Better Sum Estimation via Weighted Sampling

LORENZO BERETTA, University of Copenhagen, Kobenhavn, Denmark
JAKUB TĚTEK, University of Copenhagen, Kobenhavn, Denmark

Given a large set $U$ where each item $a \in U$ has weight $w(a)$, we want to estimate the total weight $W = \sum_{a \in U} w(a)$ to within factor of $1 \pm \epsilon$ with some constant probability $> 1/2$. Since $n = |U|$ is large, we want to do this without looking at the entire set $U$. In the traditional setting in which we are allowed to sample elements from $U$ uniformly, sampling $\Omega(n)$ items is necessary to provide any non-trivial guarantee on the estimate. Therefore, we investigate this problem in different settings: in the proportional setting we can sample items with probabilities proportional to their weights, and in the hybrid setting we can sample both proportionally and uniformly. These settings have applications, for example, in sublinear-time algorithms and distribution testing.

Sum estimation in the proportional and hybrid setting has been considered before by Motwani, Panigrahy, and Xu [ICALP, 2007]. In their article, they give both upper and lower bounds in terms of $n$. Their bounds are near-matching in terms of $n$, but not in terms of $\epsilon$. In this article, we improve both their upper and lower bounds. Our results imply an algorithm with complexity $O(\sqrt{n}/\epsilon)$ when $n$ is large, we want to do this without looking at the entire set $U$. In the traditional setting in which we are allowed to sample elements from $U$ uniformly, sampling $\Omega(n)$ items is necessary to provide any non-trivial guarantee on the estimate. Therefore, we investigate this problem in different settings: in the proportional setting we can sample items with probabilities proportional to their weights, and in the hybrid setting we can sample both proportionally and uniformly. These settings have applications, for example, in sublinear-time algorithms and distribution testing.

Our algorithms have applications in the area of sublinear-time graph algorithms. Consider a large graph $G = (V, E)$ and the task of $(1 \pm \epsilon)$-approximating $|E|$. We consider the (standard) settings where we can sample uniformly from $E$ or from both $E$ and $V$. This relates to sum estimation as follows: we set $U = V$ and the weights to be equal to the degrees. Uniform sampling then corresponds to sampling vertices uniformly. Proportional sampling can be simulated by taking a random edge and picking one of its endpoints at random. If we can only sample uniformly from $E$, then our results immediately give a $O(\sqrt{|V|}/\epsilon)$ algorithm. When we may sample both from $E$ and $V$, our results imply an algorithm with complexity $O(\sqrt{|V|}/\epsilon^3/4)$. Surprisingly, one of our subroutines provides an $(1 \pm \epsilon)$-approximation of $|E|$ using $O(d/\epsilon^2)$ expected samples, where $d$ is the average degree, under the mild assumption that at least a constant fraction of vertices are non-isolated. This subroutine works in the setting where we can sample uniformly from both $V$ and $E$. We find this remarkable since it is $O(1/\epsilon^2)$ for sparse graphs.
1 INTRODUCTION

Suppose we have a large set $U$, a weight function $w : U \rightarrow [0, \infty)$ and we want to compute a $(1 \pm \epsilon)$-approximation of the sum of all weights $W = \sum_{a \in U} w(a)$. Since $n = |U|$ is very large, we want to estimate $W$ by sampling as few elements as possible. In the traditional setting in which we are allowed to sample elements from $U$ uniformly, sampling $o(n)$ items cannot provide any non-trivial guarantee on the approximation as we may miss an element with a very large weight. This led Motwani, Panigrahy, and Xu [14] to study this problem when we are allowed to sample elements proportionally to their weights (i.e., sample $a$ with probability $w(a)/W$). In particular, they studied two settings: the proportional setting, where we can sample items proportionally to their weights, and the hybrid setting where both proportional and uniform sampling is possible.

In this article, we revisit these two settings and get both improved lower and upper bounds. We also extend the results to more general settings and show how our techniques imply new results for counting edges in sublinear time.

Motwani et al. [14] give upper and lower bounds for both proportional and hybrid settings. Their bounds are matching up to polylogarithmic factors in terms of $n$, but not in terms of $\epsilon$. In this article, we improve both their upper and lower bounds. Our bounds are matching up to polylogarithmic factors in both settings, in terms of both $n$ and $\epsilon$. Moreover, for wide parameter regimes our bounds are matching up to constant factors. No lower bounds with dependency on $\epsilon$ were known previously.

In the proportional setting, we improve their $\tilde{O}(\sqrt{n}/\epsilon^{7/2})$ algorithm to $O(\sqrt{n}/\epsilon)$. In the hybrid setting, we improve $\tilde{O}(\sqrt{n}/\epsilon^{9/2})$ to $O(\sqrt{n}/\epsilon^{4/3})$. Our algorithms are also significantly simpler. In the same article, Motwani et al. [14] write: “to efficiently derive the sum from [proportional] samples does not seem straightforward”. We would like to disagree and give a formula that outputs an estimate of $W$ from a proportional sample. This formula is not only optimal in terms of sample complexity but also very simple. Our other algorithms, although not as simple, do not have large hidden constants and we believe they are both practical and less involved than their predecessors.

In their work, Motwani et al. [14] always assume to know the size of the universe $n = |U|$. We extend these results to the case of unknown $n$. In this case, we obtain a $O(\sqrt{n}/\epsilon + \log n/\epsilon^2)$ algorithm for the proportional setting, and a $O(\sqrt{n}/\epsilon)$ algorithm for the hybrid setting. This means that in the proportional setting we may remove the need for advice without significantly impacting the complexity of the problem, while there is a major difference in the hybrid setting. We prove that this difference in the hybrid setting is necessary, by showing a matching lower bound.

We give lower bounds for all our estimation problems, both for proportional and hybrid settings, and when $n$ is either known or unknown. This is the most technically challenging part of the article. We prove lower bounds for $n$ known and unknown as well as proportional and hybrid settings; all our lower and upper bounds are matching up to a constant factor. See Table 1 for a summary of our results.

Our algorithms have particularly interesting applications in the area of sublinear-time graph algorithms, which are explained in detail in Section 1.1.
### Table 1. Results of This Article

| Advice to the algorithm | This article | Motwani, Panigrahy and Xu [14] |
|-------------------------|--------------|-------------------------------|
|                         | Proportional | Hybrid                        |
| $n$ known               | $\Theta(\sqrt{n}/\epsilon)$ | $O(\min(\sqrt{n}/\epsilon^{4/3}, n \log n))$ |
|                         | $\Omega(\min(\sqrt{n}/\epsilon^{4/3}, n))$ | $\Omega(\sqrt{n}/\epsilon^{7/2})$ |
| Known $n \geq n$       | $O(\sqrt{n}/\epsilon)$ | $O(\min(\sqrt{n}/\epsilon, n \log n))$ |
|                         | $\Omega(\sqrt{n}/\epsilon)$ | $\Omega(\min(\sqrt{n}/\epsilon, n))$ |
| No advice               | $O(\sqrt{n}/\epsilon + \log n/\epsilon^2)$ | $O(\min(\sqrt{n}/\epsilon, n \log n))$ |
|                         | $\Omega(\sqrt{n}/\epsilon)$ | $\Omega(\min(\sqrt{n}/\epsilon, n))$ |

The article is structured as follows. In what is left of Section 1, we explain applications and related work, give an overview of the techniques employed, and provide the reader with formal definitions of problems and notation. Section 2 contains our algorithms for proportional setting, while Section 3 contains algorithms for hybrid setting. In Section 4, we prove all our lower bounds. In Section 5, we show how to apply our algorithms to the problem of counting the number of edges in a graph in sublinear time. In Section 6, we raise several open problems.

### 1.1 Related Work and Applications

In this section, we show that our algorithms can be applied to get sublinear-time graph algorithms and distribution testing and discuss how this relates to previous work in these areas. We also discuss here how the widely used Metropolis-Hastings algorithm in fact implements the proportional sampling. We suggest that this could be an application domain worth investigating.

#### 1.1.1 Counting Edges in Sublinear Time

When a graph $G = (V, E)$ is very large, we may want to approximately solve certain tasks without looking at the entire $G$, thus having a time complexity that is sublinear in the size of $G$. In particular, estimating global properties of $G$ such as $|V|$ or $|E|$ in this setting is an important problem and has been studied in both theoretical and applied communities [5, 6, 8, 10, 12, 16, 18]. Since the algorithm does not have the time to pre-process (or even see) the whole graph, it is important to specify how we access $G$. Several models are employed in the literature. The models differ from each other for the set of queries that the algorithm is allowed to perform. A random vertex query returns a random vertex, a random edge query returns a random edge, a pair query takes two vertices $u, v$ as arguments and returns $(u, v) \in E$, a neighborhood query takes a vertex $v$ and an index $i$ as arguments and returns the $i$th neighbor of $v$ (or says that $d(v) < i$), a degree query, given a vertex $v$, returns its degree $\deg(v)$. We parameterize the complexities with an approximation parameter $\epsilon$, $n = |V|$ and $m = |E|$.

The problem of estimating the number of edges in a graph in sublinear time has been first considered by Feige [8]. Their algorithm works in the model where only random vertex queries and degree queries are allowed and achieves a $(2 + \epsilon)$-approximation algorithm using $\hat{O}(\frac{n}{\epsilon \sqrt{m}})$ time and queries. Their algorithm does not use neighborhood queries and the authors showed that without neighborhood queries, $2 - \epsilon$ approximation requires a linear number of queries. Goldreich and Ron [10] broke the barrier of factor 2 by using neighborhood queries. Indeed, they showed a $(1 + \epsilon)$-approximation with time and query complexity of $\hat{O}(\frac{n}{\epsilon \sqrt{m}})$. Currently, the best-known algorithm is the one by Eden, Ron, and Seshadhri [6] and has complexity $\hat{O}(\frac{n}{\epsilon \sqrt{m}})$. If pair queries are allowed, the algorithm of Tětek and Thorup [18] has complexity $\hat{O}(\frac{n}{\epsilon \sqrt{m}} + \frac{1}{\epsilon^3})$; in the same article, the
authors showed a lower bound which is near-matching for $\varepsilon \geq m^{1/6}/n^{1/3}$ as well as an algorithm with complexity $\tilde{O}(\frac{m}{\sqrt{m}} + \frac{1}{\varepsilon^2})$ in what they call the hash-ordered access model. They also show an algorithm in the more standard setting with random vertex and neighborhood queries, that runs in time $\tilde{O}(\sqrt{n}/\varepsilon)$. This is the same complexity (up to a $\log^{O(1)} n$ factor) that we achieve in the setting with random edge queries, as we discuss below. Our techniques are, however, completely different and share no similarity with the techniques used in [18].

Our algorithms can be applied to solve the edge counting problem when either (i) random edge queries only are allowed, or (ii) both random edge and random vertex queries are allowed. We remark that uniform edge sampling can be (approximately) simulated using $\tilde{O}(n/\sqrt{m})$ random vertex queries, degree queries, and neighbor queries [7]. While edge counting has not been explicitly considered in these settings before, these settings are established and have been used in several articles [1, 2, 9, 17]. We instantiate our algorithm for this graph problem setting $U = V$ and $w(v) = \text{deg}(v)$ for each $v \in V$. Uniform sampling then corresponds to sampling vertices uniformly. Proportional sampling can be simulated by taking a random edge and picking one of its endpoints at random. Degree query allows us to get the weights of sampled vertices. Since these settings have not been explicitly studied before, we compare our results with what follows directly from the known literature.

If we can only sample uniformly from $E$, the algorithms by Motwani et al. implies an algorithm for this problem that has complexity $\tilde{O}(\sqrt{n}/\varepsilon^{3/2})$. Using an algorithm from [6] and standard simulation of random vertex queries using random edge queries, one would get time $\tilde{O}(\sqrt{m}/\varepsilon^2 + \frac{m}{\varepsilon^2 \sqrt{n}})$ for $n'$ being the number of non-isolated$^1$ vertices.$^2$ Our results immediately give a $O(\sqrt{n}/\varepsilon)$ algorithm, or $\tilde{O}(\sqrt{n}/\varepsilon + 1/\varepsilon^2)$ when $n$ is not known.

