + 1)$ ▷ Recursive call on the remaining nodes

B.4 Proof of Theorem 2

We refer to the pseudocode of ACCESS (Algorithm 3).

Let G_r be the residual graph at round r. The total cost of the clustering produced by ACCESS is clearly bounded by the sum of the cost of ACC without round restriction, plus the number of edges in the residual graph G_r if r is the round at which ACCESS stops. The first term is, in expectation, at most $3\text{OPT} + n/2e$, as one can easily derive from the proof of Theorem 1. For the second term note that, if G_r contains k edges, then the probability that ACCESS stops is at most:

$$\left(1 - \frac{k}{\binom{|V_r|}{2}}\right)^{\binom{|V_r|}{2} f(n)/n^2} \leq e^{-kf(n)/n^2}$$

Thus the expected number of edges in the residual graph when ACCESS returns is bounded from above by $\max_{k\geq 1}(ke^{-kf(n)/n^2}) \leq \frac{n^2}{ef(n)}$.

Let us then move to the bound on the number of queries. The queries performed at line 1 are deterministically at most $nf(n)$. Concerning the other queries (line 7 and line 9), we divide the algorithm in two phases: the "dense" rounds r where G_r still contains at least $n^2/2f(n)$ edges, and the remaining "sparse" rounds where G_r contains less than $n^2/2f(n)$ edges.

Consider first a "dense" round r. We see G_r as an arbitrary fixed graph: for all random variables mentioned below, the distribution is thought solely as a function of the choices of the algorithm in the current round (i.e., the pivot node π_r and the queried edges). Now, let Q_r be the number of queries performed at lines 7 and 9), and $R_r = |V_r| - |V_{r+1}|$ be the number of nodes removed. Let π_r be the pivot, and let D_r be its degree in G_r. Let X_r be the indicator random variable of the event that $\sigma(\pi_r, u) = +1$ for some $u \in S_r$. Observe that:

$$Q_r \leq f(|V_r| - 1) + X_r(|V_r| - 1) \qquad \text{and} \qquad R_r = 1 + X_r\, D_r$$

Thus $\mathbb{E}[Q_r] \leq f(|V_r|-1)+\mathbb{E}[X_r]|V_r|$, while $\mathbb{E}[R_r] = 1+\mathbb{E}[X_r D_r]$. However, X_r is monotonically increasing in D_r, so $\mathbb{E}[X_r D_r] = \mathbb{E}[X_r]\mathbb{E}[D_r] + \mathrm{Cov}(X_r, D_r) \geq \mathbb{E}[X_r]\mathbb{E}[D_r]$. Moreover, by hypothesis $\mathbb{E}[D_r] \geq 2\big(n^2/2f(n)\big)/|V_r| \geq n/f(n)$. Thus:

$$\mathbb{E}[R_r] \geq 1 + \mathbb{E}[X_r]\mathbb{E}[D_r] \geq 1 + \mathbb{E}[X_r]\frac{n}{f(n)} \geq 1 + \mathbb{E}[X_r]\frac{|V_r|}{f(|V_r|)} \geq \frac{\mathbb{E}[Q_r]}{f(|V_r|)} \geq \frac{\mathbb{E}[Q_r]}{f(n)}$$

But then, since obviously $\sum_r R_r \leq n$:

$$\mathbb{E}\left[\sum_{r \text{ dense}} Q_r\right] \leq f(n)\mathbb{E}\left[\sum_{r \text{ dense}} R_r\right] \leq nf(n)$$

Consider now the "sparse" rounds, where G_r contains less than $n^2/2f(n)$ edges. With probability at least $1/2$ ACCESS finds no edge and thus stops right after lines 1–2. Hence the number of sparse rounds completed by ACCESS is at most one in expectation; the corresponding expected number of queries is then at most n.

B.5 Proof of Theorem 3

First of all, note that if the residual graph G_r contains $\mathcal{O}(n^2/f(n))$ edges, from r onwards ACCESS stops at each round independently with constant probability. The expected number of queries performed before stopping is therefore $\mathcal{O}(n)$, and the expected error incurred is obviously at most $\mathcal{O}(n^2/f(n))$.

We shall then bound the expected number of queries required before the residual graph contains $\mathcal{O}(n^2/f(n))$ edges. In fact, by definition of i', if ACCESS removes $C_{i'}, \ldots, C_\ell$, then the residual graph contains $\mathcal{O}(n^2/f(n))$ edges. We therefore bound the expected number of queries before $C_{i'}, \ldots, C_\ell$ are removed.

First of all recall that, when pivoting on a cluster of size c, the probability that the cluster is *not* removed is at most $e^{-cf(n)/n}$. Thus the probability that the cluster is not removed after $\Omega(c)$ of its nodes have been used as pivot is $e^{-\Omega(c^2)f(n)/n}$. Hence the probability that *any* of $C_{i'}, \ldots, C_\ell$ is not removed after $\Omega(c)$ of its nodes are used as pivot is, setting $c = \Omega\big(h(n)\big)$ and using a union bound, at most $p = ne^{-\Omega(h(n)^2)f(n)/n}$. Observe that $h(n) = \Omega\big(n/f(n)\big)$, for otherwise $\sum_{j=1}^{i'} \binom{C_j}{2} = o\big(n^2/f(n)\big)$, a contradiction. Therefore $p \leq ne^{-\Omega(h(n))}$. Note also that we can assume $h(n) = \omega(\ln n)$, else the theorem bound is trivially $O(n^2)$. This gives $p = \mathcal{O}\big(ne^{-\omega(\ln n)}\big) = o\big(1/\operatorname{poly}(n)\big)$. We can thus condition on the events that, at any point along the algorithm, every cluster among $C_{i'}, \ldots, C_\ell$ that is still in the residual graph has size $\Omega\big(h(n)\big)$; the probability of any other event changes by an additive $\mathcal{O}(p)$, which can be ignored.

