Learning Partitions with Optimal Query and Round Complexities

Hadley Black HABLACK@UCSD.EDU
Arya Mazumdar ARYA@UCSD.EDU
Barna Saha BSAHA@UCSD.EDU
University of California, San Diego

Editors: Nika Haghtalab and Ankur Moitra

Abstract

We consider the basic problem of learning an unknown partition of n elements into at most k sets using simple queries that reveal information about a small subset of elements. Our starting point is the popular and well-studied pairwise *same-set* queries which ask if a pair of elements belong to the same class. It is well-known that non-adaptive (fully parallel) algorithms require $\Theta(n^2)$ queries, while adaptive (fully sequential) algorithms require $\Theta(nk)$ queries, and the best known algorithm uses $k-1$ rounds of adaptivity. Many variations of this problem have been studied over the last two decades in multiple disciplines due to its fundamental nature and connections to clustering, active learning, and crowd-sourcing. In many of these applications, it is of paramount interest to reduce adaptivity, a.k.a the number of rounds, while minimizing the query complexity. In this paper, we give a complete characterization of the query complexity of this problem as a function of the number of rounds, r, which interpolates smoothly between the non-adaptive and adaptive settings: for any constant $r \geq 1$, the query complexity is $\Theta(n^{1+\frac{1}{2^r-1}}k^{1-\frac{1}{2^r-1}})$. Additionally, our algorithm only needs $O(\log\log n)$ rounds to attain the optimal $O(nk)$ query complexity, which is a double-exponential improvement over prior works when k is a polynomial in n.

Next, we consider two natural generalizations of pair-wise queries to general subsets S of size at most s: (1) weak subset queries which return the number of classes intersected by S, and (2) strong subset queries which return the entire partition restricted on S. Once again in crowd sourcing applications, queries on large sets may be prohibitive. For non-adaptive algorithms, we show $\Omega(n^2/s^2)$ strong queries are needed. In contrast, perhaps surprisingly, we show that there is a non-adaptive algorithm using weak queries that matches this bound up to log-factors for all $s \leq \sqrt{n}$. More generally, we obtain nearly matching upper and lower bounds for algorithms using weak and strong queries in terms of both the number of rounds, r, and the query size bound, s.

Keywords: Partition learning, clustering, query complexity, round complexity

1. Introduction

Learning set partitions is fundamental to many applications including unsupervised learning, entity resolution, and network reconstruction. We consider the basic algorithmic problem of learning an unknown partition $\mathcal{P} = (C_1, \dots, C_k)$ of a universe U of n elements via access to an oracle that provides information about a queried subset $S \subseteq U$. Our starting point is the *pairwise same-set* query oracle, which returns whether a queried pair of elements $\{x, y\} \subset U$ belong to the same class. As far as we know, partition learning under same-set queries goes back at least to Reyzin and Srivastava (2007) who considered the problem of learning the connected components of a graph. The problem was later introduced more broadly in the learning theory and clustering literature by independent works of Ashtiani et al. (2016), Mazumdar and Saha (2017a,b,c), and Mitzenmacher and Tsourakakis (2016). In parallel, the problem garnered interest in the database community as an important primitive to develop crowd-sourced databases Feng et al. (2011); Mozafari et al. (2012); Davidson et al. (2014). Since then, it has been studied extensively over the last decade Saha and Subramanian (2019); Huleihel et al. (2019); Bressan et al. (2020); Del Pia et al. (2022); DePavia et al. (2024) due to its fundamental nature and relevance to clustering, machine learning, and databases.

In particular, the problem naturally models a situation where one would like to learn a hidden ground-truth clustering when obtaining explicit class labels is difficult or unnecessary, but deciding whether two elements have the same class label is easy. Indeed, learning partitions in this model can be viewed as a *label-invariant* form of clustering. In many applications class labels are difficult to discern computationally due to noisy and incomplete data, but different classes are trivial to distinguish with the human eye. Thus, applications have been developed which implement clustering via same-set queries, where external crowd-workers play the role of the oracle, e.g. Franklin et al. (2011); Wang et al. (2012). These queries are also straightforward to implement Mazumdar and Pal (2017), making them broadly applicable. Beyond its many motivating applications, we believe that partition learning with same-set queries is an extremely fundamental algorithmic problem akin to basic questions like comparison-based sorting, and as such deserves a thorough theoretical study.

Learning partitions with pairwise queries in few rounds. In crowdsource clustering applications, parallelization of queries is paramount to minimizing execution time, since one may not have control over how long it takes for queries to be answered Gu and Han (2012). In the context of query-algorithms, parallelism is formalized in terms of *round-complexity*: an algorithm has round-complexity r if its queries can be batched into r rounds $Q_1, \ldots, Q_r \subset U^2$ where the round t queries, Q_t, are made all at once (in parallel) and only depend on the oracle's response to the queries in the previous rounds. We say that such an algorithm uses r *rounds of adaptivity*. An algorithm is called *non-adaptive* if $r = 1$ and *fully adaptive* if no bound is given for r. It is well-known that the same-set query complexity of partition learning is $\Theta(nk)$ for fully adaptive algorithms Reyzin and Srivastava (2007); Davidson et al. (2014); Mazumdar and Saha (2017a); Liu and Mukherjee (2022) and $\Theta(n^2)$ for non-adaptive algorithms Mazumdar and Saha (2017a); Black et al. (2024). However, despite numerous works studying same-set queries, and the clear motivation for simultaneously minimizing adaptivity and query complexity, there has not yet been a study of the round-complexity of this basic question. In particular, the best known algorithm achieving query complexity $O(nk)$ uses $k-1$ rounds Reyzin and Srivastava (2007). On the other hand, the $O(n^2)$ non-adaptive upper bound comes by trivially querying every pair of elements, and this is provably optimal, even when there are only $k = 3$ sets in the partition. We ask, *can this be significantly improved on using few rounds?*

Our work fills this glaring gap: for every constant $r \geq 1$, we show matching upper and lower bounds of $\Theta(n^{1+\frac{1}{2^r-1}} k^{1-\frac{1}{2^r-1}})$, interpolating smoothly between the non-adaptive and fully adaptive settings. Moreover, our algorithm attains the optimal query complexity using only $O(\log \log n)$ rounds. As previous state of the art algorithms use $O(k)$ rounds, this is a double-exponential improvement over prior works when $k = \text{poly}(n)$. Technically speaking, our algorithm uses a simple recursive framework which may be useful for improving round-complexity for other problems. Our lower bound uses a novel application of Turán's theorem to construct hard instances, and we believe our techniques may be useful for proving lower bounds for other query-based algorithmic problems for graphs and other combinatorial objects. See Section 2 for explicit statements and further discussion on our results for same-set queries.

Generalizing to subsets while minimizing query size and rounds. Next, we consider generalizations of the popular pairwise same-set query model to general subsets $S \subseteq U$. Beyond our specific problem, there has been recent interest in the learning theory community in learning concepts using subset, or group queries Kontonis et al. (2024). Recent works Chakrabarty and Liao (2024); Black et al. (2024) introduced the partition learning problem with access to an oracle returning the number of sets in the partition intersecting the query, $S \subseteq U$. An information-theoretic lower bound is $\Omega(n)$ and Chakrabarty and Liao (2024) obtained a matching $O(n)$ query adaptive algorithm, while for non-adaptive algorithms Black et al. (2024) showed that $O(n \log k \cdot (\log k + \log\log n)^2)$ queries is possible. From a practical perspective, an obvious downside to these algorithms is that queried subsets can be large: Chakrabarty and Liao (2024) uses $O(k)$ sized queries and Black et al. (2024) uses $O(n)$ sized queries. To address this, we investigate the query complexity of learning partitions when a bound of s is placed on the allowed query size. In both the adaptive and non-adaptive cases, we obtain algorithms with the same query complexity up to logarithmic factors, while shrinking the query size *quadratically*, which is also optimal. Moreover, we obtain nearly matching upper and lower bounds for query size s in terms of *round-complexity*, similar to our results for pairwise same-set queries.

To motivate subset queries that count the number of intersected classes (weak subset queries), we also study the strongest possible type of subset query, which returns a full representation of the partition on the queried subset (strong subset queries). Surprisingly, we show that the number of weak vs. strong subset queries that are required is the same, up to logarithmic factors, for all s up to a reasonably large threshold ($s \leq O(\sqrt{n})$ for non-adaptive and $s \leq O(\sqrt{k})$ for adaptive), while weak queries require *significantly* less communication. A detailed discussion of our results for subset queries is given in Section 2.1.

Organization All our results are summarized in Section 2. We prove our lower and upper bounds for pairwise queries in Sections 3 and 4, respectively. Our non-adaptive weak subset query algorithm is outlined and proven in Section 5 and Appendix B, respectively, and the extension to r-rounds is described in Appendix C. Our results for strong subset queries are proven in Appendix D.

2. Main Results

We first consider the basic problem of learning an arbitrary unknown k-partition[1] of n elements using pairwise same-set queries: given $x, y \in U$ and a hidden partition $\mathcal{P}$, $\textsf{same-set}(x, y, \mathcal{P}) = \textsf{yes}$ if x, y belong to the same set in $\mathcal{P}$ and $\textsf{same-set}(x, y, \mathcal{P}) = \textsf{no}$ otherwise.

Our focus is on *round-complexity*: we first design a simple r-round deterministic algorithm which attains the optimal $O(nk)$ query complexity using only $O(\log\log n)$ rounds, which is a double-exponential improvement over the previous best $k-1$ round algorithm when $k = \mathrm{poly}(n)$. In general, with $O(\log 1/\varepsilon)$ rounds, our algorithm has query complexity $O(n^{1+\varepsilon} k^{1-\varepsilon})$, which interpolates smoothly between the non-adaptive and fully adaptive settings.

Theorem 1 (Pair query upper bound) *For any $r, k \geq 1$, there exists a deterministic r-round algorithm for k-partition learning using at most $8n^{(1+\frac{1}{2^r-1})}k^{(1-\frac{1}{2^r-1})}$ pairwise same-set queries.*

1. We use the term k-partition as shorthand for a partition into *at most* k sets.

In fact, we show that our algorithm attains the optimal interpolation between the non-adaptive and fully adaptive, $O(\log\log n)$ round setting, up to a factor of r. In particular, our upper and lower bounds are exactly matching for every constant r, and are always tight up to a $O(\log\log n)$ factor.

Theorem 2 (Pair query lower bound) *For all $r \geq 1$, any r-round deterministic algorithm for k-partition learning must use at least $\Omega\big(\frac{1}{r}\cdot n^{(1+\frac{1}{2^r-1})}k^{(1-\frac{1}{2^r-1})}\big)$ pairwise same-set queries.*

We remark that it is still open to establish such a lower bound for arbitrary *randomized* algorithms, and we believe that additional technical ideas are needed to achieve such an extension.

2.1. Subset Queries

Next, we consider the two following generalizations of pairwise same-set queries to subsets. Given hidden partition $\mathcal{P}$ and a queried subset $S \subseteq U$, each oracle returns the following information.

- *Weak subset query oracle:* Given $S \subseteq U$, the oracle returns $\mathsf{count}(S,\mathcal{P}) := \sum_{X\in\mathcal{P}} \mathbf{1}(S\cap X \neq \emptyset)$, i.e. the number of parts which S intersects.[2]
- *Strong subset query oracle:* Given $S \subseteq U$, the oracle returns $\mathsf{partition}(S,\mathcal{P}) := \{S\cap X \colon X \in \mathcal{P}\}$, i.e. a full description of $\mathcal{P}$ restricted on S.

We are interested in the query complexity of learning partitions when an upper bound of $s \in [2,n]$ is placed on the allowed size of a queried subset. Strong queries are the *most informative* type of subset query that one can define for the partition learning problem and thus provide a meaningful benchmark against which to measure the effectiveness of other query types. When $s = 2$, both queries are equivalent (in fact they are the same as pairwise queries). At the other extreme, when $s = n$ a single strong query recovers the entire partition, while a weak query only returns the number of parts. Intuitively, as s increases, the gap between weak and strong queries widens. Given that weak queries require significantly less communication from the oracle ($O(\log k)$ as opposed to $O(s\log k)$ bits), as well as less computation to answer, a motivating question for this line of inquiry is: *is there a regime of $s \gg 2$ where weak and strong queries are similarly powerful?*

The non-adaptive case. Obtaining lower bounds, even for strong subset queries, is straightforward using known lower bounds for pairwise queries: observe that one s-bounded strong subset query can be simulated (non-adaptively) by $\binom{s}{2}$ pair-wise same-set queries[3]. Thus, for non-adaptive algorithms, the $\Omega(n^2)$ lower bound for pairwise queries Mazumdar and Saha (2017a); Black et al. (2024) implies that $\Omega(n^2/s^2)$ strong queries are necessary. In fact, we prove (in Section D) that there is also a simple deterministic non-adaptive algorithm matching this bound.

Theorem 3 (Non-adaptive strong queries) *For $s \in [2,n]$, the non-adaptive strong query complexity of partition learning is $\Theta(n^2/s^2)$. The algorithm is deterministic and the lower bound holds even for randomized algorithms.*

On the other hand, a weak subset query contains at most $O(\log k)$ bits of information and there are $k^{\Omega(n)}$ partitions possible, implying an information-theoretic lower bound of $\Omega(n)$, even when

2. Weak subset queries are also sometimes referred to as "rank queries", e.g. Chakrabarty and Liao (2024), or simply as "subset queries", e.g. Black et al. (2024).
3. Given a set S, one can query all pairs in S and compute the entire partition restricted on S.

$s = n$. Therefore, there is a separation between weak and strong queries when $s \gg \sqrt{n}$, but this leaves as a possibility that weak and strong queries could have similar power when $s = O(\sqrt{n})$. Previous work of Black et al. (2024) provided two algorithms making $\widetilde{O}(n^2k/s^2)$ and $\widetilde{O}(n^2/s)$ queries, respectively, but it remained open whether $\widetilde{O}(n^2/s^2)$ is possible. Our main result for non-adaptive subset queries provides an affirmative answer to this open question. (Question 1.14 of Black et al. (2024).)

Theorem 4 (Non-adaptive weak queries, Theorem 12 informal) *For all $s \in [2, \sqrt{n}]$, the weak subset query complexity of partition learning is $\widetilde{\Theta}(n^2/s^2)$. Our algorithm is randomized and succeeds with probability $1 - 1/\mathrm{poly}(n)$.*

We find this result to be surprising since intuitively strong queries seem to be *significantly* more informative, yet up to logarithmic factors they provide no advantage for $s \leq \sqrt{n}$. From an applications perspective, this provides a compelling case for weak subset queries as they are nearly as informative as the strongest possible type of subset query, while being both (a) simpler to answer and (b) requiring significantly less communication.