When we may sample both from $E$ and $V$, the algorithms by Motwani et al. imply algorithms with sample complexities of $\tilde{O}(\sqrt{n}/\varepsilon^{3/2})$ and $O(\sqrt{n}/\varepsilon^2)$. We may also use the algorithm of [6] that relies on random vertex query only. This has complexity from $O(\frac{n}{\varepsilon^2 \sqrt{n}})$. Our results imply an algorithm with complexity $O(\sqrt{n}/\varepsilon^{5/3})$. Surprisingly, one of our subroutines provides an $(1 \pm \varepsilon)$-approximation of $|E|$ using $\tilde{O}(d/\varepsilon^2)$ expected samples, where $d$ is the average degree, under the mild assumption that at least a constant fraction of vertices are non-isolated (in fact, we prove a more complicated complexity which depends on the fraction of vertices that are isolated). This subroutine works in the setting where we can sample uniformly from both $V$ and $E$. We find this interesting since it is $O(1/\varepsilon^2)$ for sparse graphs.

1.1.2 Distribution Testing. Consider a model in which we are allowed to sample from a distribution $\mathcal{D}$ on $U$, and when we do, we receive both $a \in U$ and $P_D(a)$. This model is stronger than the one we considered throughout the article. In fact, it is a special case of our model obtained by fixing $W = 1$. Similarly, we may consider such stronger variant of the hybrid setting. This model has received attention both in statistics and in theoretical computer science literature. Most notably, Horvitz and Thompson [11] in 1952 showed how to estimate the sum $\sum_{a \in U} v(a)$, for value function $v : U \to \mathbb{R}$ when having access to a sample $a$ from $\mathcal{D}$ and $P_D(a)$. More recently, many distribution testing problems have been considered in this and similar models.

First, Canonne and Rubinfeld [3] considered the model where one can (i) sample $a \in U$ according to $\mathcal{D}$; (ii) query an oracle that, given $a \in U$, returns $P_D(a)$. This allows us to both get the probability of a sampled item, but also the probability of any other item. For every permutation-invariant

---

$^1$A vertex is isolated if it has degree zero.

$^2$One may simulate uniform sampling from the set of non-isolated vertices at multiplicative overhead of $O(m/n')$ by sampling proportionally and using rejection sampling. We may then use set size estimation by birthday paradox in time $O(\sqrt{n'}/\varepsilon)$ to learn $n'$ and the algorithm of [6].
problem, any sublinear-time algorithm in this model may be transformed so that it only queries
the oracle on previously sampled items or on items chosen uniformly at random. This setting is
stronger than our hybrid setting. However, the two models become equivalent if we know \( W \).

Second, Onak and Sun [15] considered exactly the model described above, where one can sample
from a distribution \( D \), and both \( a \in U \) and \( P_D(a) \) are returned. This setting is again stronger than
our proportional setting, and it becomes equivalent once we know \( W \).

In both of these articles, the authors solved several distribution testing problems such as: uniformity
testing, identity testing, closeness testing, distance to a known distribution, and distance
between two unknown distributions in the respective models. Canonne and Rubinfeld [3] also con-
sidered the problem of providing an additive approximation of entropy. Both [3] and [15] proved
several lower bounds, showing that many of their algorithms are optimal up to constant factors.
These lower bounds also imply a separation between the two models for some of the problems
mentioned above.

In both of the articles, the authors make a point that many of their algorithms (all the ones
we mentioned) are robust with respect to noise of multiplicative error \((1 \pm \varepsilon/2)\) in the answers
of the probability oracles. Our algorithms can then serve as a reduction from the weaker models
we consider in this article to these stronger models where we know the probabilities and not just
the weights. The reason is that we can get an approximation of \( W \), meaning that we may then
approximately implement the above-described models. Since the mentioned algorithms [3, 15] are
robust, a \((1 \pm \varepsilon/2)\)-approximation of \( W \) is sufficient to simulate them.

1.1.3 Proportional Sampling in Practice: Metropolis-Hastings. Proportional sampling is often
implemented in practice using the Metropolis-Hasting algorithm. This algorithm is widely used
in statistics and statistical physics, but can also be used to sample combinatorial objects. It can
be used to sample from large sets which have complicated structures that make it difficult to use
other sampling methods. Just like our algorithms, it is suitable when the set is too large to be stored
explicitly, making it impossible to pre-process it for efficient sampling. One of the main appeals
of Metropolis-Hasting is that it does not require one to know the exact sampling probabilities, but
it is sufficient to know the items’ weights like in the proportional setting described in this arti-
cle. We thus suggest that our algorithm could find practical applications in combination with the
Metropolis-Hasting algorithm.

1.2 Overview of Employed Techniques

Here we provide a summary of the techniques employed throughout the article. We denote with
\( P_{\text{unif}}(\cdot) \) and \( P_{\text{prop}}(\cdot) \) the probabilities computed according to uniform and proportional sampling,
respectively. Although often not specified for brevity, all guarantees on the approximation factors
of estimates in this section are meant to hold with probability \( 2/3 \). These probabilities can be
amplified to \( 1 - \delta \) through \( O(\log 1/\delta) \) repetitions using the folklore median trick. We do not know
whether the dependency w.r.t. \( \delta \) is optimal and this would be an interesting further question.

1.2.1 Proportional Setting with Advice \( \tilde{n} \geq n \). Consider sampling two elements \( a_1, a_2 \in U \)
proportionally and define \( Y_{12} = 1/\omega(a_1) \) if \( a_1 = a_2 \), and \( Y_{12} = 0 \), otherwise. It is easy to show that
\( Y_{12} \) is an unbiased estimator of \( 1/W \). We could perform this experiment many times and take the
average. This would give a good approximation to \( 1/W \) and taking the inverse value, we would

\( ^3\)Since we consider permutation invariant problems, we may randomly permute the elements. Whenever we query an
element that has not yet been sampled, we may assume that it is sampled uniformly from the set of not-yet-sampled
elements. This can be simulated by sampling elements until getting a not-yet-seen item (sublinearity ensures that the step
adds multiplicative \( O(1) \) overhead).
get a good estimate on $W$. Unfortunately, we would need $\Theta(n/\epsilon^2)$ repetitions in order to succeed with a constant probability. We can fix this as follows: we take $m$ samples $a_1, \ldots, a_m$ and consider one estimator $Y_{ij}$ for each pair of samples $a_i, a_j$ for $i \neq j$. This allows us to get $\binom{n}{2}$ estimators from $m$ samples. We show that estimators $Y_{ij}$ are uncorrelated. This reduces the needed number of samples from $O(n/\epsilon^2)$ to $O(\sqrt{n}/\epsilon)$.

We now describe the estimator formally. Let $S = \{a_1, \ldots, a_m\}$ be the set of sampled items, and for each $s \in S$ define $c_s$ to be the number of times item $s$ is sampled. Then,

$$\hat{W} = \left(\frac{m}{2}\right) \cdot \left(\sum_{s \in S} \frac{c_s}{w(s)}\right)^{-1}$$

is a $(1 \pm \epsilon)$-approximation of $W$ with probability $2/3$ for $m = \Theta(\sqrt{n}/\epsilon)$. If, instead of knowing $n$ exactly, we know some $\hat{n} \geq n$ we can achieve the same guarantees by taking $\Theta(\sqrt{\hat{n}}/\epsilon)$ samples.

1.2.2 Proportional Setting. Unknown $n$. We partition $U$ into buckets $B_i = \{a \in U \mid w(a) \in [2^i, 2^{i+1})\}$. We choose some $b \in \mathbb{Z}$ and compute two estimates: $\hat{W}_b$ approximates $W_b = \sum_{a \in B_b} w(a)$, and $\hat{P}_b$ approximates $P_b = P_{prop}(a \in B_b) = W_b/W$. We then return $\hat{W}_b/\hat{P}_b$ as an estimate of $W$. If both estimates are accurate, then the returned value is accurate. To choose $b$, we sample $a_1$ and $a_2$ proportionally and define $b$ so that $\max\{a_1, a_2\} \in [2^b, 2^{b+1})$. We prove that, surprisingly, $E[1/P_b] = O(\log n)$. Computing $\hat{P}_b$ is simply a matter of estimating the fraction of proportional samples falling into $B_b$. This can be done using $O(1/(P_b\epsilon^2))$ samples, where $\epsilon$ is the approximation parameter. Therefore, the expected complexity of computing $\hat{P}_b$ is $O(\log n/\epsilon^2)$. Computing $\hat{W}_b$ is more involved. First, we design two subroutines to sample from $B_b$, one for proportional sampling and one for uniform. These subroutines work by sampling $a \in U$ until we get $a \in B_b$; the subroutine for uniform sampling then uses rejection sampling. These subroutines take, in expectation, $O(1/P_b)$ samples to output one sample from $B_b$. Since we can now sample uniformly from $B_b$, we sample items uniformly form $B_b$ until we find the first repeated item and use the stopping time to infer $|B_b|$ up to a constant factor. In expectation, we use $O(\sqrt{|B_b|})$ uniform samples and compute $\tilde{n}_b$, such that $|B_b| \leq \tilde{n}_b \leq O(|B_b|)$. Since we can also sample proportionally from $B_b$, we may use the algorithm for proportional setting with advice $\tilde{n}_b$ to estimate $W_b$. This yields a total sample complexity of $O(\sqrt{n}/\epsilon + \log n/\epsilon^2)$.

1.2.3 Hybrid Setting. We now present the techniques used in the hybrid setting. Before giving a sketch of the main algorithms, we introduce two subroutines.

Coupon-collector-based algorithm. In the hybrid setting, we can sample elements uniformly. If we know $n = |U|$ a well-known result under the name of “coupon collector problem” shows that we can retrieve with high probability all $n$ elements by performing $\Theta(n \log n)$ uniform samples. We extend this result to the case of unknown $n$. We maintain a set of retrieved elements $S \subseteq U$ and keep on sampling uniformly and adding new elements to $S$ until we perform $\Theta(|S| \log |S|)$ samples in a row without updating $S$. It turns out that this procedure retrieves the whole set $U$ with probability $2/3$ and expected complexity $O(n \log n)$. According to our lower bounds, this algorithm is near-optimal when $\epsilon = O(1/\sqrt{n})$. Therefore, we can focus on studying hybrid sampling for larger values of $\epsilon$.

Harmonic mean and estimating $W/n$ with advice $\tilde{\Theta} \geq W/n$ in hybrid setting. If we sample $a \in U$ proportionally, then $1/w(a)$ is an unbiased estimator of $n/W$. Unfortunately, we may have very

4Notice that, if we just define $b = a_i$, then we have $E[1/P_b] = n$ in the worst case where we have $n$ distinct non-empty buckets. Therefore, taking the maximum of two samples entails an exponential advantage.
small values of \( w(a) \), which can make the variance of this estimator arbitrarily large. To reduce variance, we can set a threshold \( \phi \) and define an estimator \( Y_\phi \) as \( 1/w(a) \) if \( w(a) \geq \phi \), and 0 otherwise. Set \( p = P_{\text{unif}}(w(a) \geq \phi) \), then we have \( E[Y_\phi] = pn/W \) and \( \text{Var}(Y_\phi) \leq pn/(\phi W) \). If we define \( \bar{Y}_\phi \) as the average of \( \theta/(p\phi e^2) \) copies of \( Y_\phi \), we have \( \text{Var}(\bar{Y}_\phi) \leq (\epsilon E[Y_\phi])^2 \). With such a small variance, \( \bar{Y}_\phi \) is a good estimate of \( pn/W \). Estimating \( p \) is simply a matter of estimating the fraction of uniform samples having \( w(\bullet) \geq \phi \). This can be done taking \( O(1/(p^2)) \) uniform samples. Once we have estimates of both \( p \) and \( pn/W \) we return their ratio as an estimate of \( W/n \). We employed in total \( O((1 + \frac{\theta}{\phi})/(p^2)) \) proportional and uniform samples.

Perhaps surprisingly, this corresponds to taking the harmonic mean of the samples with weight at least \( \phi \) and adjusting this estimate for the weight of the items with weight \( < \phi \).