Let now $k = \ell - i' + 1$, and suppose at a generic point $k' \leq k$ of the clusters $C_{i'}, \ldots, C_\ell$ are in the residual graph. Their total size is therefore $\Omega\big(k'h(n)\big)$. Therefore $\mathcal{O}\big(n/k'h(n)\big)$ rounds in expectation are needed for the pivot to fall among those clusters. Each time this happens, with probability $1 - e^{-\Omega(h(n))f(n)/n} = \Omega(1)$ the cluster containing the pivot is removed. Hence, in expectation a new cluster among $C_{i'}, \ldots, C_\ell$ is removed after $\mathcal{O}\big(n/k'h(n)\big)$ rounds. By summing

over all values of k', the number of expected rounds to remove all of $C_{i'}, \dots, C_\ell$ is

$$\mathcal{O}\left(\sum_{k'=1}^{k} \frac{n}{k'h(n)}\right) = \mathcal{O}\big(n(\ln n)/h(n)\big)$$

Since each round involves $\mathcal{O}(n)$ queries, the bound follows.

C Supplementary Material for Section 4

C.1 Proof of Theorem 4

Fix any C that is $(1-\varepsilon)$-knit. We show that ACC outputs a $\widehat{C}$ such that

$$\mathbb{E}\big[|\widehat{C} \cap C|\big] \geq \max\left\{\left(1 - \frac{5}{2}\varepsilon\right)|C| - 2\frac{n}{f(n)}, \left(\frac{f(n)}{n} - \frac{5}{2}\varepsilon\right)|C|\right\} \text{ and } \mathbb{E}\big[|\widehat{C} \cap \overline{C}|\big] \leq \frac{\varepsilon}{2}|C| \quad (7)$$

One can check that these two conditions together imply the first two terms in the bound. We start by deriving a lower bound on $\mathbb{E}\big[|\widehat{C} \cap C|\big]$ for KwikCluster assuming $|E_C| = \binom{|C|}{2}$. Along the way we introduce most of the technical machinery. We then port the bound to ACC, relax the assumption to $|E_C| \geq (1-\varepsilon)\binom{|C|}{2}$, and bound $\mathbb{E}\big[|\widehat{C} \cap \overline{C}|\big]$ from above. Finally, we add the $|C|e^{-|C|f(n)/5n}$ part of the bound. To lighten the notation, from now on C denotes both the cluster and its cardinality $|C|$.

For the sake of analysis, we see KwikCluster as the following equivalent process. First, we draw a random permutation π of V. This is the ordered sequence of *candidate pivots*. Then, we set $G_1 = G$, and for each $i = 1, \dots, n$ we proceed as follows. If $\pi_i \in G_i$, then π_i is used as an actual pivot; in this case we let $G_{i+1} = G_i \setminus (\pi_i \cup \mathcal{N}_{\pi_i})$ where $\mathcal{N}_v$ is the set of neighbors of v. If instead $\pi_i \notin G_i$, then we let $G_{i+1} = G_i$. Hence, G_i is the residual graph just before the i-th candidate pivot π_i is processed. We indicate the event $\pi_i \in G_i$ by the random variable P_i:

$$P_i = \mathbb{I}\{\pi_i \in G_i\} = \mathbb{I}\{\pi_i \text{ is used as pivot}\} \quad (8)$$

More in general, we define a random variable indicating whether node v is "alive" in G_i:

$$X(v,i) = \mathbb{I}\{v \in G_i\} = \mathbb{I}\big\{v \notin \cup_{j<i\,:\,P_j=1}(\pi_j \cup \mathcal{N}_{\pi_j})\big\} \quad (9)$$

Let $i_C = \min\{i : \pi_i \in C\}$ be the index of the first candidate pivot of C. Define the random variable:

$$S_C = |C \cap G_{i_C}| = \sum_{v \in C} X(v, i_C) \quad (10)$$

In words, S_C counts the nodes of C still alive in G_{i_C}. Now consider the following random variable:

$$S = P_{i_C} \cdot S_C \quad (11)$$

Let $\widehat{C}$ be the cluster that contains π_{i_C} in the output of KwikCluster. It is easy to see that $|C \cap \widehat{C}| \geq S$. Indeed, if $P_{i_C} = 1$ then $\widehat{C}$ includes $C \cap G_{i_C}$, so $|C \cap \widehat{C}| \geq P_{i_C}S_C = S$. If instead $P_{i_C} = 0$, then $S = 0$ and obviously $|C \cap \widehat{C}| \geq 0$. Hence in any case $|C \cap \widehat{C}| \geq S$, and $\mathbb{E}\big[|C \cap \widehat{C}|\big] \geq \mathbb{E}[S]$. Therefore we can bound $\mathbb{E}\big[|C \cap \widehat{C}|\big]$ from below by bounding $\mathbb{E}[S]$ from below.

Before continuing, we simplify the analysis by assuming KwikCluster runs on the graph G after all edges not incident on C have been deleted. We can easily show that this does not increase S. First, by (9) each $X(v, i_C)$ is a nonincreasing function of $\{P_i : i < i_C\}$. Second, by (10) and (11), S is a nondecreasing function of $\{X(v, i_C) : v \in C\}$. Hence, S is a nonincreasing function of $\{P_i : i < i_C\}$. Now, the edge deletion forces $P_i = 1$ for all $i < i_C$, since any $\pi_i : i < i_C$ has no neighbor $\pi_j : j < i$. Thus the edge deletion does not increase S (and, obviously, $\mathbb{E}[S]$). We can then assume $G[V \setminus C]$ is an independent set. At this point, any node not adjacent to C is isolated and can be ignored. We can thus restrict the analysis to C and its neighborhood in G. Therefore we let $\overline{C} = \{v : \{u, v\} \in E,\, u \in C, v \notin C\}$ denote both the neighborhood and the complement of C.

We turn to bounding $\mathbb{E}[S]$. For now we assume $G[C]$ is a clique; we will then relax the assumption to $|E_C| \geq (1-\varepsilon)\binom{C}{2}$. Since by hypothesis $\mathrm{cut}(C, \overline{C}) < \varepsilon C^2$, the average degree of the nodes in $\overline{C}$ is less than $\varepsilon C^2/\overline{C}$. This is also a bound on the expected number of edges between C and a node drawn

u.a.r. from $\overline{C}$. But, for any given i, conditioned on $i_C - 1 = i$ the nodes $\pi_1, \ldots, \pi_{i_C-1}$ are indeed drawn u.a.r. from $\overline{C}$, and so have a total of at most $i\varepsilon C^2/\overline{C}$ edges towards C in expectation. Thus, over the distribution of π, the expected number of edges between C and $\pi_1, \ldots, \pi_{i_C-1}$ is at most:

$$\sum_{i=0}^{n} \frac{i\varepsilon C^2}{\overline{C}} \mathbb{P}(i_C - 1 = i) = \frac{\varepsilon C^2}{\overline{C}} \mathbb{E}[i_C - 1] = \frac{\varepsilon C^2}{\overline{C}} \frac{\overline{C}}{C+1} < \varepsilon C \tag{12}$$

where we used the fact that $\mathbb{E}[i_C - 1] = \overline{C}/(C+1)$. Now note that (12) is a bound on $C - \mathbb{E}[S_C]$, the expected number of nodes of C that are adjacent to $\pi_1, \ldots, \pi_{i_C-1}$. Therefore, $\mathbb{E}[S_C] \geq (1-\varepsilon)C$.