The adaptive case. Again, since one strong subset query of size s can be simulated using $O(s^2)$ pairwise queries, *any* lower bound for pairwise query algorithms extends to subset queries with an additional $1/s^2$ factor. Thus the $\Omega(nk)$ lower bound for fully adaptive pair query algorithms Mazumdar and Saha (2017a); Liu and Mukherjee (2022) extends to an $\Omega(nk/s^2)$ lower bound for strong subset queries. More generally, our lower bound Theorem 2 extends in the same fashion. Moreover, we use our non-adaptive subset query algorithms above combined with the algorithmic strategy used to obtain our r-round pair query algorithm of Theorem 1 to prove the following bounds on the query complexity of r-round subset query algorithms for partition learning.

Theorem 5 (r-round weak subset queries) *For every $s \geq 2$, $k \geq 1$, and $r \geq 1$, there is a randomized r-round algorithm for k-partition learning that succeeds with probability $1 - 1/\mathrm{poly}(n)$ using $\widetilde{O}\big(\max\big(\frac{1}{s^2} \cdot n^{(1+\frac{1}{2^r-1})}k^{(1-\frac{1}{2^r-1})}, n\big)\big)$ s-bounded weak subset queries and any r-round deterministic algorithm for this task must use $\Omega\big(\max\big(\frac{1}{r} \cdot \frac{1}{s^2} \cdot n^{(1+\frac{1}{2^r-1})}k^{(1-\frac{1}{2^r-1})}, n\big)\big)$ s-bounded weak subset queries.*

Our upper and lower bounds are tight for constant r. Using $r = O(\log\log n)$ rounds and query size bound $s = O(\sqrt{k})$ our algorithm has nearly-linear query complexity, $\widetilde{O}(n)$. This is in contrast to Chakrabarty and Liao (2024) who obtained an $O(n)$ query algorithm using $r = O(\log k)$ and $s = O(k)$.

We obtain similar results for strong subset queries. Note that an s-bounded strong query contains $O(s \log k)$ bits of information, implying an $\Omega(n/s)$ information-theoretic lower bound.

Theorem 6 (r-round strong subset queries) *For every $s \geq 2$, $k \geq 1$, and $r \geq 1$, there is a deterministic r-round algorithm for k-partition learning using $O\big(\max\big(\frac{1}{s^2} \cdot n^{(1+\frac{1}{2^r-1})}k^{(1-\frac{1}{2^r-1})}, \frac{n}{s}\big)\big)$ s-bounded strong subset queries and any r-round deterministic algorithm for this task must use $\Omega\big(\max\big(\frac{1}{r} \cdot \frac{1}{s^2} \cdot n^{(1+\frac{1}{2^r-1})}k^{(1-\frac{1}{2^r-1})}, \frac{n}{s}\big)\big)$ s-bounded strong subset queries.*

Again, our bounds are tight for every constant r. Note that the query complexity of strong and weak subset queries is the same up to logarithmic factors when $s \leq \sqrt{n^{\frac{1}{2^r-1}}k^{1-\frac{1}{2^r-1}}}$, i.e. until the information-theoretic lower bound is reached for weak subset queries. However, strong subset queries require $\widetilde{\Omega}(s)$ bits of communication by the oracle and so the total communication is never less for strong query algorithms. Refer to Fig. 5 in the appendix for a visual comparison of strong vs. weak subset queries in terms of s for non-adaptive and fully adaptive algorithms.

3. Lower Bound for Pair Queries

In this section we prove our lower bound on r-round deterministic algorithms for learning k-partitions, Theorem 2. As a warm-up, we begin by sketching an $\Omega(n^{1+\frac{1}{2^r-1}})$ lower bound, and then proceed to the main proof. At a high level, our approach is to view queries as edges of a graph, and exploit independent sets in this graph to construct hard instances, i.e. a pair of partitions that the algorithm fails to distinguish. To find large independent sets we use Turán's theorem, which we state and prove (in Appendix A.1) for completeness.

Theorem 7 (Turán's Theorem) *Let $G = (V, E)$ be an undirected graph with n vertices and average degree d_G. Then G contains an independent set of size at least $\frac{n}{1+d_G}$.*

Warm-up: an $\Omega(n^{1+\frac{1}{2^r-1}})$ using Turán's Theorem. Let $r \leq \frac{1}{10}\log\log n$ and consider any r-round deterministic algorithm $\mathcal{A}$. For brevity, we use $\varepsilon(r) := \frac{1}{2^r-1}$ for all $r \geq 1$.

We will inductively construct a pair of k-partitions that $\mathcal{A}$ fails to distinguish. The base case is when $r = 1$. For this, suppose $\mathcal{A}$ makes strictly fewer than $\binom{n}{2}$ queries. Then there is a pair of points $x, y \in U$ for which (x, y) is not queried. Then the 3-partitions $(U, \{x, y\})$ and $(U, \{x\}, \{y\})$ are not distinguished and this completes the base case. (I.e. the oracle returns the same answer for the two partitions on every query made by $\mathcal{A}$.)

Now, let $r \geq 2$ and $k \geq r+2$ and suppose for the sake of contradiction that $\mathcal{A}$ makes fewer than $\frac{1}{3} \cdot n^{1+\varepsilon(r)}$ queries in the first round. Let $G = (U, E)$ be the graph on U whose edge-set is given by this set of queries. The average degree in this graph is at most $\frac{1}{3} \cdot n^{\varepsilon(r)}$ and so by Turán's theorem, G has an independent set Z of size $\frac{n}{1+\frac{1}{3}\cdot n^{\varepsilon(r)}} \geq n^{1-\varepsilon(r)}$ by our upper bound assumption on r. In particular, in the first round, $\mathcal{A}$ makes *no queries whatsoever* in Z. Let $\mathcal{A}_{r-1,Z}$ denote the algorithm using the remaining $r-1$ rounds of $\mathcal{A}$ restricted on Z. Construct a pair of partitions by letting $U \setminus Z$ be a set in each, and within Z inductively define the hard pair of partitions for $(r-1)$-round algorithms for learning a $(k-1)$-partition. By induction $\mathcal{A}_{r-1,Z}$ must make at least

$$\frac{1}{3} \cdot |Z|^{1+\varepsilon(r-1)} \geq \frac{1}{3} \cdot \left(n^{1-\varepsilon(r)}\right)^{1+\varepsilon(r-1)} = \frac{1}{3} \cdot n^{(1-\varepsilon(r))(1+\varepsilon(r-1))} = \frac{1}{3} \cdot n^{1+\varepsilon(r)}$$

queries, where in the last step we used item (1) of Claim 11. Thus, we have a contradiction and this completes the proof.

We now show how to strengthen the above argument to obtain an additional dependence on k, and ultimately prove Theorem 2. First, it is not too hard to see that one can repeatedly apply Turán's theorem to obtain a collection of disjoint independent sets, giving the following Corollary, which we prove in Appendix A.2.

Corollary 8 (Repeated Turán's Theorem) *Let $G = (V, E)$ be an undirected graph with n vertices and $m \geq n$ edges. Let $N \leq n^2/8m$ and $\ell \leq n/2N$. Then, G contains ℓ disjoint independent sets in G, each of size N.*

Theorem 2 proof overview. We now informally describe the proof of Theorem 2. Let $\mathcal{A}$ be an arbitrary deterministic r-round algorithm making $\ll n^{1+\varepsilon(r)}(k/r)^{1-\varepsilon(r)}$ queries, where Q_t denotes the set of queries made in the t-th round. Note that Q_1 is a fixed, pre-determined set, but for $t \geq 2$, Q_t depends on the oracle's responses to the queries in $Q_1 \cup \cdots \cup Q_{t-1}$. We need to show that there exists a pair of k-partitions that $\mathcal{A}$ does not distinguish. To accomplish this we first show that there is a single partition $\mathcal{P}$ such that after running the first $r-1$ rounds of $\mathcal{A}$ on $\mathcal{P}$, there still exists a "large" set S (the size of S will be $\approx \sqrt{n^{1+\varepsilon(r)}(k/r)^{1-\varepsilon(r)}}$) such that for *every query* $(u, v) \in Q_1 \cup \cdots \cup Q_{r-1}$ *that touches* S (at least one of u, v belong to S), u and v belong to different sets in $\mathcal{P}$. Now, we can take any un-queried pair $(x, y) \in \binom{S}{2}$ and define $\mathcal{P}_{x,y}^{(1)}$, $\mathcal{P}_{x,y}^{(2)}$ which modify $\mathcal{P}$ by either making $\{x\}, \{y\}$ two separate sets or making $\{x, y\}$ a single set (see Fig. 2). Crucially, this is *well-defined* because all queried pairs involving x or y span different sets in $\mathcal{P}$. I.e. the oracle's responses on all queries in the first $r-1$ rounds are consistent between the partitions $\mathcal{P}$ and $\mathcal{P}_{x,y}^{(b)}$. Finally, to distinguish $\mathcal{P}_{x,y}^{(1)}$, $\mathcal{P}_{x,y}^{(2)}$ the final round of queries Q_r must contain the pair (x, y). Thus, we can conclude Q_r must contain every pair in $\binom{S}{2} \setminus (Q_1 \cup \cdots \cup Q_{r-1})$, for otherwise there is some pair $\mathcal{P}_{x,y}^{(1)}$, $\mathcal{P}_{x,y}^{(2)}$ that is not distinguished by the algorithm. This shows that we must have $|Q_r| \approx n^{1+\varepsilon(r)}(k/r)^{1-\varepsilon(r)}$, contradicting the assumed upper bound on the number of queries.

Now, the main effort in the proof is in constructing the partition $\mathcal{P}$ and the set S described above. This is accomplished by inductively constructing a sequence of partitions $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{r-1}$ and sets $S^{(1)}, S^{(2)}, \ldots, S^{(r-1)}$ and taking $\mathcal{P} := \mathcal{P}_{r-1}$ and $S := S^{(r-1)}$. Essentially, the property we want each pair $(\mathcal{P}_t, S^{(t)})$ in the sequence to have is what is described in the previous paragraph: every query made in the first t rounds which touches $S^{(t)}$ is spanning different sets in $\mathcal{P}_t$. This is done in the following manner. Invoke the repeated Turán theorem on $G(U, Q_1)$ to obtain $\ell \approx k/r$ disjoint independent sets $S_1^{(1)}, \ldots, S_\ell^{(1)}$ and let $S^{(1)}$ denote their union. Define $\mathcal{P}_1 = \{U \setminus S^{(1)}, S_1^{(1)}, \ldots, S_\ell^{(1)}\}$ and observe that $(\mathcal{P}_1, S^{(1)})$ has the desired property for the first round. Now, given $(\mathcal{P}_{t-1}, S^{(t-1)})$ with the desired property for the first $t-1$ rounds, we again invoke the repeated Turán theorem, but this time on $G(S^{(t-1)}, Q_1 \cup \cdots \cup Q_{t-1})$ to obtain disjoint independent sets $S_1^{(t)}, \ldots, S_\ell^{(t)}$ and let $S^{(t)} \subseteq S^{(t-1)}$ denote their union. Now, starting from $\mathcal{P}_{t-1}$, we construct $\mathcal{P}_t$ which will define the oracle's response to the t-th round queries and which has the desired property for the first t rounds. Each round introduces $\ell \approx k/r$ new sets into the partition and so at the end we have $\approx k$ sets. A subtle aspect of the argument is that one must be very careful to ensure that the oracle's responses on the first $t-1$ rounds of queries are consistent between $\mathcal{P}_t$ and $\mathcal{P}_{t-1}$. This is because the set Q_t is determined by the oracle's responses to $Q_1 \cup \cdots \cup Q_{t-1}$ on $\mathcal{P}_{t-1}$ which then define $\mathcal{P}_t$. Thus, without this consistency property, this process would be ill-defined. (See Fig. 1 for an accompanying illustration of this argument.)

3.1. Proof of Theorem 2

For convenience throughout the proof we will use $\varepsilon(r) := \frac{1}{2^r - 1}$ for every $r \geq 1$ and $\ell := \lfloor \frac{k-3}{r-1} \rfloor$ which satisfies $\ell \geq 1$ since $r \leq k-2$ by assumption.

Consider an arbitrary r-round deterministic algorithm and for each $t \in \{1, 2, \ldots, r\}$, let Q_t denote the queries made in round t. Note that for $t \geq 2$, this set depends on the query responses on the previous queries, $Q_1 \cup \cdots \cup Q_{t-1}$. We will assume that $|Q_1 \cup \cdots \cup Q_{r-1}| \leq \frac{n\ell}{100}(n/2\ell)^{\varepsilon(r)}$ (since otherwise the theorem already holds for this algorithm) and using this assumption we will argue that $|Q_r| \geq \Omega(n\ell(n/\ell)^{\varepsilon(r)})$.