**Hybrid setting, known \( n \).** We now sketch the algorithm for sum estimation in the hybrid setting with known \( n \). We combine our formula to estimate \( W \) in the proportional setting and the above algorithm to estimate \( W/n \) in hybrid setting to obtain an algorithm that estimates \( W \) using \( O(\sqrt{n}/e^{1/3}) \) uniform and proportional samples. Define \( U_{\geq \theta} = \{ a \in U \mid w(a) \geq \theta \} \). We find a \( \theta \) such that \( P_{\text{unif}}(a \in U_{\geq \theta}) \approx n^{-1/3}e^{-2/3} \), this can be done taking enough uniform samples and picking the empirical \((1 - n^{-1/3}e^{-2/3})\)-quantile.\(^5\) We define \( p = P_{\text{prop}}(a \in U_{\geq \theta}) \geq P_{\text{unif}}(a \in U_{\geq \theta}) \approx n^{-1/3}e^{-2/3} \).

We compute an estimate \( \hat{p} \) of \( p \) counting the fraction of proportional samples falling in \( U_{\geq \theta} \). Now we have two cases. If \( \hat{p} \geq 1/2 \), we can simulate sampling from the proportional distribution on \( U_{\geq \theta} \) with \( O(1) \) overhead by sampling proportionally until we get an element of \( U_{\geq \theta} \). Then, we use the algorithm for sum estimation under proportional sampling restricted to elements in \( U_{\geq \theta} \) to estimate \( \sum_{a \in U_{\geq \theta}} w(a) = \hat{p} W \). Dividing by the estimate \( \hat{p} \approx p \), we get an estimate for \( W \).

Else, \( \hat{p} < 1/2 \), and thus, \( p \leq 2/3 \) (assuming \( \hat{p} \) is a good enough estimate of \( p \)). We then have \( \theta n \geq \sum_{a \in U_{\geq \theta}} w(a) = (1-p)W \geq W/3 \). This allows us to use the harmonic-mean-based algorithm to estimate \( W/n \) with \( \phi = \theta \) and \( \bar{\theta} = 3\theta \). In both cases, we manage to provide an estimate of \( W \) using \( O(\sqrt{n}/e^{4/3}) \) samples.

**Hybrid setting, unknown \( n \).** We now sketch our technique for sum estimation in the hybrid setting with unknown \( n \). First, we sample items uniformly until we find the first repeated item and use the stopping time to infer the total number of items \( n \). In expectation, we use \( O(\sqrt{n}) \) uniform samples and compute \( \hat{n} \), such that \( n \leq \hat{n} \) and \( E[\hat{n}] = n \). Now, we can use our algorithm for sum estimation in the proportional setting with advice \( \hat{n} \). This uses in expectation \( O(\sqrt{n}/\epsilon) = O(\sqrt{n}/\epsilon) \) samples. As we prove, this complexity is optimal up to a constant factor for \( \epsilon \geq 1/\sqrt{n} \). Again, if \( \epsilon \leq 1/\sqrt{n} \), then we use our coupon-collector-based algorithm that retrieves every element of \( U \) using \( O(n \log n) \) samples.

1.2.4 Lower Bounds. The most technically challenging part of our article is Section 4, where we prove lower bounds for all estimation problems we address in the first part of the article. All our lower bound proofs follow a common thread. We now sketch the main ideas. First, we define two different instances of the estimation problem at hand \((U_1, w_1)\) and \((U_2, w_2)\) such that a \((1 \pm \epsilon)\)-approximation of \( W \) is sufficient to distinguish between them. Then, we define our hard instance as a mixture of the two: we take \((U_1, w_1)\) with probability 1/2 and \((U_2, w_2)\) otherwise. We denote these events by \( E_1 \) and \( E_2 \), respectively. Second, we show that a Bayes classifier cannot distinguish between the two cases with probability 2/3 using too few samples; since Bayes classifiers are

\(^5\)We may assume that \( \epsilon \geq 8/\sqrt{n} \) by running the coupon-collector-based algorithm when \( \epsilon < 8/\sqrt{n} \). Under this assumption, it holds \( n^{-1/3}e^{-2/3} \in [0, 1] \) and taking the \((1 - n^{-1/3}e^{-2/3})\)-quantile then is meaningful.
risk-optimal\(^6\) this implies that no classifier can have misclassification probability less than 1/3 while using the same number of samples. To show that a Bayes classifier has a certain misclassification probability, we study the posterior distribution, conditioned on the samples it has seen. Let the multiset \(S\) represent the outcome of the samples. Since the prior is uniform, applying the Bayes theorem gives
\[
\frac{P(S | E_1)}{P(S | E_2)} = \frac{P(E_1 | S)}{P(E_2 | S)}.
\]

We denote this likelihood ratio by \(\mathcal{R}(S)\). We show that whenever \(|S|\) is (asymptotically) too small, we have \(\mathcal{R}(S) \approx 1\) with probability close to 1. When \(\mathcal{R}(S) \approx 1\), the posterior distribution is very close to uniform. This entails misclassification probability close to 1/2. If \(X\) and \(Y\) are some random variables sufficient to reconstruct \(S\) (that is, there exists an algorithm that, given \((X, Y)\), generates \(S'\) with the same distribution as \(S\)), we can define
\[
\mathcal{R}(X) = \frac{P(X | E_1)}{P(X | E_2)} \quad \text{and} \quad \mathcal{R}(Y | X) = \frac{P(Y | X, E_1)}{P(Y | X, E_2)}
\]
and we have, thanks to the Bayes theorem, \(\mathcal{R}(S) = \mathcal{R}(X) \cdot \mathcal{R}(Y | X)\). In this way, we can break the problem of proving \(\mathcal{R}(S) \approx 1\) into proving \(\mathcal{R}(X) \approx 1\) and \(\mathcal{R}(Y | X) \approx 1\). This allows us to reduce all our lower bounds to proving concentration of likelihood ratios for two basic problems: (1) distinguishing between two sets of size \(n\) and \((1-\varepsilon)n\) by uniform sampling and (2) distinguishing between two sequences of i.i.d. random variables from \(\text{Bern}(p)\) and \(\text{Bern}(p - \varepsilon)\).

We now sketch how we bound the likelihood ratio \(\mathcal{R}(S)\) for problem (1), the same technique applies to problem (2). In problem (1), we call \(E_1\) the event \(|U| = n\) and \(E_2\) the event \(|U| = (1-\varepsilon)n\), and we set \(P(E_1) = P(E_2) = 1/2\). First, we notice that \(\mathcal{R}(S)\) depends only on the number \(\ell(S)\) of distinct elements in \(S\). Then, we prove three facts. First, \(\ell(S) | E_i\) is concentrated around \(E[\ell(S) | E_i]\). Second, \(E[\ell(S) | E_1] \approx E[\ell(S) | E_2]\). Third, for small deviations of \(\ell(S)\), we have small deviations of \(\mathcal{R}(\ell(S))\). These three facts are sufficient to conclude that, with probability close to 1, \(\mathcal{R}(\ell(S))\) lies in a very narrow interval; further computations show that 1 lies in that interval, hence \(\mathcal{R}(S) \approx 1\) with probability close to 1.

1.3 Preliminaries

Problem definition. We now give a formal definition of the two settings that we consider. Let us have a set \(U\) of cardinality \(n\) and a weight function \(w: U \rightarrow [0, \infty)\). We denote by \(W\) the sum \(\sum_{a \in U} w(a)\). The following operations are allowed in the proportional sampling setting: (1) proportionally sample an item, this returns \((a, w(a))\) with probability \(w(a)/W\); (2) given two items \(a, a'\), check whether \(a = a'\). This is the only way we can interact with the items. In the hybrid setting, we may in addition (3) sample an item uniformly (that is, return \((a, w(a))\) for any \(a \in U\) with probability \(1/n\)). In both settings, we want to compute an estimate \(\hat{W}\) of \(W\) such that \((1-\varepsilon)W \leq \hat{W} \leq (1+\varepsilon)W\) with probability \(2/3\).

Notation. When \((1-\varepsilon)W \leq \hat{W} \leq (1+\varepsilon)W\) holds, we say that such \(\hat{W}\) is a \((1 \pm \varepsilon)\)-approximation of \(W\). Some of our subroutines require “advice” in the form of a constant factor approximation of some value. For sake of consistency, we denote this constant factor approximation of \(\bullet\) by \(\bullet\). Similarly, if we want to estimate some value \(\bullet\), we use \(\bullet\) to denote the estimate. Let us have some predicate \(\phi\) that evaluates true on some subset of \(U\) and false on the rest. We denote by \(P_{\text{unif}}(\phi(a))\) and \(P_{\text{prop}}(\phi(a))\) the probability of \(\phi\) evaluating to true for \(a\) being picked uniformly and proportionally, respectively.

---

\(^6\)Risk of a classifier refers to the misclassification probability under some fixed distribution. For the Bayes classifier, we implicitly assume that this distribution is the same as the prior.
We state all our results (both upper and lower bound) for some constant success probability $> 1/2$. These probabilities can be amplified to any other constants without increasing the asymptotic complexity. In pseudocode, we often say that we execute some algorithm with some failure probability. By this, we mean that one uses probability amplification to achieve that failure probability.

Relative bias estimation of a Bernoulli random variable. Let $X_1, X_2, \ldots$ be i.i.d. random variables distributed as $\text{Bern}(p)$. [13] gave a very simple algorithm that returns $\hat{p}$ such that, with probability at least $2/3$, $\hat{p}$ is a $(1 \pm \epsilon)$-approximation of $p$.

**Proposition 1.1 (follows from [13]).** Let $X_1, X_2, \ldots$ be i.i.d. random variables distributed as $\text{Bern}(p)$. There exists an algorithm that uses in expectation $O\left(\frac{1}{\epsilon^2 p}\right)$ samples and returns $\hat{p}$ such that $E[1/\hat{p}] = 1/p$ and

$$P (|\hat{p} - p| > \epsilon p) \leq \frac{1}{3}.$$  

We call this algorithm $\text{BERNOULLIESTIMATOR}(\epsilon)$. We assume that this algorithm has access to the sequence $X_1, X_2, \ldots$; we specify these random variables when invoking the algorithm.

Probability amplification and expected values. Consider an estimator that gives a guarantee on the estimate $\hat{x}$ that holds with some probability (say, guarantee that a proposition $\phi(\hat{x})$ holds with probability at least $2/3$) and at the same time, we know that $E[\hat{x}] \leq y$ for some value $y$. We sometimes need to amplify the probability of the guarantee (that is, amplify the probability that $\phi(\hat{x})$ holds) but would like to retain a bound $E[\hat{x}] = O(y)$. We now argue that using the standard median trick is sufficient. Namely, we prove that

**Lemma 1.2.** Let us have non-negative i.i.d. random variables $X_1 \ldots X_{2t-1}$ for some integer $t$, and let $X = \text{median}(X_1 \ldots X_{2t-1})$. It holds $E[X] \leq 2E[X_i]$.

**Proof.** Let $X'_1, \ldots, X'_{2t-1}$ be the random variables $X_1, \ldots, X_{2t-1}$ sorted in increasing order. We then have

$$E[X] \leq E \left[ \frac{1}{t} \cdot \sum_{i=t}^{2t-1} X'_i \right] \leq E \left[ \frac{1}{t} \cdot \sum_{i=1}^{2t-1} X_i \right] \leq \frac{2t-1}{t} \cdot E[X_1].$$

$$\square$$

## 2 SUM ESTIMATION BY PROPORTIONAL SAMPLING

In this section, we focus on sum estimation in the proportional setting. We design algorithms to estimate $W$ and our objective is to minimize the total number of samples taken in the worst case. We present two different algorithms that provide an $(1 \pm \epsilon)$-approximation of $W$ with probability $2/3$. The first one, PropEstimator, assumes to have an upper bound on the number of elements $\tilde{n} \geq n$, and achieves sample complexity of $O(\sqrt{\tilde{n}}/\epsilon)$. The second one, NoAdvicePropEstimator, does not assume any knowledge of $n$, and produce an $\epsilon$-estimate using $O(\sqrt{\tilde{n}}/\epsilon + \log n/\epsilon^2)$ samples in expectation.