Recall that P_{i_C} indicates whether π_{i_C} is not adjacent to any of $\pi_1, \ldots, \pi_{i_C-1}$. Since the distribution of π_{i_C} is uniform over C, $\mathbb{P}(P_{i_C} \mid S_C) = S_C/C$. But $S = P_{i_C} S_C$, hence $\mathbb{E}[S \mid S_C] = (S_C)^2/C$, and thus $\mathbb{E}[S] = \mathbb{E}\big[(S_C)^2\big]/C$. Using $\mathbb{E}[S_C] \geq (1-\varepsilon)C$ and invoking Jensen's inequality we obtain

$$\mathbb{E}[S] \geq \frac{\mathbb{E}[S_C]^2}{C} \geq (1-\varepsilon)^2 C \geq (1-2\varepsilon)C \tag{13}$$

which is our bound on $\mathbb{E}\big[|C \cap \widehat{C}|\big]$ for KwikCluster.

Let us now move to ACC. We have to take into account the facts that ACC performs $f(|G_r| - 1)$ queries on the pivot before deciding whether to perform $|G_r| - 1$ queries, and that ACC stops after $f(n-1)$ rounds. We start by addressing the first issue, assuming for the moment ACC has no restriction on the number of rounds.

Recall that $\mathbb{P}(P_{i_C} \mid S_C) = S_C/C$. Now, if $P_{i_C} = 1$, then we have $S_C - 1$ edges incident on π_{i_C}. It is easy to check that the probability that ACC finds some of them is at least $1 - e^{-f(n)\frac{S_C-1}{n}}$ and, if this event occurs, then $S = S_C$. Thus

$$\mathbb{E}[S \mid S_C] = \mathbb{P}(P_{i_C} \mid S_C) S_C \geq \left(1 - e^{-f(n)\frac{S_C-1}{n}}\right) \frac{S_C^2}{C} \geq \frac{S_C^2}{C} - S_C \frac{2n}{f(n)C} \tag{14}$$

where we used the facts that for $S_C \leq 1$ the middle expression in (14) vanishes, that $e^{-x} < 1/x$ for $x > 0$, and that $1/x < 2/(x+1)$ for all $x \geq 2$. Simple manipulations, followed by Jensen's inequality and an application of $\mathbb{E}[S_C] \geq (1-\varepsilon)C$, give

$$\mathbb{E}[S] \geq (1-\varepsilon)^2 C - (1-\varepsilon)C \frac{2n}{f(n)C} \geq (1-2\varepsilon)C - 2\frac{n}{f(n)} \tag{15}$$

We next generalize the bound to the case $E_C \geq (1-\varepsilon)\binom{C}{2}$. To this end note that, since at most $\varepsilon\binom{C}{2}$ edges are missing from any subset of C, then any subset of S_C nodes of C has average degree at least

$$\max\left\{0, S_C - 1 - \binom{C}{2}\frac{2\varepsilon}{S_C}\right\} \geq S_C - \frac{\varepsilon C(C-1)}{2S_C} - 1 \tag{16}$$

We can thus re-write (14) as

$$\mathbb{E}[S \mid S_C] \geq \frac{S_C}{C}\left(1 - e^{-f(n)\frac{S_C-1}{n}}\right)\left(S_C - \frac{\varepsilon C(C-1)}{2S_C}\right) \tag{17}$$

Standard calculations show that this expression is bounded from below by $\frac{S_C^2}{C} - S_C\frac{2n}{f(n)C} - \frac{\varepsilon C}{2}$, which by calculations akin to the ones above leads to $\mathbb{E}[S] \geq (1 - \frac{5}{2}\varepsilon)C - 2\frac{n}{f(n)}$.

Similarly, we can show that $\mathbb{E}[S] \geq \big(\frac{f(n)}{n} - \frac{5}{2}\varepsilon\big)C$. To this end note that when ACC pivots on π_{i_C} all the remaining cluster nodes are found with probability at least $\frac{f(n)}{n}$ (this includes the cases $S_C \leq 1$, when such a probability is indeed 1). In (14), we can then replace $1 - e^{-f(n)\frac{S_C-1}{n}}$ with $\frac{f(n)}{n}$, which leads to $\mathbb{E}[S] \geq \big(\frac{f(n)}{n} - \frac{5}{2}\varepsilon\big)C$. This proves the first inequality in (7).

For the second inequality in (7), note that any subset of S_C nodes has $\mathrm{cut}(C, \overline{C}) \leq \varepsilon\binom{C}{2}$. Thus, π_{i_C} is be incident to at most $\frac{\varepsilon}{S_C}\binom{C}{2}$ such edges in expectation. The expected number of nodes of $\overline{C}$ that ACC assigns to $\widehat{C}$, as a function of S_C, can thus be bounded by $\frac{S_C}{C}\frac{\varepsilon}{S_C}\binom{C}{2} < \frac{\varepsilon}{2}C$.

As far as the $\mathcal{O}(Ce^{-Cf(n)/n})$ part of the bound is concerned, simply note that the bounds obtained so far hold unless $i_C > f(n-1)$, in which case ACC stops before ever reaching the first node of C. If this happens, $\widehat{C} = \{\pi_{i_C}\}$ and $|\widehat{C} \oplus C| < |C|$. The event $i_C > f(n-1)$ is the event that no node of C is drawn when sampling $f(n-1)$ nodes from V without replacement. We can therefore apply Chernoff-type bounds to the random variable X counting the number of draws of nodes of C and get $\mathbb{P}\big(X < (1-\beta)\mathbb{E}[X]\big) \leq \exp(-\beta^2\mathbb{E}[X]/2\big)$ for all $\beta > 0$. In our case $\mathbb{E}[X] = f(n-1)|C|/n$, and we have to bound the probability that X equals $0 < (1-\beta)\mathbb{E}[X]$. Thus

$$\mathbb{P}(X = 0) \leq \exp\left(-\frac{\beta^2\mathbb{E}[X]}{2}\right) = \exp\left(-\frac{\beta^2 f(n-1)|C|}{2n}\right)$$

Since $f(n-1) \geq f(n)/2$ (otherwise $n = 1$ and V is trivial), choosing, e.g., $\beta > \sqrt{4/5}$ yields $\mathbb{P}(X = 0) < \exp\big(-|C|f(n)/5n\big)$. This case therefore adds at most $|C|\exp(-|C|f(n)/5n)$ to $\mathbb{E}[|\widehat{C} \oplus C|]$.