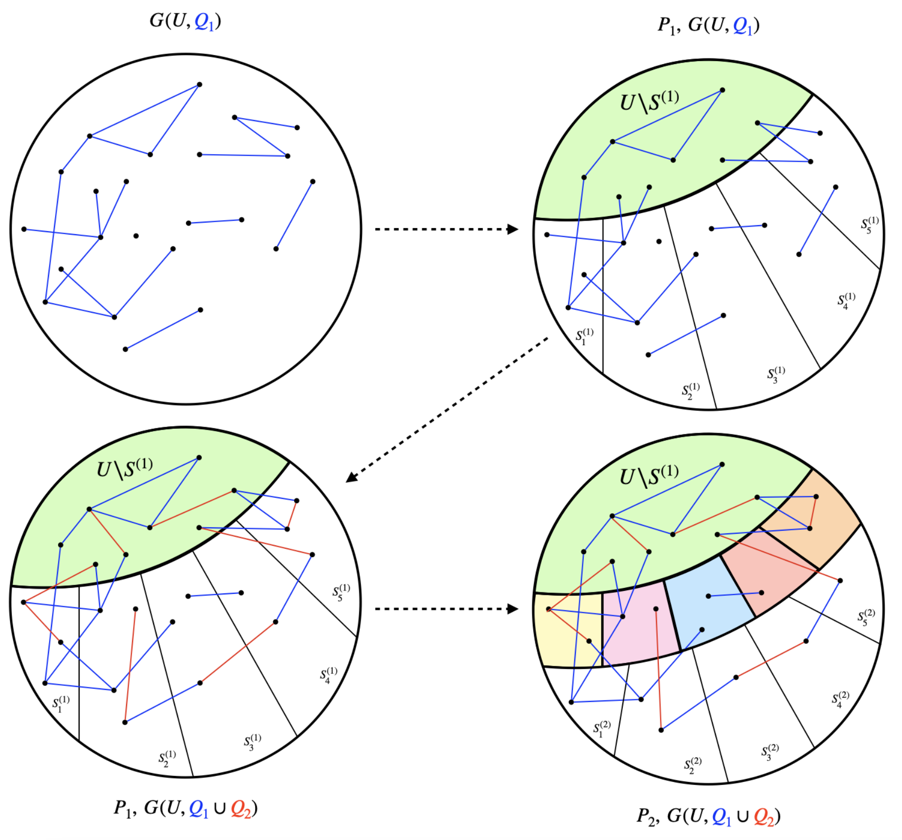

Figure 1: An illustration depicting the construction of $\mathcal{P}_1, S^{(1)}$ and $\mathcal{P}_2, S^{(2)}$ in the proof of Theorem 2. The top-left shows the graph whose edges are the first round queries, Q_1 (blue edges). Then, Lemma 9 is applied which uses Turán's theorem to find ℓ (in the picture $\ell = 5$) independent sets $S^{(1)} = S^{(1)}_1 \sqcup \cdots \sqcup S^{(1)}_5 \subset U$ with respect to Q_1. The partition $\mathcal{P}_1$ is defined based on these independent sets (top-right). Then, based on the oracle's responses to Q_1, the second round queries, Q_2, arrive (red edges, bottom-left). Again, Lemma 9 is applied to find ℓ independent sets $S^{(2)} = S^{(2)}_1 \sqcup \cdots \sqcup S^{(2)}_5 \subset S^{(1)}$ with respect to $Q_1 \cup Q_2$ and the partition $\mathcal{P}_2$ is defined (bottom-right). The (non-green) shaded regions in the bottom-right represent the sets $S^{(1)}_j \setminus S^{(2)}$ for each $j \in [\ell]$. The construction repeats in this way for $r-1$ rounds. A shaded region depicts a set in the partition which is fixed for the remainder of the construction, e.g. the green region is a set in $\mathcal{P}_1$, and remains a set in $\mathcal{P}_2, \mathcal{P}_3$, etc. A white region depicts a set in the partition which may be fragmented when a later partition is defined.

Additionally, we will assume that $|Q_1| \geq n$ and hence $|Q_1 \cup \cdots \cup Q_t| \geq n$ for all $t \geq 1$. This is without loss of generality since increasing $|Q_1|$ can only help the algorithm. Finally, we assume that $n > C\ell$ for a sufficiently large constant C.

First, $|Q_1| \leq \frac{n\ell}{100}(n/2\ell)^{\varepsilon(r)}$ and so we can apply the repeated Turán theorem (Corollary 8) on the graph $G(U, Q_1)$ to obtain ℓ disjoint independent sets of size $N_1 := \left\lfloor \left(\frac{n}{2\ell}\right)\left(\frac{2\ell}{n}\right)^{\varepsilon(r)} \right\rfloor \leq \frac{n^2}{2n\ell(n/2\ell)^{\varepsilon(r)}} < \frac{n^2}{8 \cdot \frac{n\ell}{100}(n/2\ell)^{\varepsilon(r)}} \leq \frac{n^2}{8|Q_1|}$.

Construction of $\mathcal{P}_1$ and $S^{(1)}$. Let $S_1^{(1)}, \ldots, S_\ell^{(1)}$ be the resulting independent sets in $G(U, Q_1)$, which are each of size at least N_1. Let $S^{(1)}$ denote their union and note that $|S^{(1)}| = \ell N_1$. We now define a partition $\mathcal{P}_1$ which will fix the oracle's response to each query in Q_1 as follows: (i) $U \setminus S^{(1)}$ is a set in $\mathcal{P}_1$. (ii) Each $S_i^{(1)}$ is a set in $\mathcal{P}_1$.

All queries in Q_1 with both endpoints in $U \setminus S^{(1)}$ are fixed to "yes" and all others are fixed to "no" (since each $S_j^{(1)}$ is an independent set in $G(U, Q_1)$). Given these oracle responses on Q_1, the second round of queries Q_2 is now determined. Our goal is now to repeat this idea iteratively over the first $r-1$ rounds. That is, given the t-th round of queries Q_t, we wish to construct $\mathcal{P}_t$ which fixes the oracle's responses on Q_t, and which is consistent with $\mathcal{P}_{t-1}$ on all previous queries so that this process is well-defined. The following lemma is our main tool for performing a single iteration of this process. We defer the proof of this lemma to Appendix A.3.

Lemma 9 (Carving Lemma) *Suppose we have a set of queries $Q \subseteq \binom{U}{2}$, $\ell \geq 1$ disjoint independent sets $S_1, \ldots, S_\ell$ in $G(U, Q)$, and a partition $\mathcal{P}$ of U such that $S_i \in \mathcal{P}$ for all $i \in [\ell]$. Then, given another set of queries $Q' \subseteq \binom{U}{2}$, and integers $N \leq \frac{|S|^2}{8|Q \cup Q'|}$ and $\ell' \leq |S|/2N$, there exist ℓ' disjoint independent sets $S_1', \ldots, S_{\ell'}'$ each of size N in $G(S, Q \cup Q')$ and a partition $\mathcal{P}'$ such that $|\mathcal{P}'| \leq |\mathcal{P}| + \ell'$ and the following hold:*

1. (Inductive Property) $S_j' \in \mathcal{P}'$ *for every* $i \in [\ell']$.
2. (Query Consistency) *For every* $\{x, y\} \in Q$, $\mathsf{same\text{-}set}(x, y, \mathcal{P}) = \mathsf{same\text{-}set}(x, y, \mathcal{P}')$.

The idea is now to use Lemma 9 iteratively to construct a sequence of partitions $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{r-1}$ as follows. Crucially, item (2) ensures that this process is well-defined, as we will see. In what follows, for every $t \in \{1, 2, \ldots, r-1\}$ define

$$N_t := \left\lfloor \left(\frac{n}{2\ell}\right) \cdot \left(\frac{2\ell}{n}\right)^{\frac{\varepsilon(r)}{\varepsilon(t)}} \right\rfloor > \left(\frac{n}{3\ell}\right) \cdot \left(\frac{2\ell}{n}\right)^{\frac{\varepsilon(r)}{\varepsilon(t)}} \tag{1}$$

where the inequality holds since $n > c\ell$ is sufficiently large and $\frac{\varepsilon(r)}{\varepsilon(t)} \leq \frac{\varepsilon(r)}{\varepsilon(r-1)} \leq 1/2$. (Note that this definition of N_t is consistent with the definition of N_1 above, since $\varepsilon(1) = 1$.)

Lemma 10 *For every $t \in \{1, 2, \ldots, r-1\}$, there exists a set $S^{(t)} = S_1^{(t)} \sqcup \cdots \sqcup S_\ell^{(t)}$ of size $|S^{(t)}| = \ell N_t$ and a partition $\mathcal{P}_t$ of U of size $|\mathcal{P}_t| \leq t\ell + 1$ such that the following hold.*

1. (IS Property) *For all $j \in [\ell]$, $S_j^{(t)} \in \mathcal{P}_t$ and is an independent set in $G(S^{(t-1)}, Q_1 \cup \cdots \cup Q_t)$.*
2. (Query Consistency) *For every $\{x, y\} \in Q_1 \cup \cdots \cup Q_{t-1}$,* $\mathsf{same\text{-}set}(x, y, \mathcal{P}_{t-1}) = \mathsf{same\text{-}set}(x, y, \mathcal{P}_t)$.

Proof The base case of $t = 1$ is established in the paragraphs preceding the statement of Lemma 9. I.e., we have $S^{(1)} = S_1^{(1)} \sqcup \cdots \sqcup S_\ell^{(1)}$ of size $|S^{(1)}| = \ell N_1$, and we have a partition $\mathcal{P}_1$ of size $|\mathcal{P}_1| = \ell + 1$ such that for each $j \in [\ell]$, $S_j^{(1)} \in \mathcal{P}_1$ and is an independent set in $G(U, Q_1)$.

Now let $2 \leq t \leq r-1$. Let $S^{(t-1)} = S_1^{(t-1)} \sqcup \cdots \sqcup S_\ell^{(t-1)}$ and $\mathcal{P}_{t-1}$ be the sets and partition given by induction. In particular, the query responses on $\mathcal{P}_{t-1}$ now determine the t-th round queries Q_t. We now construct a partition $\mathcal{P}_t$ which will define the query responses in Q_t. As mentioned, the responses on the previous queries $Q_1 \cup \cdots \cup Q_{t-1}$ must be consistent between $\mathcal{P}_t$ and $\mathcal{P}_{t-1}$ so that this process is well-defined. (Otherwise, we would contradict the definition of the sets $Q_2, \ldots, Q_{t-1}$.) See Fig. 1 in the appendix for an accompanying illustration.

We will invoke Lemma 9 with $Q = Q_1 \cup \cdots \cup Q_{t-1}$ and $Q' = Q_t$. By induction $|S^{(t-1)}| \geq \frac{n}{3}(2\ell/n)^{\frac{\varepsilon(r)}{\varepsilon(t-1)}}$ and recall that by assumption we have $|Q_1 \cup \cdots \cup Q_t| \leq \frac{n\ell}{100}(n/2\ell)^{\varepsilon(r)}$. Thus, using our bound on N_{t-1} from eq. (1) we have

$$\frac{|S^{(t-1)}|^2}{8|Q_1 \cup \cdots \cup Q_t|} \geq \frac{\ell^2 N_{t-1}^2}{\frac{8n\ell}{100}(n/2\ell)^{\varepsilon(r)}} \geq \frac{n^2(2\ell/n)^{\frac{2\varepsilon(r)}{\varepsilon(t-1)}}}{\frac{72n\ell}{100}(n/2\ell)^{\varepsilon(r)}}$$
$$\geq \frac{25n}{18\ell} \cdot \left(\frac{2\ell}{n}\right)^{\frac{2\varepsilon(r)}{\varepsilon(t-1)} + \varepsilon(r)} = \frac{25n}{18\ell} \cdot \left(\frac{2\ell}{n}\right)^{\frac{\varepsilon(r)}{\varepsilon(t)}} > N_t \tag{2}$$

where the equality holds since $2(2^{t-1} - 1) + 1 = 2^t - 1$. Note also that $\ell N_t < \ell N_{t-1} = |S^{(t-1)}|$. Using this observation and eq. (2), we can invoke Lemma 9 with $N := N_t$ and $\ell' := \ell$. This yields ℓ disjoint independent sets $S^{(t)} := S_1^{(t)} \sqcup \cdots \sqcup S_\ell^{(t)}$ in $G(S^{t-1}, Q_1 \cup \cdots \cup Q_t)$ and a partition $\mathcal{P}_t$ such that (a) for every $j \in [\ell]$, we have $S_j^{(t)} \in \mathcal{P}_t$ (by item 1 of Lemma 9) and (b) for every $\{x, y\} \in Q_1 \cup \cdots \cup Q_{t-1}$, $\textsf{same-set}(x, y, \mathcal{P}_{t-1}) = \textsf{same-set}(x, y, \mathcal{P}_t)$ (by item 2 of Lemma 9). Furthermore, we also have $|\mathcal{P}_t| \leq |\mathcal{P}_{t-1}| + \ell \leq t\ell + 1$, using the guarantee of Lemma 9 and the inductive hypothesis. This completes the proof of Lemma 10. ■

Now, invoking Lemma 10 with $t := r-1$ shows that there exists a partition $\mathcal{P} := \mathcal{P}_{r-1}$ and a set $S := S^{(r-1)}$ such that $S = S_1 \sqcup \cdots \sqcup S_\ell$ and for all $j \in [\ell]$, $S_j \in \mathcal{P}$ and is an independent set in $G(U, Q_1 \cup \cdots \cup Q_{r-1})$. Also note that $|\mathcal{P}| \leq (r-1)\ell + 1 \leq k-2$ using the definition of ℓ. Moreover, by Lemma 10 and the bound on N_{r-1} from eq. (1), we have $|S| = \ell N_{r-1} \geq \frac{n}{3}(2\ell/n)^{\frac{2^{r-1}-1}{2^r-1}} \geq \frac{1}{3}\sqrt{n\ell(n/\ell)^{\varepsilon(r)}}$ where the last inequality holds since $\frac{2^{r-1}-1}{2^r-1} = \frac{1}{2} \cdot \frac{2^r-2}{2^r-1} = \frac{1}{2}\left(1 - \frac{1}{2^r-1}\right)$. This means that the number of pairs in S is at least $\binom{|S|}{2} \geq \frac{n\ell}{10}(n/\ell)^{\varepsilon(r)}$. Thus, the number of un-queried pairs in S is at least

$$\left|\binom{S}{2} \setminus (Q_1 \cup \cdots \cup Q_{r-1})\right| \geq \left(\frac{1}{10} - \frac{1}{100}\right) n\ell(n/\ell)^{\varepsilon(r)} > \frac{n\ell}{12}(n/\ell)^{\varepsilon(r)} \tag{3}$$

by our assumed upper bound on $|Q_1 \cup \cdots \cup Q_{r-1}|$. Let $A := \binom{S}{2} \setminus (Q_1 \cup \cdots \cup Q_{r-1})$ denote this set of pairs in S not queried in the first $r-1$ rounds. For every such pair $(x, y) \in A$ we define two partitions $\mathcal{P}_{x,y}^{(1)}$ and $\mathcal{P}_{x,y}^{(2)}$ as follows. (See Fig. 2 in the appendix for an accompanying illustration.)

- In $\mathcal{P}_{x,y}^{(1)}$, $\{x, y\}$ form one set of size 2 and in $\mathcal{P}_{x,y}^{(2)}$, $\{x\}$ and $\{y\}$ each form one singleton set.

- In both $\mathcal{P}_{x,y}^{(1)}$ and $\mathcal{P}_{x,y}^{(2)}$, all other points are consistent with the partition $\mathcal{P}$. Formally for every $X \in \mathcal{P}$, $X \setminus \{x, y\}$ is a set in both $\mathcal{P}_{x,y}^{(1)}$ and $\mathcal{P}_{x,y}^{(2)}$.

Clearly, we have $|\mathcal{P}_{x,y}^{(b)}| \leq |\mathcal{P}| + 2 \leq k$.