### 2.1 Algorithm with Advice $\tilde{n} \geq |U|$  
Let $a_1 \ldots a_m$ be $m$ items picked independently at random from $U$ with probabilities proportional to their weights. Let $S$ be the set of sampled items, and for each $s \in S$ define $c_s$ to be the number of times item $s$ is sampled. For each $i, j \in [m]^2$ define $Y_{ij}$ to be $1/w(a_i)$ if $a_i = a_j$ and 0, otherwise.

We now estimate $W$ as follows:

---

7In that article, the authors in fact solve a more general problem. For presentation of this special case, see [19].
ALGORITHM 1: PropEstimator($\tilde{n}, \varepsilon$)

1. Given a parameter $0 < \varepsilon < 1$ and advice $\tilde{n} \geq n$
2. Perform $m = \sqrt{24n/\varepsilon} + 1$ samples
3. Return the estimate $\tilde{W} = \left( \frac{m}{2} \right) \cdot \left( \sum_{s \in S} \frac{(\varepsilon)^2}{w(s)} \right)^{-1}$
4. In case $c_s = 1$, for all $s \in S$ set $\tilde{W} = \infty$

Before we prove correctness, we need the following lemma.

**Lemma 2.1.** Given pairwise distinct $i, j, k \in [m]$, we have $E[Y_{i,j}] = 1/W$, $\text{Var}(Y_{i,j}) \leq n/W^2$, and $\text{Cov}[Y_{i,j}, Y_{i,k}] = 0$

**Proof.**

$$E[Y_{i,j}] = \sum_{a \in U} \frac{1}{w(a)} P(x_i = x_j = a) = \sum_{a \in U} \frac{w(a)}{W^2} = \frac{1}{W}$$

$$\text{Var}(Y_{i,j}) = \sum_{a \in U} \frac{1}{w(a)^2} P(x_i = x_j = a) = \sum_{a \in U} \frac{1}{W^2} = \frac{n}{W^2}$$

As for the covariance, it holds $\text{Cov}[Y_{i,j}, Y_{i,k}] = E[Y_{i,j} \cdot Y_{i,k}] - E[Y_{i,j}] \cdot E[Y_{i,k}]$. This is equal to 0 as

$$E[Y_{i,j} \cdot Y_{i,k}] = \sum_{a \in U} \frac{1}{w(a)^2} P(x_i = x_j = x_k = a) = \sum_{a \in U} \frac{w(a)}{W^3} = \frac{1}{W^2} = E[Y_{i,j}] \cdot E[Y_{i,k}]$$

□

**Theorem 2.2.** Given parameters $\tilde{n}$ and $0 < \varepsilon < 1$, PropEstimator($\tilde{n}, \varepsilon$) has sample complexity $O(\tilde{n}/\varepsilon)$ and returns an estimate $\tilde{W}$ such that $E[1/\tilde{W}] = 1/W$. If, moreover, $\tilde{n} \geq n$, then $P(|\tilde{W} - W| \leq \varepsilon W) \geq 2/3$.

**Proof.** The sample complexity is clearly as claimed. We now prove that $1/\tilde{W}$ is an unbiased estimator of $1/W$:

$$\frac{1}{\tilde{W}} = \left( \frac{m}{2} \right)^{-1} \sum_{s \in S} \frac{1}{w(s)} \left( \frac{m}{2} \right)^{-1} \sum_{1 \leq i < j \leq m} Y_{ij}$$

and thus

$$E \left[ \frac{1}{\tilde{W}} \right] = \left( \frac{m}{2} \right)^{-1} \sum_{1 \leq i < j \leq m} E[Y_{ij}] = \frac{1}{W}$$

When $i, j, k, \ell$ are all distinct, $Y_{ij}$ and $Y_{k\ell}$ are independent. Moreover, by Lemma 2.1, $Y_{ij}$ and $Y_{ik}$ are uncorrelated for $j \neq k$. Using the bound on $\text{Var}(Y_{ij})$ from Lemma 2.1, we then have that

$$\text{Var} \left[ \frac{1}{\tilde{W}} \right] = \left( \frac{m}{2} \right)^{-2} \sum_{1 \leq i < j \leq m} \text{Var}(Y_{ij})$$

$$\leq \left( \frac{m}{2} \right)^{-1} \frac{n}{W^2}$$

$$\leq \frac{1}{12} \cdot \left( \frac{\varepsilon}{W} \right)^2$$
**ALGORITHM 2: SetSizeEstimator().**

1. $S_0 \leftarrow \emptyset$
2. for $i \in \mathbb{N}$ do
   3. Sample $a_i \in U$ uniformly
   4. if $a_i \in S_i$ then
      5. $\hat{s} \leftarrow |S_i|
      6. $\hat{N} \leftarrow 4\hat{s}^2$
   7. return $\hat{N}$
   8. $S_{i+1} \leftarrow S_i \cup \{a_i\}$

By Chebyshev inequality, it holds that

$$P\left(\frac{1}{\hat{W}} - \frac{1}{W} > \frac{\varepsilon}{2W}\right) \leq \frac{\text{Var}\left[1/\hat{W}\right]}{\left(\varepsilon/2W\right)^2} \leq \frac{1}{3}$$

Finally, for $\varepsilon \leq 1$ we have

$$(1 - \varepsilon)W \leq (1 + \varepsilon/2)^{-1}W \leq \hat{W} \leq (1 - \varepsilon/2)^{-1}W \leq (1 + \varepsilon)W$$

This means that $|1/\hat{W} - 1/W| \leq \varepsilon/(2W)$ implies $|\hat{W} - W| \leq \varepsilon W$. Thus, $P(|\hat{W} - W| \leq \varepsilon W) \geq 2/3$. □

### 2.2 Algorithms for $|U|$ Unknown

In this section, we present the algorithm NoAdvicePropEstimator, which samples elements from $U$ proportionally to their weights, and computes an $(1 \pm \varepsilon)$-approximation of $W$ with probability 2/3, without any knowledge of $n = |U|$.

NoAdvicePropEstimator takes $O(\sqrt{n} + \log n/\varepsilon^2)$ samples in expectation and works as follows. We partition $U$ into buckets such that items in one bucket have roughly the same weight. We pick one bucket such that the items in this bucket are likely to have a sufficiently large total weight. We then estimate the sum restricted to this bucket. If we are able to do that, we can estimate the total weight by looking at what fraction of the proportional samples end up in this bucket. We estimate the sum restricted to the bucket as follows. Since the weights are roughly the same for all items in the bucket, we may use rejection sampling to efficiently simulate uniform samples from the bucket. That allows us to estimate the number of items in it, up to a constant factor. We use the algorithm PropEstimator with this estimated bucket size as the advice $\hat{n}$.

#### 2.2.1 Estimating $|U|$ through Uniform Sampling

As a preliminary step, we assume that we are able to sample elements from $U$ uniformly, rather than according to their weights. Under this assumption, we introduce the algorithm SetSizeEstimator that, using $O(\sqrt{n})$ expected samples, estimates $n = |U|$ up to a constant factor with probability 2/3. The intuition behind SetSizeEstimator is fairly simple: if we sample uniformly with replacement from a universe of size $n$ and we see the first repetition after $t$ samples, then it is likely that $t \approx \sqrt{n}$.

**Theorem 2.3.** SetSizeEstimator has expected sample complexity of $O(\sqrt{n})$. It returns an estimate $\hat{N}$ such that $P(n \leq \hat{N}) \geq 2/3$ and $E[\hat{N}] = O(n)$.

**Proof.** We prove that when the algorithm aborts, it holds $P(\sqrt{n}/2 \leq \hat{s}) \geq 2/3$. The bound on $P(n \leq \hat{N})$ follows by the definition of $\hat{N}$. Define the event $E_i = \{a_i \in S_i\}$, where we define that $a_i \notin S_i$ whenever the algorithm terminates before step $i$. It then holds $P(E_i) \leq P(E_i | \bigcap_{j<i} E_j) =
ALGORITHM 3: PropBucketSampler($b$).

1. Sample $(a, w(a))$ proportionally
2. while $w(a) \not\in [2^b, 2^{b+1})$ do
3. Sample $(a, w(a))$ proportionally
4. return $(a, w(a))$

ALGORITHM 4: UnifBucketSampler($b$).

1. $(a, w(a)) \leftarrow \text{PropBucketSampler} (b)$
2. while $\text{Uniform}([0, 1]) > \frac{2^b}{w(a)}$ do
3. $(a, w(a)) \leftarrow \text{PropBucketSampler} (b)$
4. return $(a, w(a))$

$i/n$. We have

$$P \left( \frac{\sqrt{n}}{2} < \hat{s} \right) = P \left( \bigcup_{i=0}^{\sqrt{n}/2-1} E_i \right)$$

$$\leq \sum_{i=0}^{\sqrt{n}/2-1} P(E_i)$$

$$\leq \sum_{i=0}^{\sqrt{n}/2-1} \frac{i}{n}$$

$$= \frac{1}{n} \left( \frac{\sqrt{n}/2}{2} \right) < \frac{1}{6}.$$

After $\sqrt{n}$ samples, each additional sample is a repetition with probability at least $1/\sqrt{n}$. The number of iterations before the algorithm returns is thus stochastically dominated by $\sqrt{n} + \text{Geom}(1/\sqrt{n})$. We may thus bound the expectation as

$$E \left[ \hat{N} \right] = E \left[ 4\hat{s}^2 \right] \leq O \left( n + E \left[ \text{Geom}(1/\sqrt{n})^2 \right] \right) \leq O(n)$$

where the last inequality is a standard result on the second moment of the geometric random variable. $\square$

2.2.2 Simulating Uniform Sampling. We define buckets $B_i = \{a \in U \mid w(a) \in [2^i, 2^{i+1})\}$ for each $i \in \mathbb{Z}$, and we show how to sample elements uniformly from $B_i$, while allowed to sample elements proportionally to their weight $w$. First, we show how to sample elements from a bucket $B_b$ given a $b \in \mathbb{Z}$ proportionally in PropBucketSampler. We then use rejection sampling to obtain a uniform sample through UnifBucketSampler.

**Lemma 2.4.** Let $p_b = P(w(a) \in [2^b, 2^{b+1}) \mid a \in U)$ sampled proportionally. Then, the expected sample complexity of both PropBucketSampler and UnifBucketSampler is $O(1/p_b)$. PropBucketSampler returns an item from the $b$th bucket with distribution proportional to the weights. UnifBucketSampler returns an item from the $b$th bucket distributed uniformly.

**Proof.** PropBucketSampler performs samples until it samples an item $a$ from bucket $B_b$; it returns $a$. This is equivalent to sampling proportionally conditioned on $a \in B_b$. This proves that
**ALGORITHM 5:** NoAdvicePropEstimator(\(\varepsilon\)).

1. Sample \((a_1, w(a_1)), (a_2, w(a_2))\) proportionally
2. \(b_1 \leftarrow \lfloor \log w(a_1) \rfloor\) for \(i = 1, 2\)
3. \(b \leftarrow \max(b_1, b_2)\)
4. \(\tilde{n}_b \leftarrow \text{SetSizeEstimator()}\) using UnifBucketSampler(b) as sampling subroutine, with success probability \(9/10\)
5. \(\tilde{W}_b \leftarrow \text{PropEstimator}(\varepsilon/3, \tilde{n}_b)\) using PropBucketSampler(b) as sampling subroutine, with success probability \(9/10\)
6. \(\hat{P}_b \leftarrow \text{BernoulliEstimator}(\varepsilon/3)\) to estimate \(P(a \in B_b)\) with success probability \(9/10\)
7. \(\hat{W} \leftarrow \tilde{W}_b/\hat{P}_b\)
8. Return \(\hat{W}\)

the output has the claimed distribution. In each step, we finish with probability \(p_b\), independent of other steps. The expected number of steps is, therefore, \(1/p_b\). This proves the sample complexity.

Similarly, we terminate UnifBucketSampler after sampling \(a \in B_b\) and Uniform([0, 1]) \(\leq 2^b w(a)\).