C.2 Proof of Theorem 5

Before moving to the actual proof, we need some ancillary results. The next lemma bounds the probability that ACC does not pivot on a node of C in the first k rounds.

Lemma 2. *Fix a subset $C \subseteq V$ and an integer $k \geq 1$, and let $\pi_1, \ldots, \pi_n$ be a random permutation of V. For any $v \in C$ let $X_v = \mathbb{I}\{v \in \{\pi_1, \ldots, \pi_k\}\}$, and let $X_C = \sum_{v \in C} X_v$. Then $\mathbb{E}[X_C] = \frac{k|C|}{n}$, and $\mathbb{P}(X_C = 0) < e^{-\frac{k|C|}{3n}}$.*

Proof. Since π is a random permutation, then for each $v \in C$ and each each $i = 1, \ldots, k$ we have $\mathbb{P}(\pi_i = v) = \frac{1}{n}$. Therefore $\mathbb{E}[X_v] = \frac{k}{n}$ and $\mathbb{E}[X_C] = \frac{k|C|}{n}$. Now, the process is exactly equivalent to sampling without replacement from a set of n items of which $|C|$ are marked. Therefore, the X_v's are non-positively correlated and we can apply standard concentration bounds for the sum of independent binary random variables. In particular, for any $\eta \in (0,1)$ we have:

$$\mathbb{P}(X_C = 0) \leq \mathbb{P}(X_C < (1-\eta)\mathbb{E}[X_C]) < \exp\Big(-\frac{\eta^2\mathbb{E}[X_C]}{2}\Big)$$

which drops below $e^{-\frac{k|C|}{3n}}$ by replacing $\mathbb{E}[X_C]$ and choosing $\eta \geq \sqrt{2/3}$. □

The next lemma is the crucial one.

Lemma 3. *Let $\varepsilon \leq \frac{1}{10}$. Consider a strongly $(1-\varepsilon)$-knit set C with $|C| > \frac{10n}{f(n)}$. Let $u_C = \min\{v \in C\}$ be the id of C. Then, for any $v \in C$, in any single run of* ACC *we have $\mathbb{P}(\mathrm{id}(v) = u_C) \geq \frac{2}{3}$.*

Proof. We bound from above the probability that any of three "bad" events occurs. As in the proof of Theorem 4, we equivalently see ACC as going through a sequence of candidate pivots $\pi_1, \ldots, \pi_n$ that is a uniform random permutation of V. Let $i_C = \min\{i : \pi_i \in C\}$ be the index of the first node of C in the random permutation of candidate pivots. The first event, B_1, is $\{i_C > f(n-1)\}$. Note that, if B_1 does not occur, then ACC will pivot on π_{i_C}. The second event, B_2, is the event that $\pi_{i_C} \in V_{sing}$ if ACC pivots on π_{i_C} (we measure the probability of B_2 conditioned on $\overline{B_1}$). The third event, B_3, is $\{\pi_{i_C} \notin P\}$ where $P = \mathcal{N}_{u_C} \cap \mathcal{N}_v$. If none among B_1, B_2, B_3 occurs, then ACC forms a cluster $\widehat{C}$ containing both u_C and v, and by the min-tagging rule sets $\mathrm{id}(v) = \min_{u \in \widehat{C}} = u_C$. We shall then show that $\mathbb{P}(B_1 \cup B_2 \cup B_3) \leq 1/3$.

For B_1, we apply Lemma 2 by observing that $i_C > f(n-1)$ corresponds to the event $X_C = 0$ with $k = f(n-1)$. Thus

$$\mathbb{P}(i_C > f(n-1)) < e^{-\frac{f(n-1)|C|}{3n}} \leq e^{-\frac{f(n-1)}{3n}\frac{10\,n}{f(n)}} = e^{-\frac{f(n-1)}{f(n)}\frac{10}{3}} < e^{-3}$$

where we used the fact that $n \geq |C| \geq 11$ and therefore $f(n-1) \geq \frac{10}{11}f(n)$. For B_2, recall that by definition every $v \in C$ has at least $(1-\varepsilon)c$ edges. Thus, if ACC pivots on π_{i_C}, we have:

$$\mathbb{P}(\pi_{i_C} \in V_{sing}) \leq \exp\Big(-\frac{f(n-1)}{n-1}(1-\varepsilon)c\Big) \leq \exp\Big(-\frac{f(n-1)}{n-1}\big(1-\frac{1}{10}\big)\frac{10\,n}{f(n)}\Big) \leq e^{-9}$$

where we used the fact that $\frac{f(n-1)}{n-1}\frac{n}{f(n)} \geq 1$. For B_3, note that the distribution of π_{i_C} is uniform over C. Now, let $\mathcal{N}_{u_C}$ and $\mathcal{N}_v$ be the neighbor sets of u_C and v in C, and let $P = \mathcal{N}_{u_C} \cap \mathcal{N}_v$. We call P the set of good pivots. Since C is strongly $(1-\varepsilon)$-knit, both u_C and v have at least $(1-\varepsilon)c$ neighbors in C. But then $|C \setminus P| \leq 2\varepsilon c$ and

$$\mathbb{P}(\pi_{i_C} \notin P) = \frac{|C \setminus P|}{|C|} \leq 2\varepsilon \leq 1/5$$

By a union bound, then, $\mathbb{P}(B_1 \cup B_2 \cup B_3) \leq e^{-3} + e^{-9} + 1/5 < 1/3$. □

We are now ready to conclude the proof. Suppose we execute ACC independently $K = 48\lceil \ln(n/p) \rceil$ times with the min-tagging rule. For a fixed $v \in G$ let X_v be the number of executions giving $\mathrm{id}(v) = u_C$. On the one hand, by Lemma 3, $\mathbb{E}[X_v] \geq \frac{2}{3}K$. On the other hand, v will not be assigned to the cluster with id u_C by the majority voting rule only if $X_v \leq \frac{1}{2}K \leq \mathbb{E}[X_v](1-\delta)$ where $\delta = \frac{1}{4}$. By standard concentration bounds, then, $\mathbb{P}(X_v \leq \frac{1}{2}K) \leq \exp(-\frac{\delta^2 \mathbb{E}[X_v]}{2}) = \exp(-\frac{K}{48})$. By setting $K = 48\ln(n/p)$, the probability that v is not assigned id u_C is thus at most p/n. A union bound over all nodes concludes the proof.