First, for this to be well-defined, we need to argue that for every query $(u, v) \in Q_1 \cup \cdots \cup Q_{r-1}$, the responses given on $\mathcal{P}$ and $\mathcal{P}_{x,y}^{(b)}$ (for either $b \in \{1, 2\}$) are consistent. Let (u, v) be an arbitrary such query. Clearly if $\{u, v\} \cap \{x, y\} = \emptyset$, then $\mathsf{same\text{-}set}(u, v, \mathcal{P}) = \mathsf{same\text{-}set}(u, v, \mathcal{P}_{x,y}^{(b)})$. Also, note that $\{u, v\} \neq \{x, y\}$ since $(x, y) \in A$. The remaining case is when $|\{u, v\} \cap \{x, y\}| = 1$. Without loss of generality, suppose $u = x$. First, if $v \notin S$, then clearly $\mathsf{same\text{-}set}(x, v, \mathcal{P}) = \mathsf{same\text{-}set}(x, v, \mathcal{P}_{x,y}^{(b)}) = \mathsf{no}$ (recall that $x \in S$ by definition of the pair (x, y)). Now, suppose that $v \in S$. The point is that since $S = S_1 \sqcup \cdots \sqcup S_k$ where each S_i is an independent set in $G(U, Q_1, \ldots, Q_{r-1})$, we must have that x, v lie in different S_i-s, and therefore $\mathsf{same\text{-}set}(x, v, \mathcal{P}) = \mathsf{same\text{-}set}(x, v, \mathcal{P}_{x,y}^{(b)}) = \mathsf{no}$ (recall that each $S_i \in \mathcal{P}$). Thus, all queries in the first $r - 1$ rounds are consistent on $\mathcal{P}$ and $\mathcal{P}_{x,y}^{(b)}$. Thus, $\mathcal{P}_{x,y}^{(1)}$ and $\mathcal{P}_{x,y}^{(2)}$ are well-defined.

Now, we will argue that it must be the case that $Q_r \supseteq A$. Suppose not, and so there is some $(x, y) \in A$ where $(x, y) \notin Q_r$. We will argue that the algorithm cannot distinguish $\mathcal{P}_{x,y}^{(1)}$ and $\mathcal{P}_{x,y}^{(2)}$. Consider any pair $(u, v) \neq (x, y)$ and observe that $\mathsf{same\text{-}set}(u, v, \mathcal{P}_{x,y}^{(1)}) = \mathsf{same\text{-}set}(u, v, \mathcal{P}_{x,y}^{(2)})$ by construction of $\mathcal{P}_{x,y}^{(1)}$ and $\mathcal{P}_{x,y}^{(2)}$. Thus, since $(x, y) \notin Q_r$ and $(x, y) \notin Q_1, \ldots, Q_{r-1}$, the algorithm does not distinguish these two partitions. This implies that $A \subseteq Q_r$ and consequently $|Q_r| \geq \frac{n\ell}{12}(n/\ell)^{\varepsilon(r)}$ by eq. (3). This completes the proof of Theorem 2.

4. Low-Round Algorithm using Pair Queries

In this section we prove Theorem 1, obtaining an r-round deterministic algorithm for learning a k-partition which interpolates between the trivial $\Theta(n^2)$ non-adaptive query complexity and the $\Theta(nk)$ fully adaptive query complexity. We use a simple recursive strategy described in pseudocode in Alg. 1: divide U into subproblems (line 6), compute the restricted partition in each part by (non-adaptively) querying every pair (line 8), and then recurse on a set R formed by taking exactly one representative from each set in each of the restricted partitions (lines 9-11). For the base case, simply use the trivial strategy of querying all pairs.

Proof of Theorem 1. For shorthand in the proof, we will use $\varepsilon(r) = \frac{1}{2^r - 1}$ for all $r \geq 1$. The algorithm is recursive and pseudocode is given in Alg. 1. We prove the theorem by induction on r. For the base case of $r = 1$ (lines (2-4)), we simply make all $\binom{n}{2}$ possible pairwise queries in U. The partition can trivially be recovered using this set of queries. Moreover, the number of queries is at most the desired bound since $\varepsilon(1) = 1$.

Now suppose $r \geq 2$. First, if $n \leq 16k$, then we simply make all pair-wise queries in U (line 3), for a total of $\binom{n}{2} \leq \frac{n^2}{2} \leq 8nk \leq 8n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}$ queries, where the last step simply used $n \geq k$. Note that in this case the algorithm is non-adaptive and the partition is trivially recovered. This completes the proof for the case of $n \leq 16k$.

Now suppose that $n > 16k$. Let us first establish correctness of the algorithm. First, we take a partition $U = U_1 \sqcup \cdots \sqcup U_\ell$ and then within each U_i we make all pair-wise queries and trivially recover the partition restricted in U_i, i.e. $\mathcal{C}_i = \{C \cap U_i \colon C \in \mathcal{C}\}$. Next, our goal is to merge these partitions to recover $\mathcal{C}$. To do so, we form a set R containing one representative from every

Algorithm 1: $\mathsf{LR\text{-}SameSetQuery}(U, r)$

1 **Input:** Pair query access to hidden partition $\mathcal{C}$ over U with n points and $|\mathcal{C}| \leq k$. An allowed number of rounds r;
2 **if** $r = 1$ ***or*** $n \leq 16k$ **then**
3   **Query** every pair $(x, y) \in \binom{U}{2}$ and **return** the computed partition;
4 **end**
5 **else**
6   Partition the n points of U arbitrarily into at most $\ell := \lceil \frac{1}{3} \left(\frac{n}{k}\right)^{1-\frac{1}{2^r-1}} \rceil$ sets $U_1, \ldots, U_\ell$ each of size $|U_i| \leq t := \lceil 3n^{\frac{1}{2^r-1}} k^{1-\frac{1}{2^r-1}} \rceil$;
7   **for** $i \in [\ell]$ **do**
8   **Query** every pair $(x, y) \in \binom{U_i}{2}$ to learn the partition $\mathcal{C}_i = \{C \cap U_i \colon C \in \mathcal{C}\}$;
9   Form R_i by taking exactly one representative from each set $C \in \mathcal{C}_i$;
10   **end**
11   Let $R = R_1 \cup \cdots \cup R_\ell$, call $\mathsf{LR\text{-}SameSetQuery}(R, r-1)$, and let $\mathcal{C}_R$ be the returned partition of R;
12   **Return** the partition $\{\bigcup_{C' \in \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_\ell \colon C' \cap C \neq \emptyset} C' \colon C \in \mathcal{C}_R\}$;
13 **end**

set in $\mathcal{C}_i$ for every $i \in [\ell]$ (lines 9 and 11), and recursively learn the partition restricted on R, i.e. $\mathcal{C}_R = \{C \cap R \colon C \in \mathcal{C}\}$ (line 11). By induction this correctly computes $\mathcal{C}_R$ and allows us to compute the final partition by merging all sets $C' \in \mathcal{C}_1, \ldots, \mathcal{C}_\ell$ which intersect the same set $C \in \mathcal{C}_R$. This completes the proof of correctness.

We now complete the proof of the claimed query complexity. Recall that we are in the case of $r \geq 2$ and $n > 16k$. Since $r \geq 2$, note that $1 - \varepsilon(r) \geq 1 - \varepsilon(2) \geq 2/3$. Using these bounds, we have $(n/k)^{1-\varepsilon(r)} > 16^{2/3} > 6$. Recalling the definition of ℓ and t in line (6), this implies that $\ell \leq \frac{1}{3}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} + 1 = \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} - \frac{1}{6}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} + 1 < \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)}$ and clearly $t \leq 4n^{\varepsilon(r)} k^{1-\varepsilon(r)}$. Thus, the first round (line 8) makes at most

$$\ell \cdot \binom{t}{2} < \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} \cdot 8n^{2\varepsilon(r)} k^{2(1-\varepsilon(r))} \leq 4n^{1+\varepsilon(r)} k^{1-\varepsilon(r)} \tag{4}$$

queries. Then, the resulting set R defined in line (11) is of size $|R| \leq k \cdot \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} = \frac{1}{2} \cdot n^{1-\varepsilon(r)} k^{\varepsilon(r)}$. By induction, the recursive call to $\mathsf{LR\text{-}SameSetQuery}(R, r-1)$ in line (11) then costs

$$8\left(\frac{1}{2} \cdot n^{1-\varepsilon(r)} k^{\varepsilon(r)}\right)^{1+\varepsilon(r-1)} k^{1-\varepsilon(r-1)} \leq 4n^{(1-\varepsilon(r))(1+\varepsilon(r-1))} k^{\varepsilon(r)(1+\varepsilon(r-1))+(1-\varepsilon(r-1))} \tag{5}$$

queries at most. To understand the exponents in the RHS we need the following claim about $\varepsilon(r)$.

Claim 11 *The following hold for all $r \geq 2$: 1. $(1 - \varepsilon(r))(1 + \varepsilon(r-1)) = 1 + \varepsilon(r)$.*
2. $\varepsilon(r)(1 + \varepsilon(r-1)) + (1 - \varepsilon(r-1)) = 1 - \varepsilon(r)$.

Proof To see that item (1) holds, observe that

$$(1 - \varepsilon(r))(1 + \varepsilon(r-1)) = \frac{2^r - 2}{2^r - 1} \cdot \frac{2^{r-1}}{2^{r-1} - 1} = 2 \cdot \frac{2^{r-1} - 1}{2^r - 1} \cdot \frac{2^{r-1}}{2^{r-1} - 1} = \frac{2^r}{2^r - 1} = 1 + \varepsilon(r).$$

To see that item (2) holds, observe that the statement is equivalent to the identity $\varepsilon(r) = \frac{\varepsilon(r-1)}{2+\varepsilon(r-1)}$. This identity holds since

$$\frac{\varepsilon(r-1)}{2+\varepsilon(r-1)} = \frac{1}{(2^{r-1}-1)(2+\frac{1}{2^{r-1}-1})} = \frac{1}{2(2^{r-1}-1)+1} = \frac{1}{2^r-1} = \varepsilon(r) \tag{6}$$

and this completes the proof. ■

Now, by Claim 11 the RHS of eq. (5) is equal to $4n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}$. Combining this with the bound eq. (4) on the number of queries in the first round shows that $\mathsf{LR\text{-}SameSetQuery}$ makes at most $8n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}$ queries. This completes the proof of Theorem 1. ■

5. Weak Subset Queries

In this section we provide a nearly-optimal non-adaptive algorithm using weak (and strong, see Theorem 3) subset queries. We then design a nearly-optimal r-round algorithm following the recursive algorithmic template for r-rounds described in Algorithm 1, with weak and strong subset queries (Appendices C and D), where the trivial all-pair-query subroutine is replaced by the best non-adaptive algorithm for the respective query type.

Theorem 12 (Weak Subset Query Non-adaptive Algorithm) *There is a non-adaptive algorithm which, for any query size bound $2 \leq s \leq \sqrt{n}$ and error probability $\delta > 0$, learns an arbitrary partition on n elements exactly using*

$$O\left(\frac{n^2}{s^2}\log(n/\delta) + n\log^4(n/\delta)\log s\right)$$

weak subset queries of size at most s, and succeeds with probability $1-\delta$. Observe that if $s \leq O(\frac{\sqrt{n}}{\log^2(n/\delta)})$, then the query complexity becomes $O(\frac{n^2}{s^2}\log(n/\delta))$.

Due to space constraints, we provide an overview of our proof in Section 5.1 and defer the full detailed proof to Appendix B. Pseudocode for the algorithm is given in Alg. 2 in the appendix.

5.1. Proof Overview of Theorem 12

The algorithm proceeds by iteratively learning the sets of the partition from largest to smallest. This is divided into three phases in which we learn the "large" sets (size at least $n/\mathrm{poly}\log(n/\delta)$, lines 4-9), then the "medium" sets (size at most $n/\mathrm{poly}\log(n/\delta)$ and at least n/s^2, lines 10-22), and finally the "small" sets (size at most n/s^2, lines 23-25). The algorithm maintains $\widetilde{C}$ containing the sets learned so far. To learn the medium and small sets it always exploits the known larger sets in $\widetilde{C}$ to perform the reconstruction. Note however that the queries made by the algorithm never depend on $\widetilde{C}$. In particular, all queries can be made in advance before any reconstruction is performed and so the algorithm is indeed non-adaptive.[4]

4. One could alternatively present the pseudocode of Alg. 2 so that all queries are made first before any reconstruction, making the non-adaptivity of the algorithm more transparent. However, we believe that presenting the query-selection and reconstruction process together makes the pseudocode much more intuitive and readable.

The large sets are easiest to learn: sample a random set R which is large enough to contain a representative from every large set with high probability, and then make all pairwise queries between R and U. The point is that one only needs $|R| \approx \operatorname{poly}\log(n/\delta)$ for this to hold. To learn the medium and small sets we exploit a subroutine LearnSparse (see Alg. 3 in the appendix) for learning partially reconstructed partitions (the subroutine is provided a collection of known sets) where every unknown set is sufficiently small. This procedure (see Lemma 13) learns all unknown sets with high probability when each unknown set has size at most c using only $\widetilde{O}(cn)$ queries of size $\sqrt{n/c}$. Note this allows us to learn the "small" sets (of size at most n/s^2) using $\widetilde{O}(n^2/s^2)$ queries of size at most s as desired.

Learning the medium-sized sets. The medium sets are too large to apply the subroutine LearnSparse directly and obtain the desired query complexity $\widetilde{O}(n^2/s^2)$. To circumvent this, the key idea is to sample smaller subsets $X \subset U$ and learn the partition restricted on each subset using LearnSparse, and then piece these solutions together. This is done in the body of the while-loop (lines 12-21) during which the iteration corresponding to value B tries to learn the remaining unknown sets of size at least n/B. To learn these sets, we sample a random "core" set R of size $\widetilde{O}(B)$ (line 14) so that with high probability R contains a representative from every unknown set of size at least n/B. Then, we sample $p = \widetilde{O}(n/B)$ random sets $X_1, \dots, X_p$ of size B (line 15). With high probability the following will hold: (i) the X_j-s cover U, (ii) R contains a representative from every unknown set of size at least n/B, and (iii) for every unknown set C (all of these are of size $\leq 2n/B$), we have $|C \cap (X_j \cup R)| \leq O(\log(n/\delta))$. By (i-ii) it suffices to learn the partition restricted on each $X_j \cup R$, and by (iii) this can be done efficiently using LearnSparse (Lemma 13). (See Fig. 3 in the appendix for an accompanying illustration.)