Sampling until \(a \in B_b\) is equivalent to sampling item \(a\) from \(B_b\) with probability \(w(a)/A_b\) where \(A_b\) is the total weight of items in bucket \(B_b\). Therefore, \(a\) is sampled in each step with probability

\[
p_b \cdot \frac{w(a)}{A_b} \cdot \frac{2^b}{w(a)} = p_b \cdot 2^b
\]

Since this probability is the same for all items \(a \in B_b\), the resulting distribution is uniform. The rejection probability is upper-bounded by \(1/2\). Therefore, UnifBucketSampler also has an expected sample complexity of \(O(1/p_b)\). □

2.2.3 Putting It Together: Estimating \(W\) without Advice. Finally, we are ready to show the algorithm NoAdvicePropEstimator, that estimates \(W\) without relying on any advice \(\tilde{n} \geq n\). To analyze it, we first need a lemma.

To analyze NoAdvicePropEstimator, we first need a lemma:

**Lemma 2.5.** Consider \(b_1, b_2\) and \(b\) as defined in NoAdvicePropEstimator, and let \(a \in U\) be a random element sampled proportionally. Then,

\[
E \left[ \frac{1}{P_{\text{prop}}(a \in B_b \mid b)} \right] = O(\log n).
\]

Moreover, if we define \(n_b\) as the number of items in \(B_b\), we have

\[
E \left[ \frac{\sqrt{n_b}}{P_{\text{prop}}(a \in B_b \mid b)} \right] = O(\sqrt{n}).
\]

**Proof.** Throughout this proof, we assume \(a\) to be sampled proportionally. We first prove the first statement. Define \(k = \max\{j \mid B_j \neq \emptyset\}\), and set \(\mathcal{B} = \{B_j \mid k - 2 \log n < j \leq k\) and \(B_j \neq \emptyset\)\). Notice that \(|\mathcal{B}| \leq 2 \log n\) and, for each \(B_j \notin \mathcal{B}, x \in B_j\) we have

\[
P(a = x) \leq \frac{2^{k-2 \log n + 1}}{W} = 2^{-2 \log n + 1} \cdot \frac{2^k}{W} \leq \frac{2}{n^2}
\]

since there exists \(y \in B_k\) and therefore \(W \geq w(y) \geq 2^k\). Hence, sampling \(a \in U\) proportionally we have

\[
P(B_a \notin \mathcal{B}) = P(\exists j, \text{ s.t. } a \in B_j \land B_j \notin \mathcal{B})
\]
Now, we notice that for each $B_j$, it holds
\[ P \left( b \in B_j \right) \leq 2 \cdot P \left( \exists i \leq j : b_2 \in B_i \right) \]
\[ = 2 \cdot P \left( a \in B_j \right) \cdot P \left( \exists i \leq j : a \in B_i \right) \]
\[ \leq 2 \cdot P \left( a \in B_j \right) \]
where the factor two is given by the union bound, and we used $a \sim b_1 \sim b_2$. Then, we can write
\[
E \left[ \frac{1}{P \left( a \in B_b \mid b \right)} \right] = \sum_{B_j \neq \emptyset} \frac{P \left( b \in B_j \right)}{P \left( a \in B_j \right)}
\]
\[ = \sum_{B_j \in B} \frac{P \left( b \in B_j \right)}{P \left( a \in B_j \right)} + \sum_{B_j \neq \emptyset} \frac{P \left( b \in B_j \right)}{P \left( a \in B_j \right)}
\]
\[ \leq 2 \cdot |B| + \sum_{B_j \neq \emptyset} 2P \left( \exists i \leq j : a \in B_i \right)
\]
\[ \leq 2 \cdot |B| + \sum_{B_j \neq \emptyset} 2P \left( B_a \notin B \right)
\]
\[ \leq 4 \log n + 2n \cdot P \left( B_a \notin B \right)
\]
\[ \leq 4 \log n + 4 = O(\log n). \]
The fist inequality is obtained using Equation (1). The second inequality descends from the fact that $B_j \notin B$ and $i \leq j$ imply $B_i \notin B$. The last two inequalities are obtained using $n$ as an upper bound on the number of nonempty buckets $B_j$ and recalling that $P(B_a \notin B) \leq 2/n$.

Now we can prove the second statement. Denote by $n_b$ the number of elements in $B_b$ and define $\ell = \arg \max_{i \in \mathbb{Z}} n_i 2^{i/2}$. If we define $S_i = \sum_{j \leq i} \sum_{a \in B_j} w(a)$, then we can rewrite the already proven inequality $P \left( b \in B_j \right) \leq 2 \cdot P \left( a \in B_j \right) \cdot P \left( \exists i \leq j : a \in B_i \right)$ as
\[
P \left( b \in B_j \right) \leq 2 \cdot P \left( a \in B_j \right) \cdot \frac{S_j}{W}.
\]
We now prove that there exists a constant $C > 0$ such that, for all $i \in \mathbb{Z}$, it holds $S_{\ell - i} \leq C \cdot S_{\ell} \cdot 2^{-i/2}$. Notice that, by definition of $\ell$, we have $n_j \cdot 2^{j/2} \leq n_\ell \cdot 2^{\ell/2}$ for all $j \in \mathbb{Z}$. We can now bound
\[
S_{\ell - i} \leq \sum_{j \leq \ell - i} n_j \cdot 2^{j+1}
\]
\[ \leq n_\ell \cdot \sum_{j \leq \ell - i} 2^{\ell/2-j/2} \cdot 2^{j+1}
\]
\[ = 2n_\ell 2^\ell \cdot \sum_{j \leq \ell - i} 2^{j/2-\ell/2}
\]
\[ \leq 2S_\ell \cdot \sum_{j \leq \ell - i} 2^{j/2-\ell/2} \leq C \cdot S_\ell 2^{-\ell/2}.
\]
Therefore, we have $\sum_{j<\ell} S_j = O(S_\ell)$. Notice that by the definition of $\ell$ we have $n_{\ell+i} \leq n_\ell \cdot 2^{-i/2}$ for each $i \geq 0$. Now we are ready to prove our final result.

$$E \left[ \frac{\sqrt{n_i}}{P(a \in B_b \mid b)} \right] = \sum_{j \in \mathbb{Z}} P(b \in B_j) \cdot \frac{\sqrt{n_j}}{P(a \in B_j)} \leq \sum_{j \in \mathbb{Z}} 2 \frac{S_j}{W} \cdot \sqrt{n_j} \leq \sum_{j<\ell} 2 \frac{S_j}{W} \cdot \sqrt{n} + \sum_{j\geq\ell} 2 \sqrt{n_j} \leq \frac{O(S_\ell)}{W} \cdot \sqrt{n} + \sum_{i \geq 0} 2 \sqrt{n_\ell} \cdot 2^{-i/4} = O(\sqrt{n}).$$

The first inequality uses Equation (2), the second inequality is obtained splitting the series in two parts and using $S_j \leq W$. The last inequality is obtained plugging in $\sum_{j<\ell} S_j = O(S_\ell)$ and $n_{\ell+i} \leq n_\ell \cdot 2^{-i/2}$ for $i \geq 0$. □

Now we are ready to analyze NoAdvicePropEstimator.

**Theorem 2.6.** Let $\hat{W}$ be the estimate returned by NoAdvicePropEstimator. Then $\hat{W}$ is an $(1 \pm \epsilon)$-approximation of $W$ with probability $2/3$. Moreover, its expected sample complexity is

$$O\left(\frac{\sqrt{n}}{\epsilon} + \frac{\log(n)}{\epsilon^2}\right).$$

**Proof.** We start by proving correctness. Define $W_b = \sum_{x \in B_b} w(x)$ and $P_b = P(a \in B_b \mid b) = W_b/W$. Notice that $W_b$ and $P_b$ are random variables, since they depend on $b$. Now we prove that $\hat{W}$ is a $(1 \pm \epsilon)$-approximation of $W$ with probability $2/3$. Define the event $E_{\ell} = \{n \leq \hat{n}_b\}$, we have $P(E_{\ell}) \geq 9/10$ (where we use Theorem 2.3 together with probability amplification to amplify the success probability of $2/3$ to $9/10$). Define the event

$$E_2 = \{(1 - \epsilon/3)W_b \leq \hat{W}_b \leq (1 + \epsilon/3)W_b\}$$

then, we have $P(E_2 \mid E_{\ell}) \geq 9/10$ (where we use Theorem 2.2 and probability amplification). Define the event

$$E_3 = \{(1 - \epsilon/3)P_b \leq \hat{P}_b \leq (1 + \epsilon/3)P_b\}$$

then it holds $P(E_3 \mid E_2 \cap b) \geq 9/10$ (where we use Proposition 1.1 and probability amplification). On the event $E_2 \cap E_3$, it holds

$$(1 - \epsilon) \cdot W \leq \frac{1 - \epsilon/3}{1 + \epsilon/3} \cdot \frac{W_b}{P_b} \leq \frac{\hat{W}_b}{\hat{P}_b} \leq \frac{1 + \epsilon/3}{1 - \epsilon/3} \cdot \frac{W_b}{P_b} \leq (1 + \epsilon) \cdot W.$$

Then we can apply union bound and prove

$$P \left( \hat{W} < (1 - \epsilon)W \text{ or } \hat{W} > (1 + \epsilon)W \right) \leq P \left( E_2 \cup E_3 \right) \leq P \left( E_{\ell} \right) + P \left( E_2 \mid E_{\ell} \right) + P \left( E_3 \right) \leq \frac{1}{10} + \frac{1}{10} + \frac{1}{10} \leq \frac{1}{3}.$$
number of samples employed during the $i$th call to UnifBucketSampler($b$) on line 4 with $\eta_1^{(i)}$; similarly, we denote the number of samples taken during the $i$th call to PropBucketSampler($b$) on line 5 with $\eta_2^{(i)}$. We can then write

$$\sigma_1 = \sum_{i=1}^{\tau_1} \eta_1^{(i)} \quad \text{and} \quad \sigma_2 = \sum_{i=1}^{\tau_2} \eta_2^{(i)}$$

where $\tau_1$ and $\tau_2$ are the number of calls to UnifBucketSampler($b$) performed on lines 4 and 5, respectively. First, we notice that, thanks to Lemma 2.4, there exists a constant $K > 0$ such that $E[\eta_1^{(i)} \mid b], E[\eta_2^{(i)} \mid b] \leq \frac{K}{P(a \in B_k \mid b)}$. Thanks to Theorem 2.3, we have $E[\tau_1 \mid b] = O(\sqrt{n_b})$ and $\tau_2 = O(\sqrt{n_b} / \epsilon)$. Now we are ready to bind $E[\sigma_1]$ and $E[\sigma_2]$. We have

$$E[\sigma_1] = E \left[ E \left[ \sum_{i=1}^{\tau_1} \eta_1^{(i)} \mid b \right] \right] = E \left[ E \left[ \tau_1 \mid b \right] \cdot E \left[ \eta_1^{(i)} \mid b \right] \right] \leq E \left[ K \cdot O(\sqrt{n_b}) \right] = O(\sqrt{n})$$

where the first equality is by the Wald’s identity and the last equality is obtained applying Lemma 2.5. Similarly,

$$E[\sigma_2] = E \left[ E \left[ \sum_{i=1}^{\tau_2} \eta_2^{(i)} \mid b \right] \right] = E \left[ E \left[ \tau_2 \mid b \right] \cdot E \left[ \eta_2^{(i)} \mid b \right] \right] \leq E \left[ \frac{K \cdot (\sqrt{80n_b} / \epsilon + 1)}{P(a \in B_k \mid b)} \right] = O \left( \frac{\sqrt{n}}{\epsilon} + \log n \right) = O \left( \frac{\sqrt{n}}{\epsilon} \right)$$

where we used that $E[\sqrt{n_b}] \leq \sqrt{E[n_b]} \leq \sqrt{n}$, which holds thanks to Theorem 2.3 and Jensen inequality. In order to bound $E[\sigma_3 \mid b]$, recall that, thanks to Proposition 1.1, there exists a $C > 0$ such that, conditioning on the value of $b$, BernoulliEstimator($\frac{\eta}{b}$) takes in expectation at most $\frac{C}{P(a \in B_k \mid b)\epsilon^2}$ samples in order to estimate $P(a \in B_k)$. Therefore, we have

$$E[\sigma_3] = E \left[ E[\sigma_3 \mid b] \right] \leq E \left[ \frac{C}{P(a \in B_k \mid b)\epsilon^2} \right] = O \left( \frac{\log n}{\epsilon^2} \right)$$

where the last equality holds by Lemma 2.5. This concludes the proof since the total number of samples taken by NoAdvicePropEstimator is $\sigma_1 + \sigma_2 + \sigma_3$. By the bounds we have proven above, the expectation of $\sigma_1 + \sigma_2 + \sigma_3$ is as claimed. \hfill \qed

3 \quad SUM ESTIMATION BY HYBRID SAMPLING

In this section, we assume that we can sample elements both proportionally and uniformly. Again, we solve the task of providing an estimate $\hat{W}$ of $W$ such that $\hat{W}$ is a $(1 \pm \epsilon)$-approximation of $W$ with probability $2/3$.