D Supplementary Material for Section 6

D.1 Proof of Theorem 8

We prove that there exists a distribution over labelings σ with $\mathrm{OPT} = 0$ on which any deterministic algorithm has expected cost at least $\frac{n\varepsilon^2}{8}$. Yao's minimax principle then implies the claimed result.

Given $V = \{1, \ldots, n\}$, we define σ by a random partition of the vertices in $d \geq 2$ isolated cliques $T_1, \ldots, T_d$ such that $\sigma(v, v') = +1$ if and only if v and v' belong to the same clique. The cliques are formed by assigning each node $v \in V$ to a clique I_v drawn uniformly at random with replacement from $\{1, \ldots, d\}$, so that $T_i = \{v \in V \,:\, I_v = i\}$. Consider a deterministic algorithm making queries $\{s_t, r_t\} \in \mathcal{E}$. Let E_i be the event that the algorithm never queries a pair of nodes in T_i with $|T_i| \geq \frac{n}{2d} > 5$. Apply Lemma 4 below with $d = \frac{1}{\varepsilon}$. This implies that the expected number of non-queried clusters of size at least $\frac{n}{2d}$ is at least $\frac{d}{2} = \frac{1}{2\varepsilon}$. The overall expected cost of ignoring these clusters is therefore at least

$$\frac{d}{2}\left(\frac{n}{2d}\right)^2 = \frac{n^2}{8d} = \frac{\varepsilon n^2}{8}$$

and this concludes the proof.

Lemma 4. *Suppose $d > 0$ is even, $n \geq 16d\ln d$, and $B < \frac{d^2}{50}$. Then for any deterministic learning algorithm making at most B queries,*

$$\sum_{i=1}^{d} \mathbb{P}(E_i) > \frac{d}{2}\,.$$

Proof. For each query $\{s_t, r_t\}$ we define the set L_t of all cliques T_i such that $s_t \notin T_i$ and some edge containining both s_t and a node of T_i was previously queried. The set R_t is defined similarly using r_t. Formally,

$$\begin{aligned} L_t &= \{i \,:\, (\exists \tau < t)\; s_\tau = s_t \,\wedge\, r_\tau \in T_i \,\wedge\, \sigma(s_\tau, r_\tau) = -1\} \\ R_t &= \{i \,:\, (\exists \tau < t)\; r_\tau = r_t \,\wedge\, s_\tau \in T_i \,\wedge\, \sigma(s_\tau, r_\tau) = -1\}\,. \end{aligned}$$

Let D_t be the event that the t-th query discovers a new clique of size at least $\frac{n}{2d}$, and let $P_t = \max\{|L_t|, |R_t|\}$. Using this notation,

$$\sum_{t=1}^{B} \mathbb{I}\{D_t\} = \sum_{t=1}^{B} \mathbb{I}\{D_t \,\wedge\, P_t < d/2\} + \underbrace{\sum_{t=1}^{B} \mathbb{I}\{D_t \,\wedge\, P_t \geq d/2\}}_{N}\,. \tag{18}$$

We will now show that unless $B \geq \frac{d^2}{50}$, we can upper bound N deterministically by $\sqrt{2B}$.

Suppose $N > \frac{d}{2}$, and let $t_1, \ldots, t_N$ be the times t_k such that $\mathbb{I}\{D_{t_k} \wedge P_{t_k} \geq d/2\} = 1$. Now fix some k and note that, because the clique to which s_{t_k} and r_{t_k} both belong is discovered, neither s_{t_k} nor r_{t_k} can occur in a future query $\{s_t, r_t\}$) that discovers a new clique. Therefore, in order to have $\mathbb{I}\{D_t \wedge P_t \geq d/2\} = 1$ for $N > \frac{d}{2}$ times, at least

$$\binom{N}{2} \geq \frac{d^2}{8}$$

queries must be made, since each one of the other $N - 1 \geq \frac{d}{2}$ discovered cliques can contribute with at most a query to making $P_t \geq \frac{d}{2}$. So, it takes at least $B \geq \frac{d^2}{8}$ queries to discover the first $\frac{d}{2}$ cliques of size at least two, which contradicts the lemma's assumption that $B \leq \frac{d^2}{16}$. Therefore, $N \leq \frac{d}{2}$.

Using the same logic as before, in order to have $\mathbb{I}\{D_t \wedge P_t \geq d/2\} = 1$ for $N \leq \frac{d}{2}$ times, at least

$$\frac{d}{2} + \left(\frac{d}{2} - 1\right) + \cdots + \left(\frac{d}{2} - N + 1\right)$$

queries must be made. So, it must be

$$B \geq \sum_{k=1}^{N} \left(\frac{d}{2} - (k-1)\right) = (d+1)\frac{N}{2} - \frac{N^2}{2}$$

or, equivalently, $N^2 - (d+1)N + 2B \geq 0$. Solving this quadratic inequality for N, and using the hypothesis $N \leq \frac{d}{2}$, we have that $N \leq \frac{(d+1) - \sqrt{(d+1)^2 - 8B}}{2}$. Using the assumption that $B \leq \frac{d^2}{50}$ we get that $N \leq \sqrt{2B}$.

We now bound the first term of (18) in expectation. The event D_t is equivalent to $s_t, r_t \in T_i$ for some $i \in \neg L_t \cap \neg R_t$, where for any $S \subseteq \{1, \ldots, d\}$ we use $\neg S$ to denote $\{1, \ldots, d\} \setminus S$.