The subroutine for learning sparse partitions. We now describe the subroutine LearnSparse, which is described in pseudocode in Alg. 3 (in the appendix). The idea is to learn all pairs of elements which belong to *different* sets in the partition. We say that a set I is an *independent set* if every element of I belongs to a different set in the partition. Upon querying an independent set I, the oracle responds with $\mathsf{count}(I) = |I|$, and we learn that all pairs in I belong to different sets, i.e. we learn the relationship of $\approx |I|^2$ pairs. If all sets in the partition are of size at most c, then a random I of size $\approx \sqrt{n/c}$ will be an independent set with constant probability, allowing us to learn essentially $\approx n/c$ pairwise relationships per query, leading to a $\approx cn$ query algorithm, since there are at most n^2 pairs to learn in total.

However, this is not the whole story: recall that the only guarantee of Lemma 13 is that the *unknown* sets are of size at most c, while the *known* sets can be arbitrarily large. Let K denote the union of the known sets. Then, we just need to learn every pair of elements $u, v \in U \setminus K$ which belong to different unknown sets. The point is that since K is known, the oracle response to $I \setminus K$ can be simulated using the query to I (see line 7 and Fig. 4 in the appendix for an illustration). Therefore, it suffices to query I and this query does not depend on K at all. Thus, the queries selected by the subroutine can be made without knowledge of K, allowing the main algorithm (Alg. 2 in the appendix) to be implemented non-adaptively.

Acknowledgments

All authors supported by NSF TRIPODS Institute grant 2217058 (EnCORE) and NSF 2133484.

References

Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. *Advances in neural information processing systems*, 2016.

Hadley Black, Euiwoong Lee, Arya Mazumdar, and Barna Saha. Clustering with non-adaptive subset queries. In *Advances in Neural Information Processing Systems. (NeurIPS)*, 2024.

Marco Bressan, Nicolò Cesa-Bianchi, Silvio Lattanzi, and Andrea Paudice. Exact recovery of mangled clusters with same-cluster queries. *Advances in Neural Information Processing Systems*, 2020.

Deeparnab Chakrabarty and Hang Liao. Learning partitions using rank queries. In *Proceedings, Foundations of Software Technology and Theoretical Computer Science. (FSTTCS)*, 2024.

Susan B. Davidson, Sanjeev Khanna, Tova Milo, and Sudeepa Roy. Top-k and clustering with noisy comparisons. *ACM Trans. Database Syst.*, 2014.

Alberto Del Pia, Mingchen Ma, and Christos Tzamos. Clustering with queries under semi-random noise. In *Conference on Learning Theory*. PMLR, 2022.

Adela DePavia, Olga Medrano Martin del Campo, and Erasmo Tani. Optimal algorithms for learning partitions with faulty oracles. In *Advances in Neural Information Processing Systems. (NeurIPS)*, 2024.

Amber Feng, Michael Franklin, Donald Kossmann, Tim Kraska, Samuel R Madden, Sukriti Ramesh, Andrew Wang, and Reynold Xin. Crowddb: Query processing with the vldb crowd. 2011.

Michael J. Franklin, Donald Kossmann, Tim Kraska, Sukriti Ramesh, and Reynold Xin. Crowddb: answering queries with crowdsourcing. In *Proceedings of the 2011 international conference on Management of data*, pages 61–72, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0661-4.

Quanquan Gu and Jiawei Han. Towards active learning on graphs: An error bound minimization approach. In *2012 IEEE 12th International Conference on Data Mining*, pages 882–887. IEEE, 2012.

Wasim Huleihel, Arya Mazumdar, Muriel Médard, and Soumyabrata Pal. Same-cluster querying for overlapping clusters. *Advances in Neural Information Processing Systems*, 32, 2019.

Vasilis Kontonis, Mingchen Ma, and Christos Tzamos. Active learning with simple questions. In *Conference on Learning Theory (COLT)*, 2024.

Xizhi Liu and Sayan Mukherjee. Tight query complexity bounds for learning graph partitions. In *Conference on Learning Theory (COLT)*, 2022.

Arya Mazumdar and Soumyabrata Pal. Semisupervised clustering, and-queries and locally encodable source coding. *Advances in Neural Information Processing Systems*, 30, 2017.

Arya Mazumdar and Barna Saha. Clustering with noisy queries. In *Advances in Neural Information Processing Systems. (NeurIPS)*, 2017a.

Arya Mazumdar and Barna Saha. Query complexity of clustering with side information. In *Advances in Neural Information Processing Systems. (NeurIPS)*, 2017b.

Arya Mazumdar and Barna Saha. A theoretical analysis of first heuristics of crowdsourced entity resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017c.

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

Barzan Mozafari, Purnamrita Sarkar, Michael J Franklin, Michael I Jordan, and Samuel Madden. Active learning for crowd-sourced databases. *arXiv preprint arXiv:1209.3686*, 2012.

Lev Reyzin and Nikhil Srivastava. Learning and verifying graphs using queries with a focus on edge counting. In *Algorithmic Learning Theory (ALT)*, 2007.

Barna Saha and Sanjay Subramanian. Correlation clustering with same-cluster queries bounded by optimal cost. In *27th Annual European Symposium on Algorithms (ESA 2019)*. Schloss-Dagstuhl-Leibniz Zentrum für Informatik, 2019.

Jiannan Wang, Tim Kraska, Michael J. Franklin, and Jianhua Feng. Crowder: crowdsourcing entity resolution. *Proc. VLDB Endow.*, 5(11):1483–1494, July 2012. ISSN 2150-8097. doi: 10.14778/2350229.2350263.

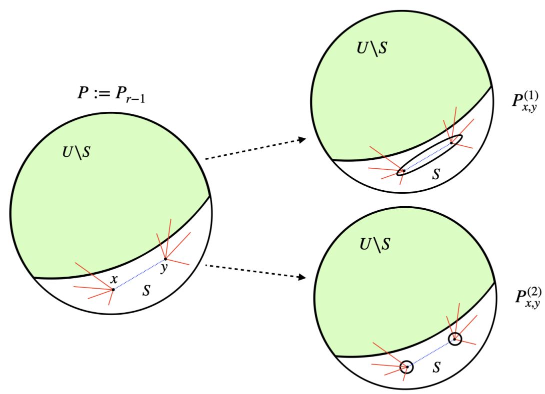

Figure 2: An illustration depicting the final pair of partitions $\mathcal{P}^{(1)}_{x,y}$ and $\mathcal{P}^{(2)}_{x,y}$ which the r-round algorithm fails to distinguish. The set S is depicted in white. Note that we have not tried to depict the partition within the sets S and $U \setminus S$. The point is that the partition $\mathcal{P} = \mathcal{P}_{r-1}$ has been defined a such a way that every queried pair $(u, v) \in Q_1 \cup \cdots \cup Q_{r-1}$ that touches S is such that u, v belong to different sets in $\mathcal{P}$. In particular, any queried pair that touches x or y is given query response "no" under $\mathcal{P}$, which is consistent with the partitions $\mathcal{P}^{(1)}_{x,y}$ and $\mathcal{P}^{(2)}_{x,y}$. Thus, these final partitions are well-defined and allow us to the lower bound the final round of queries, Q_r.

Appendix A. Deferred Proofs from Section 3

A.1. Proof of Turán's Theorem

Proof of Theorem 7. Pick a random ordering π on the vertices and construct an independent set I_π greedily according to π. I.e., iterating over $i \in [n]$, the $\pi(i)$-th vertex is added to I_π iff it has no neighbors in I_π. Then, v appears in I_π if it comes before all of its neighbors in π, which occurs with probability exactly $1/(1 + d_G(v))$. Thus,

$$\mathbb{E}_\pi[|I_\pi|] = \sum_{v \in V} \frac{1}{1 + d_G(v)} = n \cdot \mathbb{E}_{v \in V}\left[\frac{1}{1 + d_G(v)}\right]. \tag{7}$$

The function $1/(1 + x)$ for $x \geq 0$ is convex and so by Jensen's inequality

$$\mathbb{E}_{v \in V}\left[\frac{1}{1 + d_G(v)}\right] \geq \frac{1}{1 + \mathbb{E}_{v \in V}[d_G(v)]} = \frac{1}{1 + d_G}$$

and this completes the proof. ■

A.2. Proof of the Repeated Turán's Theorem

Proof of Corollary 8. The average degree in G is $2m/n$, and so by Turán's Theorem 7, it contains an independent set of size at least $\frac{n}{(2m/n)+1} \geq \frac{n^2}{4m}$, where here we have used $m \geq n$. Thus, we can take S_1 to be an independent set of size $N \leq \frac{n^2}{8m}$. Now, suppose we have constructed $1 \leq t < \ell$ disjoint independent sets $S_1, \ldots, S_t$, each of size N. Let $G_t = G[U \setminus (S_1 \cup \cdots \cup S_t)]$ denote the induced subgraph obtained by removing these independent sets. The number of edges in G_t is clearly still at most m and the number of vertices is at least

$$n - tN \geq n - \ell N \geq n/2$$

since $t < \ell \leq n/2N$. Thus, the average degree in G_t is at most $2m/(n/2) \leq 4m/n$, and again by Turán's Theorem 7 we get an independent set in G_t of size at least $\frac{n}{(4m/n)+1} \geq \frac{n^2}{8m}$ and in particular we can take S_{t+1} to be an independent set of size N. This completes the proof. ■

A.3. Proof of the Carving Lemma 9

First, we apply the repeated Turán Theorem (Corollary 8) in $G(S, Q \cup Q')$ to obtain the independent sets $S'_1, \ldots, S'_{\ell'}$ each of size N and let S' denote their union. Note that this is possible by the assumed bounds on N and ℓ'. We now define the partition $\mathcal{P}'$ as follows: (a) each $S'_i \in \mathcal{P}'$ is a set of the partition, and (b) for each $X \in \mathcal{P}$, we make $X \setminus S' \in \mathcal{P}'$ a set in the partition. Note that $\mathcal{P}'$ clearly is a partition of U, $|\mathcal{P}'| \leq |\mathcal{P}| + \ell'$, and item (1) holds by construction.

We now prove that item (2) holds. Consider any $\{x, y\} \in Q$. We break into cases depending on where x, y lie. Note that by our assumption about S and $\mathcal{P}$ and the construction of $\mathcal{P}'$, for every set $X \in \mathcal{P} \setminus \{S_1, \ldots, S_\ell\}$, we also have $X \in \mathcal{P}'$, i.e. these sets are preserved. Thus, if $x, y \in U \setminus S$, then clearly $\textsf{same-set}(x, y, \mathcal{P}) = \textsf{same-set}(x, y, \mathcal{P}')$. This also implies that if exactly one of x or y lie in $U \setminus S$, then $\textsf{same-set}(x, y, \mathcal{P})] = \textsf{same-set}(x, y, \mathcal{P}') = \textsf{no}$.

The remaining case to consider is when both $x, y \in S$. Now recall that $S = S_1 \sqcup \cdots \sqcup S_\ell$ where each S_i is an independent set in $G(U, Q)$. In particular, this means that x, y lie in *different* S_i-s. Let $x \in S_i, y \in S_j$ where $i \neq j$. This means that $\textsf{same-set}(x, y, \mathcal{P}) = \textsf{no}$. Now, note that $S_i \setminus S' \in \mathcal{P}'$ and $S_j \setminus S' \in \mathcal{P}'$ where (a) $S' = S'_1 \sqcup \cdots \sqcup S'_{\ell'}$, (b) each S'_i is an independent set in $G(S, Q \cup Q')$, and (c) each $S'_i \in \mathcal{P}'$ is a set in $\mathcal{P}'$. In particular, if both, or exactly one, of x, y lie in $S \setminus S'$, then we clearly have $\textsf{same-set}(x, y, \mathcal{P}') = \textsf{no}$. Finally, if both $x, y \in S'$, then by (b) we must have that x, y are in different S'_i-s, implying that $\textsf{same-set}(x, y, \mathcal{P}')] = \textsf{no}$. This completes the proof.

Appendix B. Non-adaptive Weak Subset Query Algorithm: Proof of Theorem 12

Let $\mathcal{C} = (C_1, \ldots, C_k)$ denote the hidden partition on U. For each $B \in [1, n]$, let $\mathcal{C}_B = \{C \in \mathcal{C} \colon |C| \geq n/B\}$ denote the collection of sets in $\mathcal{C}$ of size at least n/B. Our main subroutine will be from the following lemma, which we prove in Section B.1.

Lemma 13 (Subroutine for learning a sparse, partially known partition) *Let $\mathcal{C}$ be a hidden partition over a universe U on n points. Suppose that a subset of the partition $\mathcal{K} \subseteq \mathcal{C}$ is completely known and let $K = \bigcup_{C \in \mathcal{K}} C$. There is a non-adaptive procedure with subset query access to $\mathcal{C}$,* $\textsf{LearnSparse}(\mathcal{K}, c, \delta)$ *which makes $2cn \ln(n^2/\delta)$ queries of size $\lfloor \sqrt{n/c} \rfloor$ and returns a graph G on $U \setminus K$ whose connected components are exactly the set of unknown sets $\mathcal{C} \setminus \mathcal{K}$ with probability $1 - \delta$, if every unknown set $C \in \mathcal{C} \setminus \mathcal{K}$ satisfies $|C| \leq c$. Moreover, the queries made by the procedure do not depend on the known sets, $\mathcal{K}$.*

Pseudocode for our algorithm is given in Alg. 2. The algorithm works by maintaining a set $\widetilde{\mathcal{C}}$, which at any point in the algorithm's execution will be equal to $\mathcal{C}_B$ for some $B \in [1, n]$ with high probability, as we will show. In particular, when the algorithm terminates, we will have $\widetilde{\mathcal{C}} = \mathcal{C}_n$

with probability $1 - \delta$. In lines (6-9) we employ a simple strategy to learn all of the sufficiently large sets.