We notice that if we take $\Theta(n \log n)$ uniform samples then, with probability $2/3$, we see every element of $U$. This simple analysis is well-known under the name of Coupon Collector problem. If we know $n$ (or any constant-factor approximation of it) we may simply take $\Theta(n \log n)$ samples, assume we have seen every element at least once, and compute $W$ exactly. If we do not know $n$, we first use SetSizeEstimator to compute an estimate $\hat{N}$ of $n$ accurate up to a constant factor. If SetSizeEstimator succeeds, then taking $O(\hat{N} \log \hat{N})$ samples suffices to collect all elements.

Therefore, it is sufficient to show an algorithm with complexity $T(n, \epsilon)$ to obtain a complexity of the form $O(\min(T(n, \epsilon), n \log n))$, as we can just run the coupon-collector algorithm in parallel and take the result provided by the first of the two algorithms to finish its execution. In what follows
we only show how to achieve a complexity of $O(\sqrt{n}/\varepsilon^{4/3})$ when $|U|$ is known, and of $O(\sqrt{n}/\varepsilon)$ when $|U|$ is unknown. As a consequence, the complexities that we achieve in this settings are $O(\min(\sqrt{n}/\varepsilon^{4/3}, n \log n))$ and of $O(\min(\sqrt{n}/\varepsilon, n \log n))$ respectively.

### 3.1 Algorithms for $|U|$ Known

In this section, we show an algorithm that, given $n = |U|$, returns a $(1 \pm \varepsilon)$-approximation of $W$ with probability $2/3$ using $O(\sqrt{n}/\varepsilon^{4/3})$ samples. First, we introduce a subroutine that uses harmonic mean to estimate the average weight $W/n$; then we combine it with PropEstimator to obtain the main algorithm of this section.

#### 3.1.1 Harmonic-mean-based Estimator

Here we show the algorithm HarmonicEstimator$(\varepsilon, \hat{\theta}, \phi)$ that returns an $(1 \pm \varepsilon)$-approximation $\hat{\theta}$ of $W/n$ with probability $2/3$. HarmonicEstimator$(\varepsilon, \hat{\theta}, \phi)$ takes as advice an upper bound on the average weight $\hat{\theta} \geq W/n$, and a parameter $\phi$ such that we expect $P_{unif}(w(a) \geq \phi)$ not to be too small and $\hat{\theta}/\phi$ not to be too large. A more formal statement follows.

To see the intuition behind this algorithm, consider the case when $\phi \leq w(a)$ for all $a \in U$. It then holds $\hat{\theta} \approx 1$. We take $k$ samples, and let $1/H$ be the harmonic mean of the weights of the sampled items. We have $E[H] = n/W$, and $\hat{\theta} \approx 1/H$ as $\hat{\theta} \approx 1$. Unfortunately, $H$ might have a high variance due to elements having very small weights. To fix this, we consider a parameter $\phi$ such that $w(a) < \phi$ for some $a \in U$. Instead of $E[H] = n/W$, we then have $E[H] = n'/W$ for $n' = |\{a \in U \mid w(a) \geq \phi\}|$. We then multiply $1/H$ by $\hat{\theta}$ in order to adjust for the fraction of items that were ignored. Note that, while increasing $\phi$, the variance of $1/H$ decreases; however, also $n'$ decreases and this means that computing an estimate $\hat{\theta} \approx n'/n$ requires more samples. This introduces a tradeoff between the algorithm’s complexity and the variance of $H$.

**Lemma 3.1.** Given parameters $\hat{\theta}$ and $0 < \varepsilon < 1$, HarmonicEstimator$(\varepsilon, \hat{\theta}, \phi)$ has expected sample complexity $O((1 + \hat{\theta}^2)/(p \varepsilon^2))$ where $p = P_{unif}(w(a) \geq \phi)$. It returns an estimate $\hat{W}$ such that $P(\hat{\theta} < W/(20n)) \leq 1/20$. If, moreover, $\hat{\theta} \geq W/n$, then $P(|\hat{W} - W| \leq \varepsilon W) \geq 2/3$.

**Proof.** We start by proving that $\hat{\theta}$ is a $(1 + \varepsilon)$-approximation of $W/n$ with probability $2/3$ when $\hat{\theta} \geq W/n$. Define the event $\mathcal{E} = \{\hat{\theta} \geq W/n\}$. By Proposition 1.1 and using
probability amplification, we have \( P(\mathcal{E}) \geq 9/10 \). For each \( i = 1 \ldots k \) we have
\[
E[b_i] = \sum_{a \in U, \ w(a) \geq \phi} \frac{1}{w(a)} \cdot \frac{w(a)}{W} = \frac{n_{\geq \phi}}{W} = \frac{p \cdot n}{W}
\]
where \( n_{\geq \phi} \) is the number of elements in \( a \in U \) with \( w(a) \geq \phi \). Notice that this implies \( E[H] = p \cdot n/W \). Moreover, for each \( i = 1 \ldots k \)
\[
\text{Var}(b_i) \leq E\left[b_i^2\right] = \sum_{a \in U, \ w(a) \geq \phi} \frac{1}{w^2(a)} \cdot \frac{w(a)}{W} \leq \frac{n_{\geq \phi}}{\phi \cdot W} = \frac{p \cdot n}{\phi \cdot W}.
\]
Conditioning on \( \mathcal{E} \), we have that \( (1 - \varepsilon/3)\hat{\theta} \leq p \). The way we have set \( k \) allows us to bound
\[
\text{Var}(H \mid \mathcal{E}) = \frac{\text{Var}(b_i)}{k} = \frac{p \cdot n}{\phi \cdot W} \cdot \frac{\varepsilon^2(1 - \varepsilon/3)^2}{45} \cdot \frac{\phi}{\hat{\theta}} \leq \left(\frac{p \cdot n}{W}\right)^2 \cdot \frac{\varepsilon^2}{45} = E[H]^2 \cdot \frac{\varepsilon^2}{45}
\]
where we used that \( \hat{\theta} \geq W/n \). It holds \( E[H \mid \mathcal{E}] = E[H] \). We may thus apply Chebyshev’s inequality to get
\[
P\left(\left|H - E[H]\right| > \frac{\varepsilon}{3 E[H]} \mid \mathcal{E}\right) \leq \frac{\text{Var}(H \mid \mathcal{E})}{(\varepsilon E[H])^2} \leq \frac{9}{45} = \frac{1}{5}
\]
Since \( \varepsilon < 1 \), we have \( (1 - \varepsilon/3)^{-1} \leq 1 + \varepsilon/2 \) and \( (1 + \varepsilon/3)^{-1} \geq 1 - \varepsilon/2 \). Therefore,
\[
P\left(\frac{1}{H} - \frac{1}{E[H]} > \frac{\varepsilon}{2 E[H]} \mid \mathcal{E}\right) \leq \frac{1}{5}.
\]
Again, since \( \varepsilon < 1 \), we have \( (1 + \varepsilon/3) \cdot (1 + \varepsilon/2) \leq 1 + \varepsilon \) and \( (1 - \varepsilon/3) \cdot (1 - \varepsilon/2) \geq 1 - \varepsilon \). Hence, using the union bound
\[
P\left(\frac{\hat{\theta}}{H} - \frac{p}{E[H]} > \varepsilon \cdot \frac{p}{E[H]} \mid \mathcal{E}\right) \leq P(\mathcal{E}) + P\left(\left|\frac{1}{H} - \frac{1}{E[H]}\right| > \frac{\varepsilon}{2 E[H]} \mid \mathcal{E}\right) \leq \frac{1}{3}.
\]
Since \( p/E[H] = W/n \), we have that the estimate \( \hat{\theta} = \hat{\theta}/H \) is a \( (1 + \varepsilon) \)-approximation of \( W/n \) with probability \( \geq 2/3 \).

We now argue the sample complexity. The expected number of samples used on line 1 is by Proposition 1.1 equal to \( O(1/(p \varepsilon^2)) \). In the rest of the algorithm, we use \( k \) samples. It holds
\[
E[k] = E\left[\frac{1}{\hat{\theta}}\right] \cdot O\left(\frac{\hat{\theta}}{p \phi \varepsilon^2}\right) = O\left(\frac{\hat{\theta}}{p \phi \varepsilon^2}\right)
\]
where the second equality holds by Proposition 1.1. The sample complexity is thus as claimed.

Finally, we prove that, regardless of \( \hat{\theta} \), it holds \( P(\hat{\theta} < W/(20n)) \leq 1/20 \). It holds \( E[H/\hat{\theta}] = E[H]E[1/\hat{\theta}] = n/W \) where \( E[1/\hat{\theta}] = 1/p \) by Proposition 1.1. Therefore, by the Markov’s inequality
\[
P\left(\frac{\hat{\theta}}{H} \leq \frac{W}{20n}\right) = P\left(\frac{H}{\hat{\theta}} \geq \frac{20n}{W}\right) \leq \frac{1}{20}.
\]
\[
\square
\]
ALGORITHM 7: HybridEstimator(n, ε).

Abort this algorithm if it uses more than $C \frac{n^{1/3}}{\varepsilon^{2/3}}$ samples, where $C$ is a large enough constant.
1. Find $\theta$ such that $P(\frac{n^{2/3}}{\varepsilon^{2/3}} \leq |\{a \in U | w(a) \geq \theta\}| \leq 2 \cdot \frac{n^{2/3}}{\varepsilon^{2/3}}) \geq \frac{19}{20}$
2. $\hat{\rho} \leftarrow$ BernoulliEstimator($P_{prop}(w(a) \geq \theta), \frac{\varepsilon}{2}$) with success probability 19/20
3. if $\hat{\rho} \geq 1/2$
4. $\tilde{n}_{\geq \theta} \leftarrow 2 \cdot \frac{n^{2/3}}{\varepsilon^{2/3}}$
5. $\hat{W}_{\geq \theta} \leftarrow$ PropEstimator($\tilde{n}_{\geq \theta}, \frac{\varepsilon}{2}$) with success probability 19/20, run on proportional samples conditioned on $w(a) \geq \theta$, obtained by rejecting elements with $w(a) < \theta$
6. $\hat{W} \leftarrow \hat{W}_{\geq \theta}/\hat{\rho}$
7. else
8. $\hat{\rho} \leftarrow$ HarmonicEstimator($\varepsilon, 3\theta, \theta$) with success probability 19/20
9. $\hat{W} \leftarrow n \cdot \hat{\rho}$
10. return $\hat{W}$

3.1.2 Combining the Two Algorithms. Here, we combine HarmonicEstimator with PropEstimator to obtain HybridEstimator. It works in the hybrid setting and provides a $(1 \pm \varepsilon)$-approximation of $W$ with probability 2/3, using in expectation $O(\sqrt{n}/\varepsilon^{1/3})$ samples. While analyzing HybridEstimator, we can restrict ourselves to $\varepsilon \geq 8/\sqrt{n}$. Indeed, for very small values of $\varepsilon$ (namely, $\varepsilon \leq 1/(\sqrt{n} \log n)$ we use the coupon-collector algorithm, and for intermediate values of $\varepsilon$ (namely, $1/(\sqrt{n} \log n) < \varepsilon < 8/\sqrt{n}$) we use PropEstimator($n, \varepsilon$). The coupon collector algorithm gives a sample complexity of $O(n \log n)$, that is better than $O(\sqrt{n}/\varepsilon^{1/3})$ for $\varepsilon < 1/(\sqrt{n} \log n)$. PropEstimator gives a sample complexity of $O(\sqrt{n}/\varepsilon)$, that is better than $O(\sqrt{n}/\varepsilon^{1/3})$ for $\varepsilon < 8/\sqrt{n}$.