Let $\mathbb{P}_t = \mathbb{P}\big(\cdot \mid P_t < d/2\big)$. For L', R' ranging over all subsets of $\{1, \ldots, d\}$ of size strictly less than $\frac{d}{2}$,

$$\begin{aligned}
\mathbb{P}_t(D_t) &= \sum_{L',R'} \sum_{i \in \neg L' \cap \neg R'} \mathbb{P}_t\big(s_t \in T_i \wedge r_t \in T_i \,\big|\, L_t = L',\, R_t = R'\big)\, \mathbb{P}_t(L_t = L' \wedge R_t = R') \\
&= \sum_{L',R'} \sum_{i \in \neg L' \cap \neg R'} \mathbb{P}_t\big(s_t \in T_i \,\big|\, L_t = L'\big)\, \mathbb{P}_t\big(r_t \in T_i \,\big|\, R_t = R'\big)\, \mathbb{P}_t(L_t = L' \wedge R_t = R') \qquad (19) \\
&= \sum_{L',R'} \sum_{i \in \neg L' \cap \neg R'} \frac{1}{|\neg L'|}\, \frac{1}{|\neg R'|}\, \mathbb{P}_t(L_t = L' \wedge R_t = R') \qquad (20) \\
&= \sum_{L',R'} \frac{|\neg L' \cap \neg R'|}{|\neg L'|\,|\neg R'|}\, \mathbb{P}_t(L_t = L' \wedge R_t = R') \\
&\leq \frac{2}{d}\,. \qquad (21)
\end{aligned}$$

Equality (19) holds because $P_t = \max\{L_t, R_t\} < \frac{d}{2}$ implies that there are at least two remaining cliques to which s_t and r_t could belong, and each node is independently assigned to one of these cliques. Equality (20) holds because, by definition of L_t, the clique of s_t is not in L_t, and there were no previous queries involving s_t and a node belonging to a clique in $\neg L_t$ (similarly for r_t). Finally, (21) holds because $|\neg L'| \geq \frac{d}{2}$, $|\neg R'| \geq \frac{d}{2}$, and $|\neg L' \cap \neg R'| \leq \min\{|\neg L'|, |\neg R'|\}$. Therefore,

$$\sum_{t=1}^{B} \mathbb{P}\big(D_t \wedge P_t < d/2\big) \leq \sum_{t=1}^{B} \mathbb{P}\big(D_t \mid P_t < d/2\big) \leq \frac{2B}{d}\,.$$

Putting everything together,

$$\mathbb{E}\left[\sum_{t=1}^{B} \mathbb{I}\{D_t\}\right] \leq \frac{2B}{d} + \sqrt{2B}\,. \qquad (22)$$

On the other hand, we have

$$\sum_{t=1}^{B} \mathbb{I}\{D_t\} = \sum_{i=1}^{d} \Big(\mathbb{I}\big\{|T_i| \geq \tfrac{n}{2d}\big\} - \mathbb{I}\{E_i\} \Big) = d - \sum_{i=1}^{d} \Big(\mathbb{I}\big\{|T_i| < \tfrac{n}{2d}\big\} + \mathbb{I}\{E_i\} \Big) \tag{23}$$

Combining (22) and (23), we get that

$$\sum_{i=1}^{d} \mathbb{P}(E_i) \geq d - \sum_{i=1}^{d} \mathbb{P}\big(|T_i| < \tfrac{n}{2d}\big) - \frac{2B}{d} - \sqrt{2B}\,.$$

By Chernoff-Hoeffding bound, $\mathbb{P}\big(|T_i| < \frac{n}{2d}\big) \leq \frac{1}{d^2}$ for each $i = 1, \ldots, d$ when $n \geq 16d \ln d$. Therefore,

$$\sum_{i=1}^{d} \mathbb{P}(E_i) \geq d - \frac{2B+1}{d} - \sqrt{2B}\,.$$

To finish the proof, suppose on the contrary that $\sum_{i=1}^{d} \mathbb{P}(E_i) \leq \frac{d}{2}$. Then from the inequality above, we would get that

$$\frac{d}{2} \geq d - \frac{2B+1}{d} - \sqrt{2B}$$

which implies $B \geq \left(\frac{2-\sqrt{2}}{4}\right)^2 d^2 > \frac{d^2}{50}$, contradicting the assumptions. Therefore, we must have $\sum_{i=1}^{d} \mathbb{P}(E_i) > \frac{d}{2}$ as required. □

D.2 Proof of Theorem 9

Choose a suitably large n and let $V = [n]$. We partition V in two sets A and B, where $|A| = \alpha n$ and $|B| = (1-\alpha)n$; we will eventually set $\alpha = 0.9$, but for now we leave it free to have a clearer proof. The set A is itself partitioned into $k = 1/\varepsilon$ subsets $A_1, \ldots, A_k$, each one of equal size $\alpha n/k$ (the subsets are not empty because of the assumption on ε). The labeling σ is the distribution defined as follows. For each $i = 1, \ldots, k$, for each pair $u, v \in A_i$, $\sigma(u, v) = +1$; for each $u, v \in B$, $\sigma(u, v) = -1$. Finally, for each $v \in B$ we have a random variable i_v distributed uniformly over $[k]$. Then, $\sigma(u, v) = +1$ for all $u \in A_{i_v}$ and $\sigma(u, v) = -1$ for all $u \in A \setminus A_{i_v}$. Note that the distribution of i_v is independent of the (joint) distributions of the i_w's for all $w \in B \setminus \{v\}$.

Let us start by giving an upper bound on $\mathbb{E}[\mathrm{OPT}]$. To this end consider the (possibly suboptimal) clustering $\mathcal{C} = \{C_i : i \in [k]\}$ where $C_i = A_i \cup \{v \in B : i_v = i\}$. One can check that $\mathcal{C}$ is a partition of V. The expected cost $\mathbb{E}[\Delta_{\mathcal{C}}]$ of $\mathcal{C}$ can be bound as follows. First, note the only mistakes are due to pairs $u, v \in B$. However, for any such fixed pair u, v, the probability of a mistake (taken over σ) is $\mathbb{P}(i_u \neq i_v) = 1/k$. Thus,

$$\mathbb{E}[\mathrm{OPT}] \leq \mathbb{E}[\Delta_0] < \frac{|B|^2}{k} = \frac{(1-\alpha)^2 n^2}{k} \tag{24}$$