Algorithm 2: $\mathsf{NA\text{-}WeakSubsetQuery}(n, k, s, \delta)$

---

1 **Input:** Subset query access to a hidden partition $\mathcal{C}$ of $|U| = n$ points into at most k sets. An error probability $\delta > 0$;
2 **Output:** A partition $\widetilde{\mathcal{C}}$ of U which is equal to $\mathcal{C}$ with probability $1 - \delta$;
3 Initialize hypothesis partition $\widetilde{\mathcal{C}} \leftarrow \emptyset$;
4 *(Learn the large sets)*;
5 Sample a set $R \subset U$ of $\ln^2(n/\delta)\ln(k/\delta)$ i.i.d. uniform random elements;
6 **for** $x \in R$, $y \in U$ **do**
7   **Query** $\{x, y\}$ and let $C_x = \{y \in U : \mathsf{count}(\{x, y\}) = 1\}$ denote the set containing x;
8   **If** $|C_x| \geq \frac{n}{\ln^2(n/\delta)}$, then $\widetilde{\mathcal{C}} \leftarrow \widetilde{\mathcal{C}} \cup \{C_x\}$;
9 **end**
10 $\backslash\backslash \triangleright$ $\widetilde{\mathcal{C}}$ now contains all sets in the partition of size at least $\frac{n}{\ln^2(n/\delta)}$, with probability $1 - \delta$. $\backslash\backslash$
11 *(Learn the medium sets)*;
12 $B \leftarrow 2\ln^2(n/\delta))$;
13 **while** $B \leq s^2$ **do**
14   *(Learn the unknown sets of size at least $\frac{n}{B}$)*;
15   Sample a set $R \subset U$ of $B\ln(6B\log(s^2)/\delta)$ i.i.d. uniform random elements;
16   Sample $p = \frac{n}{B}\ln(6n\log(s^2)/\delta)$ sets $X_1, \ldots, X_p$ each of B i.i.d. uniform random elements;
17   **for** $j \in [p]$ **do**
18   Let $\mathcal{K}_j = \{C \cap (X_j \cup R) : C \in \widetilde{C}\}$ denote the current known sets in partition restricted on $X_j \cup R$;
19   Run $\mathsf{LearnSparse}(\mathcal{K}_j, 72\ln(n/\delta), \delta/(2p\log(s^2)))$ on $X_j \cup R$ and let G_j be the returned partition-graph on $(X_j \cup R) \setminus \bigcup_{K \in \mathcal{K}_j} K$;
20         $\backslash\backslash \triangleright$ Note that the queries made by $\mathsf{LearnSparse}$ do not depend on the set $\widetilde{\mathcal{C}}$. $\backslash\backslash$
21   **end**
22   Let $\widetilde{C}_1, \ldots, \widetilde{C}_\ell$ denote the connected components of the union $G = G_1 \cup \cdots \cup G_p$;
23   Update $\widetilde{\mathcal{C}} \leftarrow \widetilde{\mathcal{C}} \cup \{\widetilde{C}_j : j \in [\ell], |\widetilde{C}_j| \geq n/B\}$ and $B \leftarrow 2B$;
24 **end**
25 $\backslash\backslash \triangleright$ $\widetilde{\mathcal{C}}$ now contains all sets in the partition of size at least $\frac{2n}{s^2}$ with probability $1 - 2\delta$. $\backslash\backslash$
26 *(Learn the small sets)*;
27 Run $\mathsf{LearnSparse}(\widetilde{C}, 2n/s^2, \delta)$ and let G denote the returned partition-graph on $U \setminus \bigcup_{C \in \widetilde{\mathcal{C}}} C$;
28 $\backslash\backslash \triangleright$ Note that the queries made by $\mathsf{LearnSparse}$ do not depend on the set $\widetilde{\mathcal{C}}$. $\backslash\backslash$
29 Add the connected components of G to $\widetilde{\mathcal{C}}$;
30 $\backslash\backslash \triangleright$ At this stage $\widetilde{\mathcal{C}}$ is exactly equal to $\mathcal{C}$ with probability $1 - 3\delta$. $\backslash\backslash$
31 **Return** $\widetilde{\mathcal{C}}$;

Claim 14 *After line (9) of Alg. 2, we have* $\widetilde{\mathcal{C}} = \mathcal{C}_{n/\ln^2(n/\delta)}$ *with probability* $1-\delta$*. Moreover, the number of queries made in line (7) is at most* $n\ln^2(n/\delta)\ln(k/\delta)$ *and every query is of size* 2*.*

Proof The number of queries made in line (7) is exactly $|U|\cdot|R| = n\ln^2(n/\delta)\ln(k/\delta)$, as claimed. Observe that in line (7), the set C_x is exactly the set in $\mathcal{C}$ which contains the point x. Thus, the claim holds as long as $R\cap C \neq \emptyset$ for every $C \in \mathcal{C}_{n/\ln^2(n/\delta)}$. For such a fixed C, we have

$$\Pr_R[R\cap C=\emptyset] = \left(1-\frac{|C|}{n}\right)^{|R|} \leq \left(1-\frac{1}{\ln^2(n/\delta)}\right)^{\ln^2(n/\delta)\ln(k/\delta)} \leq \delta/k$$

and so the claim holds by a union bound over the at most k sets in $\mathcal{C}_{n/\ln^2(n/\delta)}$. ∎

Next, we employ a different strategy in lines (13-23) to learn the sets with sizes in $[\frac{2n}{s^2}, \frac{n}{\ln^2(n/\delta)}]$. This is the most involved phase of the algorithm.

Lemma 15 *If* $\widetilde{\mathcal{C}} = \mathcal{C}_{n/\ln^2(n/\delta)}$ *in line (9) of Alg. 2, then in line (23), we have* $\widetilde{\mathcal{C}} = \mathcal{C}_{2n/s^2}$ *with probability* $1-\delta$*. Moreover, the number of queries made by* LearnSparse *in line (19) is* $O(n\ln^4(n/\delta)\ln s)$ *and every query is of size* s*.*

Proof Note that if $s^2 \leq 2\ln^2(n/\delta)$, then the lemma is vacuously correct. We will prove the lemma by an induction on each setting of B during the while loop in lines (13-23), where $2\ln^2(n/\delta) \leq B \leq s^2$ and B doubles after each iteration, in line (21). The following claim proves correctness and the desired query complexity for each iteration, and Lemma 15 then follows immediately since there are at most $\log(s^2)$ iterations in total.

Claim 16 *Consider an arbitrary iteration of the while loop beginning in line (13). If in line (13) it holds that* $\widetilde{\mathcal{C}} = \mathcal{C}_{B/2}$*, then in line (22), we have* $\widetilde{\mathcal{C}} = \mathcal{C}_B$ *with probability* $1-\frac{\delta}{\log(s^2)}$*. Moreover, during this iteration the number of queries made by the calls to* LearnSparse *in line (19) is* $O(n\ln^4(n/\delta))$*, and every query is of size at most* s*.*

Proof First, we define the following good events. We will argue that conditioned on these events, we will have $\widetilde{\mathcal{C}} = \mathcal{C}_B$ with probability $1-\frac{\delta}{2\log(s^2)}$. We will then argue that with probability $1-\frac{\delta}{2\log(s^2)}$ all the events occur, and this will prove the claim by a union bound.

- Let $\mathcal{E}_{R,\mathsf{cover}}$ denote the event that $R\cap C\neq\emptyset$ for every $C\in\mathcal{C}_B$.
- Let $\mathcal{E}_{X,\mathsf{cover}}$ denote the event that $X_1\cup\cdots\cup X_p = U$.
- Let $\mathcal{E}_{\mathsf{sparse}}$ denote the event that $|(X_j\cup R)\cap C| \leq 72\ln(n/\delta)$ for every $C\in\mathcal{C}\setminus\mathcal{C}_{B/2}$ and every $j\in[p]$.

Recall we are conditioning on $\widetilde{\mathcal{C}} = \mathcal{C}_{B/2}$ and so in line (18) $\mathcal{K}_j$ is exactly this collection of sets in the partition restricted on the set $X_j\cup R$. If the call to LearnSparse on $X_j\cup R$ in line (19) succeeds, then the connected components of the returned graph G_j are exactly $X_j\cap C$ for each $C\in\mathcal{C}_B$. Moreover, if $\mathcal{E}_{R,\mathsf{cover}}$ occurs, then the connected components of the union $G = G_1\cup\cdots\cup G_p$ are exactly $(X_1\cup\cdots\cup X_p)\cap C$ for each $C\in\mathcal{C}_B$. This shows that if $\mathcal{E}_{X,\mathsf{cover}}$ occurs, then the set of connected components of G is exactly $\mathcal{C}_B$ and we get $\widetilde{\mathcal{C}} = \mathcal{C}_B$ as desired in line (22). Finally, conditioned on $\mathcal{E}_{\mathsf{sparse}}$, Lemma 13 guarantees that each call to LearnSparse in line (18) succeeds with probability $1 - \delta/(2p\log(s^2))$, and so by a union bound over the p calls, all of them succeed with probability $1 - \delta/2\log(s^2)$ and we have $\widetilde{\mathcal{C}} = \mathcal{C}_B$ as argued.

Now, to complete the proof it suffices to show that each good event $\mathcal{E}_{R,\mathsf{cover}}$, $\mathcal{E}_{X,\mathsf{cover}}$, and $\mathcal{E}_{\mathsf{sparse}}$ occurs with probability at least $1 - \delta/6\log(s^2)$.

Recall from line (14) that R is a set of $B\ln(6B\log(s^2)/\delta)$ i.i.d. uniform random elements. Thus, observe that for $C \in \mathcal{C}_B$,

$$\Pr_R[R \cap C = \emptyset] \leq \left(1 - \frac{1}{B}\right)^{|R|} \leq \delta/(6B\log(s^2)) \implies \Pr_R[\mathcal{E}_{R,\mathsf{cover}}] \geq 1 - \delta/(6\log(s^2)) \quad (8)$$

by a union bound since there can be at most B sets in $\mathcal{C}_B$.

Now, recalling the definition of p and $X_1, \ldots, X_p$ in line (15), observe that $X = X_1 \cup \cdots \cup X_p$ is simply a set of $n\ln(6n\log(s^2)/\delta)$ i.i.d. uniform random samples from U. Thus, the probability that $x \notin X$ for a fixed $x \in U$ is at most $(1 - 1/n)^{|X|} \leq \delta/(6n\log(s^2))$ and so by a union bound over all $x \in U$, we have

$$\Pr_X[\mathcal{E}_{X,\mathsf{cover}}] \geq 1 - \delta/(6\log(s^2)). \quad (9)$$

We now lower bound the probability of $\mathcal{E}_{\mathsf{sparse}}$. First, $X_j \cup R$ is simply a set of

$$|R \cup X_j| \leq |R| + |X_j| \leq 2B\ln(6B\log(s^2)/\delta) \leq 4B\ln(n/\delta) \quad (10)$$

independent uniform random elements[5]. Fix a set $C \in \mathcal{C} \setminus \mathcal{C}_{B/2}$, i.e. C is of size at most $2n/B$. Thus, a uniform random element lands in C with probability at most $2/B$, and so

$$\Pr_{R,X_j}[|(X_j \cup R) \cap C| > t] \leq \binom{4B\ln(n/\delta)}{t}\left(\frac{2}{B}\right)^t \leq \left(\frac{4eB\ln(n/\delta)}{t} \cdot \frac{2}{B}\right)^t \leq \left(\frac{24\ln(n/\delta)}{t}\right)^t$$

where the first inequality is by taking a union bound over each subset of t random elements from $X_j \cup R$ and considering the probability that they all land in C. The second inequality follows from the inequality $\binom{n}{t} \leq (\frac{en}{k})^k$. Thus, if we set $t = 24e\ln(n/\delta) < 72\ln(n/\delta)$ we get $\Pr_{R,X_j}[|(X_j \cup R) \cap C| > t] \leq (\delta/n)^{24}$, and so (recalling the definition of $\mathcal{E}_{\mathsf{sparse}}$) clearly by a union bound over every $C \in \mathcal{C} \setminus \mathcal{C}_{B/2}$ and every $j \in [p]$, we have

$$\Pr_{R,X_1,\ldots,X_p}[\mathcal{E}_{\mathsf{sparse}}] \geq 1 - \frac{\delta}{6\log(s^2)} \quad (11)$$

as desired. Thus, the correctness follows from eq. (8), eq. (9), and eq. (11).

For the query complexity, each call to LearnSparse in line (19) is on a set of size at most $4B\ln(n/\delta)$ (recall eq. (10)) where each set is of size at most $72\ln(n/\delta)$. Thus, by Lemma 13, the number of queries made in line (19) is $O(B\ln^3(n/\delta))$ and each query is of size at most

$$\sqrt{\frac{4B\ln(n/\delta)}{72\ln(n/\delta)}} = \sqrt{B/18} < s$$

5. This inequality holds since $B \leq s^2 \leq n$ and so the argument of the log is $6B\log(s^2)/\delta \leq n^2/\delta < (n/\delta)^2$.

Since $\mathsf{LearnSparse}$ is called $p = O(\frac{n}{B}\ln(n/\delta))$ times during an iteration, the total number of such queries in an iteration is $O(n\ln^4(n/\delta))$. ■

This completes the proof of Lemma 15. ■

Finally, we show that the sets of size at most $2n/s^2$ are learned easily in the final stage of the algorithm.

Claim 17 *If $\widetilde{\mathcal{C}} = \mathcal{C}_{2n/s^2}$ in line (23), then at the end of the algorithm's execution we have $\widetilde{\mathcal{C}} = \mathcal{C}$ with probability $1-\delta$. Moreover, the number of queries made in line (24) is $O(\frac{n^2}{s^2}\ln(n/\delta))$ and every query is of size at most s.*

Proof The proof follows immediately from Lemma 13. We are conditioning on $\widetilde{\mathcal{C}} = \mathcal{C}_{2n/s^2}$ and so every unknown set is of size at most $2n/s^2$. Thus, by Lemma 13 the number of queries made by $\mathsf{LearnSparse}$ in line (24) is $O(\frac{n^2}{s^2}\ln(n/\delta))$ and every query is of size at most $\sqrt{n/(2n/s^2)} \leq s$. ■

Finally, combining Claim 14, Lemma 15, and Claim 17, at the end of the algorithm's execution we have $\widetilde{\mathcal{C}} = \mathcal{C}$ with probability $1-3\delta$, and this completes the proof of Theorem 12.

B.1. Learning a Sparse, Partially Known Partition: Proof of Lemma 13

Pseudocode for the algorithm is given in Alg. 3. Let $K = \bigcup_{C\in\mathcal{K}} C$ denote the set of points belonging to a known set.