To implement line 1 it is sufficient to sample uniformly $120 n^{1/3} \varepsilon^{2/3}$ elements of $U$ and define $\theta$ as the element with the 180-th largest weight. (Note that for $\varepsilon \geq 8/\sqrt{n}$ we have $120 n^{1/3} \varepsilon^{2/3} \geq 480$, so this is well-defined.) Let $k$ be such that there are $k$ elements $a \in U$ with $w(a) \geq \theta$. A standard analysis using Chebyshev inequality shows that $k$ is concentrated around $\frac{3}{2} n^{1/3}/\varepsilon^{2/3}$.

THEOREM 3.2. Given $\varepsilon$ such that $8/\sqrt{n} \leq \varepsilon < 1$, HybridEstimator($n, \varepsilon$) uses $O(\sqrt{n}/\varepsilon^{1/3})$ samples. With probability at least 2/3, the returned estimate $\hat{W}$ is a $(1 \pm \varepsilon)$-approximation of $W$.

PROOF. We first analyze a variant of the algorithm that does not abort. (That is, an algorithm with the same pseudocode except for the abortion condition removed.) We now show that this algorithm returns a $(1 \pm \varepsilon)$-approximation with a probability of at least 5/6.

The scheme summarized above to find $\theta$ at line 1 succeeds with a probability of at least 19/20. We call this event $E_1$. Define the event $E_2 = \{\hat{\rho} \text{ is a } (1 \pm \varepsilon/3)\text{-approximation of } \rho\}$. It holds $P(E_2) \geq 19/20$.

We now consider the case $\hat{\rho} \geq 1/2$. On line 5, we employ the algorithm PropEstimator. Whenever it performs a sample, we simulate a proportional sample from the set $U_\theta = \{a \in U | w(a) \geq \theta\}$ by sampling until we sample item $a$ such that $w(a) \geq \theta$. It is easy to see that the distribution obtained with this sampling scheme is exactly the proportional distribution on the set $U_\theta$. Conditioning on $E_1$, $\tilde{n}_{\geq \theta}$ is a valid advice and (by Theorem 2.2) PropEstimator returns a $(1 \pm \varepsilon/3)$-approximation of $W_\theta = \sum_{w(a) \geq \theta} w(a)$ with a probability of at least 2/3. We amplify this probability to 19/20. Hence, we have

$$P\left(\left\{\hat{W}_\theta \text{ is a } (1 \pm \varepsilon)\text{-approximation of } W_{\geq \theta}\right\} \cap E_2\right) \geq 1 - \left(\frac{1}{20} + P(E_1) + P(E_2)\right) \geq \frac{5}{6}.$$
Algorithm 8: NoAdviceHybridEstimator(ε).

1. \( \hat{n} \leftarrow \text{SetSizeEstimator}() \), with success probability 5/6 (using uniform sampling)
2. \( \hat{W} \leftarrow \text{PropEstimator}(\hat{n}, \varepsilon) \), with success probability 5/6 (using proportional sampling)
3. return \( \hat{W} \)

On this event, since \( \varepsilon < 1 \), we have
\[
(1 - \varepsilon)W \leq \frac{1 - \varepsilon/3}{1 + \varepsilon/3} W \leq \hat{W} \leq \frac{1 + \varepsilon/3}{1 - \varepsilon/3} W \leq (1 + \varepsilon)W.
\]

Now we analyse the case \( \hat{p} < 1/2 \). Whenever \( 3\theta \geq W/n \), HarmonicEstimator(\( \varepsilon \), \( 3\theta \), \( \theta \)) returns a \((1 \pm \varepsilon)\)-approximation of \( W/n \) with probability 2/3 (by Lemma 3.1). We amplify that probability to 20/21. Now we argue that, conditioning on \( \mathcal{E}_2 \), we have \( 3\theta \geq W/n \). Define \( p = P_{\text{prop}}(w(a) \geq \theta) \), then (on \( \mathcal{E}_2 \)) we have \( p \leq (1 + \varepsilon/3)\hat{p} \leq (1 + 1/3)1/2 \leq 2/3 \). Hence, \( n\theta \geq (1-p)W \geq W/3 \) and thus \( 3\theta \geq W/n \), where the first inequality is obtained using \( \sum w(a) < \theta w(a) = (1-p)W \). Applying union bound gives that HarmonicEstimator(\( \varepsilon \), \( 3\theta \), \( \theta \)) succeeds with a probability of at least \( 1 - (1/20 + P(\mathcal{E}_2)) \geq 5/6 \).

Therefore, regardless of the value of \( \hat{p} \), we have shown that \( \hat{W} \) is a \((1 \pm \varepsilon)\)-approximation of \( W \) with a probability of at least 5/6. Thus, the modified algorithm without abortion is correct with a probability of at least 5/6. We now argue that the probability that Algorithm 7 is aborted is at least 1/6 (for \( C \) large enough). By the union bound, its success probability is at least 2/3.

In what follows, we compute how many samples are taken on each line. On line 1, we use only \( 120n^{1/3}e^{2/3} \) samples. Thanks to Proposition 1.1, BernoulliEstimator on line 2 uses \( O(1/ (p\varepsilon^2)) \) samples in expectation. In what follows, we condition on \( \mathcal{E}_1 \cap \mathcal{E}_2 \). It holds \( p \geq P_{\text{unif}}(w(a) \geq \theta) \) and thus we have \( p \geq \frac{1}{e^{2/3}n^{1/3}} \). The number of samples used on line 2 is then in expectation \( O(n^{1/3} \varepsilon e^{1/3}) \).

By the (conditional) Markov’s inequality the probability that we use more than \( C_1 n^{1/3} / \varepsilon e^{1/3} \) is at most 1/30, for some \( C_1 \) large enough. PropEstimator uses \( O(\sqrt{n^{2/3} \varepsilon^{2/3} / \varepsilon}) = O(n^{1/3} / \varepsilon^{1/3}) \) samples in the worst case. The rejection causes only constant factor expected slowdown. Again, by the (conditional) Markov’s inequality, for \( C_2 \) large enough, we use more than \( C_2 n^{1/3} / \varepsilon^{1/3} \) with probability at most 1/30. Since we have \( p \geq \frac{1}{e^{2/3}n^{1/3}} \), on line 8 HarmonicEstimator takes \( O(n^{1/3} / \varepsilon^{1/3}) \) samples in expectation (by Lemma 3.1). Thus, there exists a constant \( C_3 \) such that on line 8 we use more than \( C_3 n^{1/3} / \varepsilon^{1/3} \) samples with a probability of at most 1/30.

Set \( C = 120 + C_1 + C_2 + C_3 \). It then holds by the union bound that we use more than \( C\sqrt{n} / \varepsilon^{1/3} \) samples (and thus abort) with a probability of at most \( P(\mathcal{E}_1) + P(\mathcal{E}_2) + 1/30 + 1/30 = 1/6 \).

3.2 Algorithms for \(|U| \) Unknown

In this section we show an algorithm that, without any knowledge of \( n = |U| \), provides a \((1 \pm \varepsilon)\)-approximation of \( W \) with a probability of 2/3 using \( O(\sqrt{n} / \varepsilon) \) samples. This complexity is strictly higher than the one of Section 3.1 for \( \varepsilon = \omega(1/\sqrt{n}) \). However, it is near-optimal when we do not know \( n \), as we will see in Section 4.

Theorem 3.3. NoAdviceHybridEstimator(\( \varepsilon \)) uses in expectation \( O(\sqrt{n} / \varepsilon) \) samples and, with a probability of at least 2/3, we have \( (1 - \varepsilon)W \leq \hat{W} \leq (1 + \varepsilon)W \).

Proof. By Theorem 2.3, SetSizeEstimator takes \( O(\sqrt{n}) \) samples and returns \( \hat{n} \) such that \( n \leq \hat{n} \) with probability 2/3. We amplify this probability to 5/6. Conditioning on \( n \leq \hat{n} \), PropEstimator returns \((1 \pm \varepsilon)\)-estimate of \( W \) with a probability of at least 2/3 by Theorem 2.2. We amplify this probability to 5/6. By the union bound, the algorithm returns a \((1 \pm \varepsilon)\)-estimate with a probability of at least 2/3.
Moreover, by Theorem 2.3 and Lemma 1.2, \( E[\tilde{n}] = O(n) \). By the Jensen inequality, PropEstimator then uses in expectation \( O(\sqrt{n}/\varepsilon) \) samples.

### 3.2.1 Coupon Collector Algorithm for Unknown \(|U|\)

We can design a simple \( O(n \log n) \) time algorithm that computes \( W \) exactly with probability \( 2/3 \) in the hybrid setting, without knowing \( n = |U| \). We denote this algorithm with NoAdviceCouponCollector. First, we use SetSizeEstimator to give an estimate \( \hat{N} \) of \( n \) such that \( n \leq \hat{N} \leq Cn \) for some constant \( C \), with probability \( \geq 5/6 \). This guarantee holds thanks to Theorem 2.3 combined with probability amplification and Markov’s inequality. We say that SetSizeEstimator succeeds if that is the case. Then, by standard coupon collector we know that, conditioned on SetSizeEstimator succeeding, \( O(\hat{N} \log \hat{N}) \) samples suffice to collect all elements in \( U \), with probability \( 5/6 \). Then, we simply compute the sum of their weights and by union bound this sum equals \( W \) with probability \( \geq 2/3 \).

We can use this to ensure we never use more than \( O(n \log n) \) samples in the hybrid setting. Specifically, we may execute in parallel this algorithm together with one of the above algorithms and abort when one of them returns an estimate. If one wishes to implement this in practice, it is possible to instead do the following. Given the parameter \( \varepsilon \), compute a threshold \( n_0 \) such that we would like to (if we knew \( n \)) execute NoAdviceCouponCollector if \( n \leq n_0 \) and NoAdviceHybridEstimator if \( n > n_0 \). We then run NoAdviceCouponCollector and we abort if in case we find \( n_0 \) distinct elements. If this happens, we then run NoAdviceHybridEstimator.

### 4 LOWER BOUNDS

In this section, we give lower bounds for the problems we study in this article. The proofs of the lower bounds all follow a common thread. In what follows, we use the term risk to refer to the misclassification probability of a classifier.

The roadmap of all our proofs follows. First, we define two different instances of the problem \((U_1, w_1) \) and \((U_2, w_2) \) such that a \((1 \pm \varepsilon)\)-approximation of \( W \) is sufficient to distinguish between them. Then, we define our hard instance as an equally likely mixture of the two, namely we take \((U_1, w_1) \) with probability \( 1/2 \) and \((U_2, w_2) \), otherwise. We denote these events by \( \mathcal{E}_1 \) and \( \mathcal{E}_2 \), respectively. Second, we show that a Bayes classifier\(^{8}\) cannot distinguish between the two cases with probability \( 2/3 \) using too few samples; since Bayes classifiers are risk-optimal this implies that no classifier can have risk less than \( 2/3 \) while using the same number of samples. To show that a Bayes classifier has a certain risk, we study the posterior distribution. Let \( S \) represent the outcome of the samples. Since the prior is uniform, applying Bayes theorem gives

\[
\frac{P(\mathcal{E}_1|S)}{P(\mathcal{E}_2|S)} = \frac{P(S|\mathcal{E}_1)}{P(S|\mathcal{E}_2)}.
\]

We call this ratio \( \mathcal{R}(S) \) and show that \( \mathcal{R}(S) \approx 1 \) with probability close to 1. When \( \mathcal{R}(X) \approx 1 \), the posterior distribution is very close to uniform, and this entails a risk close to \( 1/2 \). First, we show this formally with some technical lemmas, and then we instantiate our argument for each of the studied problems.