Let us now turn to the lower bound on the expected cost of the clustering produced by an algorithm. For each $v \in B$ let Q_v be the total number of distinct queries the algorithm makes to pairs $\{u, v\}$ with $u \in A$ and $v \in B$. Let Q be the total number of queries made by the algorithm; obviously, $Q \geq \sum_{v \in B} Q_v$. Now let S_v be the indicator variable of the event that one of the queries involving v returned $+1$. Both Q_v and S_v as random variables are a function of the input distribution and of the choices of the algorithm. The following is key:

$$\mathbb{P}(S_v \wedge Q_v < k/2) < \frac{1}{2} \tag{25}$$

The validity of (25) is seen by considering the distribution of the input limited to the pairs $\{u, v\}$. Indeed, $S_v \wedge Q_v < k/2$ implies the algorithm discovered the sole positive pair involving v in less than $k/2$ queries. Since there are k pairs involving v, and for any fixed j the probability (taken over the input) that the algorithm finds that particular pair on the j-th query is exactly $1/k$. Now,

$$\mathbb{P}(S_v \wedge Q_v < k/2) + \mathbb{P}(\overline{S_v} \wedge Q_v < k/2) + \mathbb{P}(Q_v \geq k/2) = 1 \tag{26}$$

and therefore

$$\mathbb{P}(\overline{S_v} \wedge Q_v < k/2) + \mathbb{P}(Q_v \geq k/2) > \frac{1}{2} \tag{27}$$

Let us now consider R_v, the number of mistakes involving v made by the algorithm. We analyse $\mathbb{E}[R_v \mid \overline{S_v} \wedge Q_v < k/2]$. For all $i \in [k]$ let Q_v^i indicate the event that, for some $u \in A_i$, the algorithm queried the pair $\{u, v\}$. Let $I = \{i \in [k] : Q_v^i = 0\}$; thus I contains all i such that the algorithm did not query any pair u, v with $u \in A_i$. Suppose now the event $\overline{S_v} \wedge Q_v < k/2$ occurs. On the one hand, $\overline{S_v}$ implies that:

$$\mathbb{P}(\sigma(u,v) = +1 \mid I) = \begin{cases} 1/|I| & u \in A_i, i \in I \\ 0 & u \in A_i, i \in [k] \setminus I \end{cases} \tag{28}$$

Informally speaking, this means that the random variable i_v is distributed uniformly over the (random) set I. Now observe that, again conditioning on the joint event $\overline{S_v} \wedge Q_v < k/2$, whatever label s the algorithm assigns to a pair u, v with $u \in A_i$ where $i \in I$, the distribution of $\sigma(u, v)$ is independent of s. This holds since s can obviously be a function only of I and of the queries made so far, all of which returned -1, and possibly of the algorithm's random bits. In particular, it follows that:

$$\mathbb{P}(\sigma(u,v) \neq s \mid I) \geq \min\left\{1/|I|, 1 - 1/|I|\right\} \tag{29}$$

However, $Q_v < k/2$ implies that $|I| \geq k - Q_v > k/2 = 2/\varepsilon > 2$, which implies $\min\{1/|I|, 1-1/|I|\} \geq 1/|I|$. Therefore, $\mathbb{P}(\sigma(u,v) \neq s \mid I) \geq 1/|I|$ for all $u \in A_i$ with $i \in I$.

We can now turn to back to R_v, the number of total mistakes involving v. Clearly, $R_v \geq \sum_{i=1}^{k} \sum_{u \in A_i} \mathbb{I}\{\sigma(u,v) \neq s\}$. Then:

$$\mathbb{E}[R_v \mid E] = \mathbb{E}\Big[\sum_{i=1}^{k} \sum_{u \in A_i} \mathbb{I}\{\sigma(u,v) \neq s\} \;\Big|\; \overline{S_v} \wedge Q_v < k/2\Big] \tag{30}$$

$$= \mathbb{E}\Big[\mathbb{E}\Big[\sum_{i=1}^{k} \sum_{u \in A_i} \mathbb{I}\{\sigma(u,v) \neq s\} \;\Big|\; I\Big] \;\Big|\; \overline{S_v} \wedge Q_v < k/2\Big] \tag{31}$$

$$\geq \mathbb{E}\Big[\mathbb{E}\Big[\sum_{i \in I} \sum_{u \in A_i} \mathbb{I}\{\sigma(u,v) \neq s\} \;\Big|\; I\Big] \;\Big|\; \overline{S_v} \wedge Q_v < k/2\Big] \tag{32}$$

$$\geq \mathbb{E}\Big[\mathbb{E}\Big[\sum_{i \in I} \sum_{u \in A_i} \frac{1}{|I|} \;\Big|\; I\Big] \;\Big|\; \overline{S_v} \wedge Q_v < k/2\Big] \tag{33}$$

$$= \mathbb{E}\Big[\mathbb{E}\Big[\frac{\alpha n}{k}\Big] \;\Big|\; \overline{S_v} \wedge Q_v < k/2\Big] \tag{34}$$

$$= \frac{\alpha n}{k} \tag{35}$$

And therefore:

$$\begin{aligned} \mathbb{E}[R_v] &\geq \mathbb{E}[R_v \mid \overline{S_v} \wedge Q_v < k/2] \cdot \mathbb{P}(\overline{S_v} \wedge Q_v < k/2) \\ &> \frac{\alpha n}{k} \cdot \mathbb{P}(\overline{S_v} \wedge Q_v < k/2) \end{aligned}$$

This concludes the bound on $\mathbb{E}[R_v]$. Let us turn to $\mathbb{E}[Q_v]$. Just note that:

$$\mathbb{E}[Q_v] \geq \frac{k}{2} \cdot \mathbb{P}(Q_v \geq k/2) \tag{36}$$

By summing over all nodes, we obtain:

$$\mathbb{E}[Q] \geq \sum_{v \in B} \mathbb{E}[Q_v] \geq \frac{k}{2}\Big(\sum_{v \in B} \mathbb{P}(Q_v \geq k/2)\Big) \tag{37}$$

$$\mathbb{E}[\Delta] \geq \sum_{v \in B} \mathbb{E}[R_v] > \frac{\alpha n}{k}\Big(\sum_{v \in B} \mathbb{P}(\overline{S_v} \wedge Q_v < k/2)\Big) \tag{38}$$

to which, by virtue of (27), applies the constraint:

$$\Big(\sum_{v\in B}\mathbb{P}(Q_v \geq k/2)\Big) + \Big(\sum_{v\in B}\mathbb{P}(\overline{S_v}\wedge Q_v < k/2)\Big) > |B|\frac{1}{2} = \frac{(1-\alpha)n}{2} \tag{39}$$

This constrained system gives the bound. Indeed, by (37), (38) and (39), it follows that if $\mathbb{E}[Q] < \frac{k}{2}\frac{(1-\alpha)n}{4} = \frac{(1-\alpha)nk}{8}$ then $\mathbb{E}[\Delta] > \frac{\alpha n}{k}\frac{(1-\alpha)n}{2} = \frac{\alpha(1-\alpha)n^2}{4k}$. It just remains to set α and k properly so to get the statement of the theorem.