Algorithm 3: $\mathsf{LearnSparse}(\mathcal{K}, c, \delta)$

---

1 **Input:** (1) Subset query access to hidden partition $\mathcal{C}$ over U with n points. (2) Known sets $\mathcal{K} \subseteq \mathcal{C}$ with promise that $|C| \leq c$ for every $C \in \mathcal{C}\setminus\mathcal{K}$. (3) Error probability $\delta > 0$;

2 **Output:** A partition $\widetilde{\mathcal{C}}$ of U, which is equal to $\mathcal{C}$ with probability $1-\delta$;

3 Initialize $E \leftarrow \emptyset$ and let $K = \bigcup_{C\in\mathcal{K}} C$ denote the union of known sets;

4 **Repeat** $2cn\ln(n^2/\delta)$ times:

5 $\longrightarrow$ Let I be a uniform random subset of U of size $\lfloor\sqrt{n/c}\rfloor$;

6 $\longrightarrow$ Let $c_{I,K} = \mathsf{count}(I \cap K)$ which we can compute without making any queries since the partition $\mathcal{K}$ of K is known;

7 $\longrightarrow$ **Query** I and if $\mathsf{count}(I) - c_{I,K} = |I \setminus K|$ (i.e., $I\setminus K$ is an independent set), then $E \leftarrow E \cup \binom{I\setminus K}{2}$;

8 \\ ▷ Note that here we use the full power of the count query. In particular, we are not simply simulating independent set queries. This appears to be crucial to achieve non-adaptivity. \\

9 \\ ▷ Note that all queries can be specified without any knowledge of $\mathcal{K}$. In particular, this allows $\mathsf{NA\text{-}WeakSubsetQuery}$ to be implemented non-adaptively. \\

10 **Return** the graph $G(U\setminus K, \overline{E})$;

The claimed query complexity and query size bound are obvious by definition of the algorithm. We now show that the connected components of the graph returned in line (8) correspond exactly with the collection of unknown sets $\mathcal{C}\setminus\mathcal{K}$ with probability $1-\delta$.

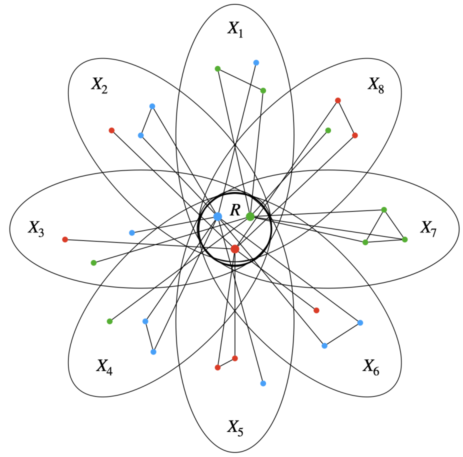

Figure 3: An illustration depicting how Alg. 2 learns the “medium”-sized sets. The sets of the partition are represented by the three colors (here $k = 3$). The random core set R (chosen in line 14) is large enough so that with high probability it contains a representative from every $C \in \mathcal{C}_B$ which hasn't yet been learned. Then, we use Alg. 3 to learn the partition restricted on $X_j \cup R$ for every X_j (chosen in line 15). The key is that these sets are small enough so that every unlearned set in the partition restricted on $X_j \cup R$ will be very small, allowing us to learn this restricted partition with few queries using Alg. 3. Then, since the X_j-s cover U, we recover all unlearned sets in $\mathcal{C}_B$ in lines 20-21 by taking all connected components of size at least n/B.

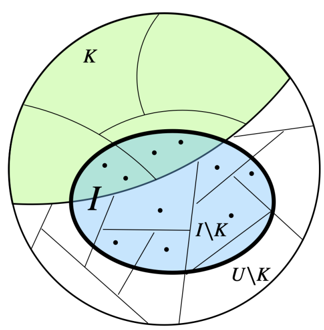

Figure 4: An illustration depicting the key idea of Alg. 3 (LearnSparse). The green region represents the points belonging to known sets in the partition (the larger sets) and the white regions represent the sets we are trying to learn. The queried set I (pictured in blue) is drawn in line 5. Since the partition on the green region K is known, the reconstruction process can simulate the oracle on the set $I \setminus K$ using the oracle's response to the set I (line 6-7). In particular, the algorithm deduces that $I \setminus K$ is an independent set (line 7), even though I itself is not, and can use this information to recover the unknown sets of the partition.

Let us call a pair of points $x, y \in U \setminus K$ *separated* if they belong to different, unknown sets. If a separated pair is contained in some I for which $I \setminus K$ is an independent set (refer to Alg. 3 line (5)), then we learn that x, y do not belong to the same set (i.e. we learn that this is a non-edge of the partition-graph over $U \setminus K$). When this occurs, we record that x, y is a non-edge by adding it to E in line (7). Therefore, if we learn this for *every separated pair*, then we can recover the partition-graph. In particular, at the end of the algorithm's execution E will be exactly the set of non-edges, and so $G(U \setminus K, \overline{E})$ will be partition-graph on $U \setminus K$. Thus, it suffices to show that every separated pair is contained in some I for which $I \setminus K$ is an independent set. Importantly, to tell if $I \setminus K$ is an independent set it suffices to only query I by exploiting the already known partition on K (line 7).

Let $\ell = 2cn \ln(n^2/\delta)$ be the total number of I-s that are sampled in line (5), and let them be denoted by $I_1, \ldots, I_\ell$. For each separated pair x, y and for each $i \in [\ell]$, let $\mathcal{E}_{i,x,y}$ denote the event that $x, y \in I_i$ and I_i is an independent set (IS). We will show the following claim, which completes the proof of Lemma 13 by a union bound over all (at most n^2) separated pairs.

Claim 18 *For every separated pair x, y, we have*

$$\Pr_{I_1,\ldots,I_\ell}[\neg\mathcal{E}_{1,x,y} \wedge \cdots \wedge \neg\mathcal{E}_{\ell,x,y}] \leq \delta/n^2$$

Proof We will drop x, y from the subscript for brevity. Let I be a uniform random subset of U of size $\lfloor\sqrt{n/c}\rfloor$. We have $\Pr[\mathcal{E}_i] = \Pr[x, y \in I] \cdot \Pr[I \setminus K \text{ is an IS} \mid x, y \in I]$. Then,

$$\Pr[x, y \in I] = \frac{\binom{n-2}{|I|-2}}{\binom{n}{|I|}} = \frac{|I|(|I|-1)}{n(n-1)} \geq \frac{1}{cn}.$$

Now, recall that we are assuming every unknown set $C \in \mathcal{C} \setminus \mathcal{K}$ has size at most c and so the probability of two points (either both uniform random, or one being x or y and the other being uniform random) landing in the same unknown set is at most c/n. Thus, by a union bound over all pairs of points I, we have

$$\Pr[I \setminus K \text{ not an IS} \mid x, y \in I] \leq \binom{|I|}{2}\frac{c}{n} \leq 1/2$$

and so $\Pr[\mathcal{E}_i] \geq \frac{1}{2cn}$. Finally, since the I_i-s are independent, the $\mathcal{E}_i$ events are also independent. Thus,

$$\Pr_{I_1,\ldots,I_\ell}[\neg\mathcal{E}_{1,x,y} \wedge \cdots \wedge \neg\mathcal{E}_{\ell,x,y}] \leq \left(1 - \frac{1}{2cn}\right)^\ell \leq \delta/n^2$$

and this completes the proof. ■

Appendix C. Low-Round Weak Subset Query Algorithm

In this section, we give a nearly optimal r-round algorithm for partition learning using weak subset queries of bounded size s. We prove the following theorem.

Theorem 19 *For any $r \geq 1$, there is a randomized r-round algorithm for learning a k-partition using at most*

$$O\left(\frac{n^{\left(1+\frac{1}{2^r-1}\right)}k^{\left(1-\frac{1}{2^r-1}\right)}}{s^2} \cdot \log^5(n/\delta)\right)$$

weak subset queries of size at most $2 \leq s \leq \sqrt{n^{\left(\frac{1}{2^r-1}\right)}k^{\left(1-\frac{1}{2^r-1}\right)}}$ that succeeds with probability $1-\delta$ for any $\delta > 0$.

In particular, with $O(\log\log n)$ rounds and query-size bound $\sqrt{k}$ our algorithm has query complexity $\widetilde{O}(n)$. In general, with $O(\log\log n)$ rounds and $s \leq \sqrt{k}$, we get $\widetilde{O}(\frac{nk}{s^2})$ queries, matching the $\Omega(\frac{nk}{s^2})$ fully adaptive lower bound up to $\operatorname{poly}\log n$ factors. Our main building block is the nearly optimal non-adaptive algorithm of Theorem 12. Our r-round algorithm is obtained by combining the non-adaptive algorithm with the recursive approach used to obtain an optimal r-round pair-query algorithm in Section 4. For simplicity of presentation, we will use the following slightly weaker statement of Theorem 12. (Note that in some cases the log-factors can be improved in the non-adaptive algorithm and this also leads to the same improvements in the resulting r-round algorithm.)

Theorem 20 (Corollary of Theorem 12) *There is a non-adaptive algorithm,* NA-WeakSubsetQuery, *which for any query size bound $2 \leq s \leq \sqrt{n}$ and error probability $\delta > 0$, learns an arbitrary partition on n elements exactly with probability $1-\delta$ using at most $\frac{n^2}{s^2} \cdot C\log^5(n/\delta)$ weak subset queries of size at most s, where $C > 0$ is a universal constant.*

Proof of Theorem 19. For shorthand in the proof, we will use $\varepsilon(r) = \frac{1}{2^r-1}$ for all $r \geq 1$. We will use the non-adaptive s-bounded query algorithm NA-WeakSubsetQuery of Theorem 20, which uses at most

$$\frac{n^2}{s^2} \cdot C\log^5(n/\delta) \tag{12}$$

queries of size at most s where $s \leq [2, \sqrt{n}]$, and succeeds with probability $1-\delta$. The algorithm is recursive and pseudocode is given in Alg. 4.

Query complexity. We prove by induction on r that Alg. 4 uses at most

$$\frac{n^{\left(1+\frac{1}{2^r-1}\right)}k^{\left(1-\frac{1}{2^r-1}\right)}}{s^2} \cdot 16C\log^5(n/\delta) \tag{13}$$

weak subset queries of size at most s. For the base case of $r = 1$, LR-WeakSubsetQuery simply runs NA-WeakSubsetQuery, and so the base case is correct by Theorem 20.

Now, suppose $r \geq 2$. First, if $n \leq 16k$, then the algorithm simply runs NA-WeakSubsetQuery on U using query size s (line 3), for a total of

$$\frac{n^2}{s^2} \cdot C\log^5(n/\delta) \leq \frac{nk}{s^2} \cdot 16C\log^5(n/\delta) \leq \frac{n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \cdot 16C\log^5(n/\delta)$$

queries, where the last step simply used $n \geq k$. Note that in this case the algorithm is non-adaptive. This completes the proof for the case of $n \leq 16k$.

Algorithm 4: $\textsf{LR-WeakSubsetQuery}(U, s, \delta, r)$

Input: Subset query access (on subsets of size at most s) to hidden partition $\mathcal{C}$ over U with n points and $|\mathcal{C}| \leq k$. An allowed error probability δ, and an allowed number of rounds r. Let $\varepsilon(r) := \frac{1}{2^r - 1}$;

if $r = 1$ ***or*** $n \leq 16k$ **then**

  Run $\textsf{NA-WeakSubsetQuery}(U, s, \delta)$ and output the returned partition;

end

else

  Partition the n points of U arbitrarily into $\ell = \lceil \frac{1}{3} \left(\frac{n}{k}\right)^{1-\varepsilon(r)} \rceil$ sets $U_1, \dots, U_\ell$ each of size $|U_i| \leq t := \lceil 3 n^{\varepsilon(r)} k^{1-\varepsilon(r)} \rceil$;

  for $i \in [\ell]$ **do**

    Run $\textsf{NA-WeakSubsetQuery}(U_i, s, \delta)$ and let $\widetilde{\mathcal{C}}_i$ be the returned partition;

    \\ ▷ Note that $s \leq \sqrt{|U_i|}$ and so this call to NA-WeakSubsetQuery is valid. \\

    \\ ▷ Note that $\widetilde{\mathcal{C}}_i = \{C \cap U_i \colon C \in \mathcal{C}\}$ with probability $1 - \delta$. \\

    Form R_i by taking exactly one representative from each set $C \in \widetilde{\mathcal{C}}_i$;

  end

  Set $R = R_1 \cup \cdots \cup R_\ell$ and $s' = \min(s, \sqrt{|R|^{\varepsilon(r-1)} k^{1-\varepsilon(r-1)}})$ (this is so that the recursive call to LR-WeakSubsetQuery has a valid query-size bound);

  Recursively call $\textsf{LR-WeakSubsetQuery}(R, s', \delta, r-1)$ and let $\widetilde{\mathcal{C}}_R$ be the returned partition of R;

  Return the partition $\{\bigcup_{C' \in \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_\ell \colon C' \cap C \neq \emptyset} C' \colon C \in \widetilde{\mathcal{C}}_R\}$;

end

Now suppose $n > 16k$. Since $r \geq 2$, note that $1 - \varepsilon(r) \geq 1 - \varepsilon(2) \geq 2/3$. Using these bounds, we have $(n/k)^{1-\varepsilon(r)} > 16^{2/3} > 6$. Recalling the definition of ℓ and t in line (6), this implies that

$$\ell \leq \frac{1}{3}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} + 1 = \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} - \frac{1}{6}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} + 1 < \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} \tag{14}$$

and clearly $t \leq 4n^{\varepsilon(r)}k^{1-\varepsilon(r)}$. Then, using the bound eq. (12) on the query complexity of NA-WeakSubsetQuery, the first round (lines 7-9) makes at most

$$\begin{aligned}\ell \cdot \frac{t^2}{s^2} C \log^5(n/\delta) &< \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} \cdot \left(\frac{4n^{\varepsilon(r)}k^{1-\varepsilon(r)}}{s}\right)^2 \cdot C \log^5(n/\delta) \\ &= \frac{n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \cdot 8C \log^5(n/\delta) \end{aligned} \tag{15}$$

queries, all of size at most s. Then, the resulting set R is of size

$$|R| \leq k \cdot \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} = \frac{1}{2} \cdot n^{1-\varepsilon(r)}k^{\varepsilon(r)}. \tag{16}$$

We now show that the total number of queries made over the remaining $r - 1$ rounds is at most $\frac{n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \cdot 8C \log^5(n/\delta)$. Combining this with eq. (15) shows that the total number of queries is at most $\frac{n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \cdot 16C \log^5(n/\delta)$, which completes the proof.