Let $\alpha = 9/10$ and recall that $k = 1/\varepsilon$. Then, first, $\frac{(1-\alpha)nk}{8} = \frac{nk}{80} = \frac{n}{80\,\varepsilon}$. Second, (24) gives $\mathbb{E}[\mathrm{OPT}] < \frac{(1-\alpha)^2n^2}{k} = \frac{n^2}{100k} = \frac{\varepsilon n^2}{100}$. Third, $\frac{\alpha(1-\alpha)n^2}{4k} = \frac{9n^2}{400k} = \frac{9\varepsilon n^2}{400} > \mathbb{E}[\mathrm{OPT}] + \frac{\varepsilon n^2}{80}$. The above statement hence becomes: if $\mathbb{E}[Q] < \frac{n}{80\varepsilon}$, then $\mathbb{E}[\Delta] > \mathbb{E}[\mathrm{OPT}] + \frac{\varepsilon n^2}{80}$. An application of Yao's minimax principle completes the proof.

As a final note, we observe that for every $c \geq 1$ the bound can be put in the form $\mathbb{E}[\Delta] \geq c\cdot\mathbb{E}[\mathrm{OPT}] + \Omega(n^2\varepsilon)$ by choosing $\alpha \geq c/(c+1/4)$.

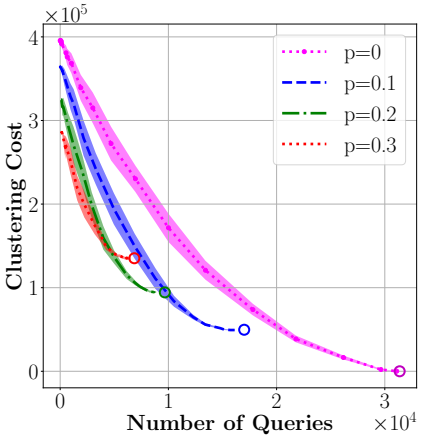
E Supplementary Material for Section 7

We report the complete experimental evaluation of ACC including error bars (see the main paper for a full description of the experimental setting). The details of the datasets are found in Table 1.

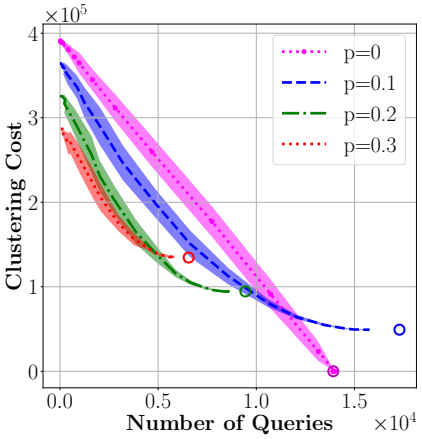
Table 1: Description of the datasets.

Datasets	Type	$\lvert V\rvert$	#Clusters
captchas	Real	244	69
cora	Real-world	1879	191
gym	Real	94	12
landmarks	Real	266	12
skew	Synthetic	900	30
sqrt	Synthetic	900	30

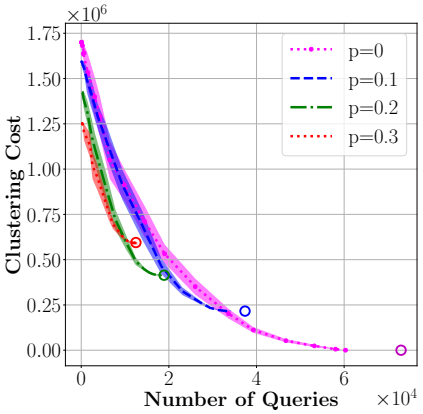
Figure 2: Clustering cost vs. number of queries. The dotted curves are the average cost and the shaded areas around them measure the standard deviation. The trailing diamonds refer to KwikCluster's performance.



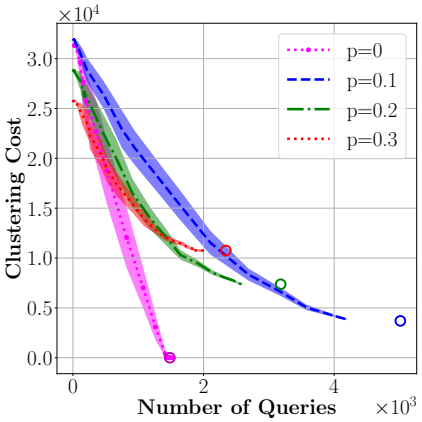
(a) skew.



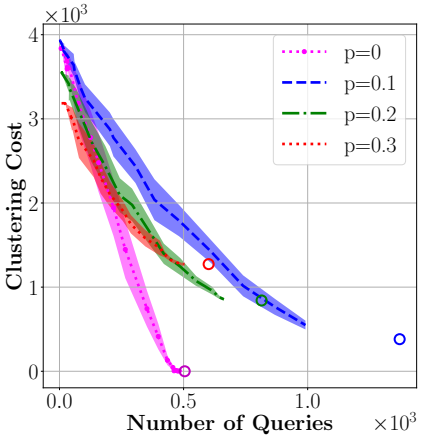
(b) sqrt.



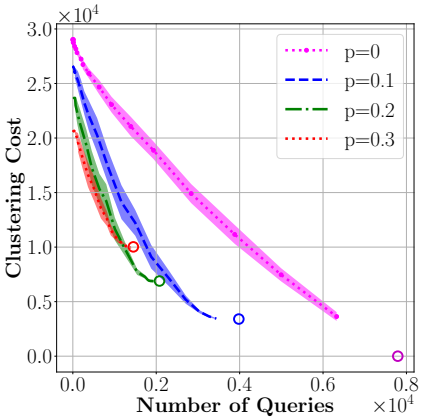
(c) cora.



(d) landmarks.



(e) gym.



(f) captchas.