First, suppose that $s' = \sqrt{|R|^{\varepsilon(r-1)}k^{1-\varepsilon(r-1)}}$. By induction, the recursive call in line (12) costs at most

$$\begin{aligned}\frac{|R|^{1+\varepsilon(r-1)}k^{1-\varepsilon(r-1)}}{|R|^{\varepsilon(r-1)}k^{1-\varepsilon(r-1)}} \cdot 16C \log^5(n/\delta) &= |R| \cdot 16C \log^5(n/\delta) \\ &\leq 8Cn \log^5(n/\delta) \leq \frac{n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \cdot 8C \log^5(n/\delta)\end{aligned}$$

queries, where we used the bound on $|R|$ and the fact that $k \leq n$. Now, suppose $s' = s$. By induction, using the upper on $|R|$ from eq. (16), we obtain that running LR-WeakSubsetQuery$(R, s, \delta, r - 1)$ costs at most

$$\begin{aligned}&\frac{\left(\frac{1}{2} \cdot n^{1-\varepsilon(r)}k^{\varepsilon(r)}\right)^{1+\varepsilon(r-1)} k^{1-\varepsilon(r-1)}}{s^2} \cdot 16C \log^5(n/\delta) \\ &\leq \frac{n^{(1-\varepsilon(r))(1+\varepsilon(r-1))}k^{\varepsilon(r)(1+\varepsilon(r-1))+(1-\varepsilon(r-1))}}{s^2} \cdot 8C \log^5(n/\delta) \\ &= \frac{n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \cdot 8C \log^5(n/\delta)\end{aligned} \tag{17}$$

queries where the equality is by Claim 11. This completes the proof of the query complexity.

Correctness. Observe that the total number of calls to NA-WeakSubsetQuery per round in Alg. 4 is clearly upper bounded by n. Thus, by a union bound all such calls are successful with probability at least $1 - rn\delta$. We can then simply set $\delta = \delta/rn$, only increasing the query complexity by a constant factor. Now, conditioned on all calls to NA-WeakSubsetQuery being successful, the proof of correctness for LR-WeakSubsetQuery is simple and completely analogous to that of Theorem 1. This completes the proof of Theorem 19. ■

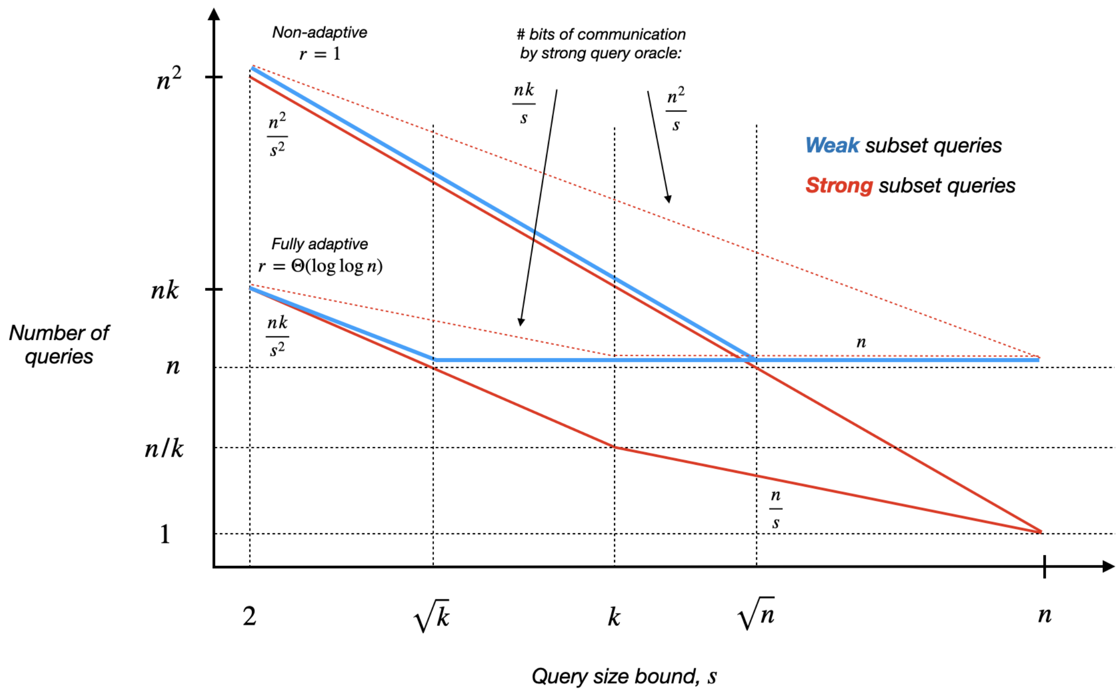

Figure 5: A comparison of weak vs. strong subset queries for non-adaptive and fully adaptive algorithms as a function of the allowed query size, s, ignoring $\operatorname{poly}\log(n)$ factors. For the purposes of the diagram we have used $k \leq \sqrt{n}$, but note in general there is no such restriction on k. Our results reveal that in the relatively small s regime ($s \leq \sqrt{n}$ for non-adaptive and $s \leq \sqrt{k}$ for adaptive), strong subset queries give no advantage over weak subset queries, which we find to be very surprising. Moreover, answering a weak query requires only $O(\log k)$ bits of communication by the oracle whereas answering a strong query requires $O(s \log k)$ bits of communication. Thus, in the weak query model the total communication follows the query complexity (the blue lines in the diagram), while in the strong query model the total communication is larger than the query complexity by a factor $O(s \log k)$ (pictured by the dotted red lines in the diagram). Thus, in terms of total communication, the weak query model is superior to the strong query model for all values of s. For clarity, we have chosen to picture the case of $r = 1$ and $r = \Theta(\log \log n)$, but for any intermediate value of r, the query complexities follow a similar shape, lying in between these two cases. In general, the query complexities in the strong and weak models are the same for s up to $\sqrt{n^{\varepsilon(r)} k^{1-\varepsilon(r)}}$, at which point the information-theoretic lower bound kicks in for weak queries. The strong query complexity continues to improve at the same rate up to $s = n^{\varepsilon(r)} k^{1-\varepsilon(r)}$, where the query complexity becomes $(n/k)^{1-\varepsilon(r)}$, after which the information theoretic lower bound of n/s kicks in.

Appendix D. Strong Subset Queries

In this section, we investigate the query complexity of learning a k-partition over $|U| = n$ elements via access to a strong subset query oracle. The algorithm is given a query size bound s and can query the oracle on any set $S \subseteq U$ of size $|S| \leq s$. For hidden partition $\mathcal{P}$, the oracle returns $(X \cap S \colon X \in \mathcal{P})$, the partition restricted on S. We first design a simple deterministic non-adaptive algorithm in Theorem 3, which is optimal among all (even randomized) non-adaptive algorithms. We then use this result to obtain an r-round deterministic algorithm in Theorem 22 which is (nearly) optimal for all r.

Theorem 21 *For any even $2 \leq s \leq n$, there is a deterministic non-adaptive algorithm for learning an arbitrary partition using at most $\frac{5n^2}{s^2}$ strong subset queries of size s.*

Proof It suffices to design a collection of sets of size at most s such that for every pair of points x, y there is some subset containing both x and y. We claim that the following construction has this property. Partition the n points into $\ell = \lceil \frac{n}{s/2} \rceil \leq \frac{3n}{s}$ sets (since $s \leq n$) $T_1, \ldots, T_\ell$ of size $|T_i| = s/2$. Then the set of queries is $T_i \cup T_j$ for every $i \neq j$ for a total of $\binom{\ell}{2} < \frac{5n^2}{s^2}$ queries of size exactly s. If x, y are in the same T_i, then clearly this pair is covered by any query involving T_i, and if $x \in T_i, y \in T_j$ for $i \neq j$, then clearly this pair is covered by the query $T_i \cup T_j$. ■

Theorem 22 *For any $r \geq 1$, there is a deterministic r-round algorithm for learning a k-partition using*

$$\frac{80n^{\left(1+\frac{1}{2^r-1}\right)}k^{\left(1-\frac{1}{2^r-1}\right)}}{s^2}$$

strong subset queries of size at most $s \leq n^{\left(\frac{1}{2^r-1}\right)}k^{\left(1-\frac{1}{2^r-1}\right)}$ where $s \geq 2$ is even.

In particular, with $O(\log\log n)$ rounds and $s \leq k$, we get $O(\frac{nk}{s^2})$ queries, matching the $\Omega(\max(\frac{nk}{s^2}, \frac{n}{s}))$ fully adaptive lower bound. With $O(\log\log n)$ rounds we get an algorithm making $O(\frac{n}{k})$ strong subset queries of size k. To design our r-round algorithm we essentially follow the same strategy presented in Section 4 for pair queries, replacing the non-adaptive subroutine with that of Theorem 3.

Proof of Theorem 22. For shorthand in the proof, we will use $\varepsilon(r) = \frac{1}{2^r-1}$ for all $r \geq 1$. We will use the non-adaptive s-bounded strong subset query algorithm NA-StrongSubsetQuery of Theorem 3. By Theorem 3, there this algorithm learns a k-partition of n elements using at most $\frac{5n^2}{s^2}$ queries of size at most s where $s \leq [2, n]$ is even.

The algorithm is recursive and pseudocode is given in Alg. 5. We prove the theorem by induction on r. For the base case of $r = 1$, LR-StrongSubsetQuery simply runs NA-StrongSubsetQuery, and correctness follows by Theorem 3.

Now suppose $r \geq 2$. First, if $n \leq 16k$, then we simply run NA-StrongSubsetQuery on U using query size s (line 3), for a total of

$$\frac{5n^2}{s^2} \leq \frac{80nk}{s^2} \leq \frac{80n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2}$$

Algorithm 5: LR-StrongSubsetQuery(U, s, r)

1 **Input:** Strong subset query access (on subsets of size at most s) to hidden partition $\mathcal{C}$ over U with n points and $|\mathcal{C}| \leq k$. An allowed number of rounds r. Let $\varepsilon(r) = \frac{1}{2^r - 1}$;
2 **if** $r = 1$ ***or*** $n \leq 16k$ **then**
3   Run NA-StrongSubsetQuery(U, s) and output the returned partition;
4 **end**
5 **else**
6   Partition the n points of U arbitrarily into $\ell = \lceil \frac{1}{3} \left(\frac{n}{k}\right)^{1-\varepsilon(r)} \rceil$ sets $U_1, \ldots, U_\ell$ each of size $|U_i| \leq t := \lceil 3n^{\varepsilon(r)} k^{1-\varepsilon(r)} \rceil$;
7   **for** $i \in [\ell]$ **do**
8   Run NA-StrongSubsetQuery(U_i, s) to learn the partition $\mathcal{C}_i = \{C \cap U_i \colon C \in \mathcal{C}\}$;
9   \\\\ ▷ Note that $s \leq |U_i|$ and so this is a valid call to NA-StrongSubsetQuery. \\\\
10   Form R_i by taking exactly one representative from each set $C \in \mathcal{C}_i$;
11   **end**
12   Set $R = R_1 \cup \cdots \cup R_\ell$ and $s' = \min(s, |R|^{\varepsilon(r-1)} k^{1-\varepsilon(r-1)})$ (this is so that the recursive call to LR-StrongSubsetQuery has a valid query-size bound);
13   Recursively call LR-WeakSubsetQuery$(R, s', r-1)$ and let $\mathcal{C}_R$ be the returned partition of R;
14   **Return** the partition $\{\bigcup_{C' \in \mathcal{C}_1 \cup \cdots \cup \mathcal{C}_\ell \colon C' \cap C \neq \emptyset} C' \colon C \in \mathcal{C}_R\}$;
15 **end**

queries, where the last step simply used $n \geq k$. Note that in this case the algorithm is non-adaptive and again correctness follows from Theorem 3. This completes the proof for the case of $n \leq 16k$.

Now suppose that $n > 16k$. The proof of correctness is simple and completely analogous to that of Theorem 1. Now let us prove the desired query complexity. Since $r \geq 2$, note that $1 - \varepsilon(r) \geq 1 - \varepsilon(2) \geq 2/3$. Using these bounds, we have $(n/k)^{1-\varepsilon(r)} > 16^{2/3} > 6$. Recalling the definition of ℓ and t in line (6), this implies that

$$\ell \leq \frac{1}{3}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} + 1 = \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} - \frac{1}{6}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} + 1 < \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} \tag{18}$$

and clearly $t \leq 4n^{\varepsilon(r)} k^{1-\varepsilon(r)}$. Thus, using the bound on the query complexity of NA-StrongSubsetQuery, the first round (line 8) makes at most

$$\ell \cdot \frac{5t^2}{s^2} < \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} \cdot \frac{80n^{2\varepsilon(r)} k^{2(1-\varepsilon(r))}}{s^2} \leq \frac{40n^{1+\varepsilon(r)} k^{1-\varepsilon(r)}}{s^2} \tag{19}$$

queries. Then, the resulting set R is of size

$$|R| \leq k \cdot \frac{1}{2}\left(\frac{n}{k}\right)^{1-\varepsilon(r)} = \frac{1}{2} \cdot n^{1-\varepsilon(r)} k^{\varepsilon(r)}. \tag{20}$$

We now show that the total number of queries in the remaining $r-1$ rounds is at most $\frac{40n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2}$. Combining this with eq. (18) shows that the total number of queries is then at most this $\frac{80n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2}$, which completes the proof.

First, suppose that $s' = |R|^{\varepsilon(r-1)}k^{1-\varepsilon(r-1)}$. By induction, the recursive call to LR-StrongSubsetQuery in line (12) costs at most

$$80\frac{|R|^{1+\varepsilon(r-1)}k^{1-\varepsilon(r-1)}}{|R|^{2\varepsilon(r-1)}k^{2(1-\varepsilon(r-1))}} = 80\left(\frac{|R|}{k}\right)^{1-\varepsilon(r-1)} \tag{21}$$

queries. If $|R| \leq k$, then the RHS above is at most 80, which clearly satisfies the desired bound. Otherwise, the RHS of eq. (21) is at most $80|R|/k$ and using eq. (20), this is at most $40(n/k)^{1-\varepsilon(r)}$. Now, since $s \leq n^{\varepsilon(r)}k^{1-\varepsilon(r)}$, observe that

$$\frac{40n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2} \geq 40\left(\frac{n}{k}\right)^{1-\varepsilon(r)}$$

and so the number of queries in the remaining $r-1$ rounds satisfies the desired bound.

Now, suppose $s' = s$. By induction, using the upper on $|R|$ from eq. (20), we obtain that running LR-StrongSubsetQuery$(R, s, r-1)$ costs at most

$$\begin{aligned}
&\frac{80}{s^2}\left(\frac{1}{2}\cdot n^{1-\varepsilon(r)}k^{\varepsilon(r)}\right)^{1+\varepsilon(r-1)}k^{1-\varepsilon(r-1)} \\
&\leq \frac{40n^{(1-\varepsilon(r))(1+\varepsilon(r-1))}k^{\varepsilon(r)(1+\varepsilon(r-1))+(1-\varepsilon(r-1))}}{s^2} = \frac{40n^{1+\varepsilon(r)}k^{1-\varepsilon(r)}}{s^2}
\end{aligned}$$

queries where the equality is by Claim 11. This completes the proof. ■