**Lemma 4.1.** Given two disjoint events \( \mathcal{E}_1, \mathcal{E}_2 \) such that \( P(\mathcal{E}_1) = P(\mathcal{E}_2) = 1/2 \) and a random variable \( X \), we define the ratio

\[
\mathcal{R}(x) = \frac{P(X = x | \mathcal{E}_1)}{P(X = x | \mathcal{E}_2)}.
\]

\(^{8}\) Suppose we have a partition of the probability space \( \Omega \) into events \( \mathcal{E}_1, \ldots, \mathcal{E}_k \). We want to guess which event happened, based on observation \( X \). Bayes classifier outputs as its guess \( \mathcal{E}_\ell \) that maximizes \( P(\mathcal{E}_\ell | X) \).
Notice that $\mathcal{R}(X)$ is a random variable since it depends on the outcome of $X$. If
\[ P \left( \frac{7}{8} \leq \mathcal{R}(X) \leq \frac{8}{7} \right) \geq \frac{14}{15} \]
then any classifier taking $X$ and classifying $\mathcal{E}_1, \mathcal{E}_2$ has risk $\geq 2/5$.

**Proof.** First, we notice that since Bayes classifiers (BC) are risk-optimal, then it is sufficient to prove our statement for a Bayes classifier. Define $p_i = P(\mathcal{E}_i \mid X)$ for $i = 1, 2$, then Bayes classifier simply returns $\arg\max_{i=1,2} p_i$. By Bayes theorem and because $P(\mathcal{E}_1) = P(\mathcal{E}_2)$ we have $\mathcal{R}(X) = p_1/p_2$. If $7/8 \leq \mathcal{R}(X) \leq 8/7$, then
\[
\frac{1}{p_2} = \frac{p_1 + p_2}{p_2} = 1 + \mathcal{R}(X) \in \left[ \frac{15}{8}, \frac{15}{7} \right]
\]
and the same holds for $p_1$, therefore, $7/15 \leq p_1, p_2 \leq 8/15$. Hence, conditioning on $7/8 \leq \mathcal{R}(X) \leq 8/7$, the probability of correct classification is at most $8/15$. Finally, we have
\[
P(\text{BC returns correct answer}) \leq P \left( \mathcal{R}(X) < \frac{7}{8} \lor \mathcal{R}(X) > \frac{8}{7} \right) + P \left( \text{BC returns correct answer} \mid \mathcal{R}(X) \leq \frac{8}{7} \right)
\]
\[
\leq \frac{1}{15} + \frac{8}{15} = \frac{3}{5}.
\]
\[\square\]

Before proving the main theorems, we need several lemmas.

**Lemma 4.2.** Consider an instance of our problem $(U, w)$ so that $n = |U|$ and $w(a) = 1$ for each $a \in U$. We take $m$ independent samples and denote by $\ell$ the number of distinct elements obtained. Then, $\text{Var}(\ell) \leq m^2/n$. Moreover, for each $\delta > 0$
\[
P \left( \left| \ell - \mathbb{E}[\ell] \right| \geq \frac{1}{\sqrt{\delta}} \cdot \frac{m}{\sqrt{n}} \right) \leq \delta.
\]

Note that, since all weights are the same, the proportional sampling is equivalent to sampling uniformly from $U$.

**Proof.** We use the Efron–Stein inequality to prove a bound on the variance of $\ell$. Let $X_i$ be the $i$-th sample. Now $\ell$ is a function of $X_1, \ldots, X_m$ and we write it as $\ell = f(X_1, \ldots, X_m)$. Let $X'_1, \ldots, X'_m$ be an independent copy of $X_1, \ldots, X_m$. Let $\ell'_i = f(X_1, \ldots, X_{i-1}, X'_i, X_{i+1}, \ldots, X_m)$. It clearly holds that $|\ell - \ell'_i| \in \{0, 1\}$, moreover $\ell - \ell'_i = 1$ if and only if $X_i$ does not collide with any $X_j$ for $j \neq i$ and $X'_i$ does collide with some $X_k$ for $k \neq i$. It holds that $|\{X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_m\}| \leq m$. Therefore, the probability that a $X'_i$ lies in this set is $\leq m/n$. Since $\ell$ and $\ell'_i$ are symmetric, we have that also $\ell - \ell'_i = -1$ with probability $\leq m/n$. Hence, $\mathbb{E}[|\ell - \ell'_i|^2] = P(|\ell - \ell'_i| = 1) \leq 2m/n$. Applying Efron-Stein we get
\[
\text{Var}(\ell) \leq \frac{1}{2} \cdot \sum_{i=1}^{m} \mathbb{E} \left[ (\ell - \ell'_i)^2 \right] \leq \frac{m^2}{n}
\]
Now we just plug this bound on the variance into Chebyshev inequality and we get the desired inequality. \[\square\]
Fingerprints. Given a sample \( S \), we define its fingerprint \( F \) as the set of tuples \( (c_a, w(a)) \) where for each distinct item \( a \) in \( S \), we add to \( F \) such a tuple with \( c_a \) being equal to the number of copies of \( a \) in \( S \). Having a fingerprint of \( S \) is sufficient for any algorithm, oblivious of \( (U, w) \), to produce a sample \( S' \) that is equal to \( S \), up to relabeling of the elements. Since the only allowed queries are testing equality of two items and the weight query, one may easily prove that the execution of the algorithm on these two samples is the same (indeed, these two samples are indistinguishable by the equality and weight queries). Therefore, we can safely assume that an algorithm in the proportional setting takes as an input the fingerprint \( F \) of \( S \), rather than \( S \). For algorithms in the hybrid setting, we can assume that it takes as input separately the fingerprint of the proportional and the fingerprint of the uniform samples.

Lemma 4.3. Let us have parameters \( n \) and \( \epsilon < 1/3 \). Let \( N = n \) with probability \( 1/2 \), and \( N = (1-\epsilon)n \) otherwise. Consider the random instance of the sum estimation problem \( (U, w) \) with \( |U| = N \) and \( w(a) = 1 \) for each \( a \in U \). Consider a sample of size \( m \) and its fingerprint \( F_m = \{(c_i, w(a_i))\}_{i=1}^{\ell} \) and define the ratio

\[
\mathcal{R}(F_m) = \frac{P(F_m | N = n)}{P(F_m | N = (1-\epsilon)n)}.
\]

If \( m = o(\sqrt{n}/\epsilon) \) and \( m = o(n) \), then

\[
P\left( \frac{98}{100} \leq \mathcal{R}(F_m) \leq \frac{100}{98} \right) \geq \frac{99}{100}
\]

for \( n \) large enough.

Proof. We can explicitly compute the likelihood of a given fingerprint \( F_m = \{(c_i, w(a_i))\}_{i=1}^{\ell} \) where \( \ell \) is the number of distinct elements as

\[
P(F_m | N = r) = \frac{\ell!}{r^m} \binom{r}{\ell} \left( \frac{m}{c_1 \ldots c_\ell} \right)
\]

\[
= \frac{1}{r^{m-\ell}} \prod_{i=1}^{\ell-1} \left( 1 - \frac{i}{r} \right) \left( \frac{m}{c_1 \ldots c_\ell} \right)
\]

and, therefore,

\[
\mathcal{R}(F_m) = (1 - \epsilon)^{m-\ell} \cdot \prod_{i=1}^{\ell-1} \frac{1 - i/n}{1 - i/(1 - \epsilon)n}
\]

\[
= (1 - \epsilon)^{m-\ell} \cdot \prod_{i=1}^{\ell-1} \left( 1 + \frac{\epsilon i}{(1 - \epsilon)n - i} \right)
\]

Note that \( \mathcal{R}(F_m) \) depends only on \( \ell \). From now on we denote it with \( \mathcal{R}(\ell) \).

Now we define an interval \([a, b]\) such that \( P(\ell \in [a, b]) \geq 99/100 \). To do so, we first compute the expectation of \( \ell \) and then use the concentration bound of Lemma 4.2. We prove that

\[
E[\ell | N = (1-\epsilon)n] \leq E[\ell | N = n] \leq E[\ell | N = (1-\epsilon)n] + O\left( \frac{m^2}{n} \right).
\]

The expression of \( E[\ell | N = n] \) is given by

\[
E[\ell | N = n] = n \cdot \left( 1 - \left( 1 - \frac{1}{n} \right)^m \right)
\]
since each item is not sampled with probability \((1 - \frac{1}{n})^m\). From this expression, it is apparent that \(E[\ell | N = n]\) is increasing in \(n\). Expanding this formula, we get
\[
E[\ell | N = n] - E[\ell | N = (1 - \epsilon)n] = \epsilon \frac{m^2}{2n} + O\left(\frac{m^3}{n^2}\right) = O\left(\frac{m^2}{n}\right)
\]
where the last estimate uses the \(m = o(n)\) assumption. Now we define \(a = E[\ell | N = (1 - \epsilon)n] - 10m/\sqrt{n}\) and \(b = E[\ell | N = n] + 10m/\sqrt{n}\). Using the last result we proved, we have
\[
|b - a| = 20 \frac{m}{\sqrt{n}} + O\left(\frac{m^2}{n}\right) = o\left(\frac{1}{\epsilon}\right).
\]
Note that, like all asymptotics in this proof, the \(o(1/\epsilon)\) is for the limit \(n \to +\infty\) and makes sense even for \(\epsilon\) being a constant. Thanks to Lemma 4.2 we have \(P(\ell \notin [a, b] | N = n) \leq 1/100\) and \(P(\ell \notin [a, b] | N = (1 - \epsilon)n) \leq 1/100\), and therefore \(P(\ell \notin [a, b]) \leq 1/100\).

Now we give bounds on \(\mathcal{R}(a)\) and \(\mathcal{R}(b)\). It is apparent from the explicit formula above that \(\ell \mapsto \mathcal{R}(\ell)\) is an increasing function. We have
\[
\mathcal{R}(a) \cdot \frac{99}{100} \leq \mathcal{R}(a) \sum_{k \in [a, b] \cap \mathbb{Z}} P(\ell = k | N = (1 - \epsilon)n) \leq \sum_{k \in [a, b] \cap \mathbb{Z}} \mathcal{R}(k) P(\ell = k | N = (1 - \epsilon)n) \leq \sum_{k \in [a, b] \cap \mathbb{Z}} P(\ell = k | N = n) \leq 1
\]
and thus, \(\mathcal{R}(a) \leq 100/99\). Analogously,
\[
\frac{1}{\mathcal{R}(b)} \cdot \frac{99}{100} \leq \frac{1}{\mathcal{R}(b)} \sum_{k \in [a, b] \cap \mathbb{Z}} P(\ell = k | N = n) \leq \sum_{k \in [a, b] \cap \mathbb{Z}} \frac{1}{\mathcal{R}(k)} P(\ell = k | N = n) \leq \sum_{k \in [a, b] \cap \mathbb{Z}} P(\ell = k | N = (1 - \epsilon)n) \leq 1
\]
and thus, \(\mathcal{R}(b) \geq 99/100\). We now have an upper bound on \(\mathcal{R}(a)\) and a lower bound on \(\mathcal{R}(b)\). However, we need a lower bound on \(\mathcal{R}(a)\) and an upper bound on \(\mathcal{R}(b)\) (that is, the other way around). For each \(k < m\) we have
\[
\frac{\mathcal{R}(k + 1)}{\mathcal{R}(k)} = \frac{1}{1 - \epsilon} \cdot \left(1 + \frac{ek}{(1 - \epsilon)n - k}\right) \leq (1 + 2\epsilon) \cdot \left(1 + \frac{2\epsilon m}{n}\right) \leq 1 + 3\epsilon
\]
for \(n\) large enough, where we used \(k < m = o(n)\) and \(\epsilon \leq 1/3\). Hence,
\[
\mathcal{R}(b) \leq \mathcal{R}(a) \cdot (1 + 3\epsilon)^{|[b-a]|} \leq \frac{100}{99} \cdot e^{3\epsilon |[b-a]|} \leq \frac{100}{98} \cdot e^{o(1)} \leq \frac{100}{98}
\]
where the last inequality holds for \(n\) large enough because \(m = o(\sqrt{n}/\epsilon)\).
\[
\mathcal{R}(a) \geq \mathcal{R}(b) \cdot (1 + 3\epsilon)^{-|[b-a]|} \geq \frac{99}{100} \cdot e^{-3\epsilon |[b-a]|}
\]
Finally, we have for $n$ large enough that
\[
P\left(\mathcal{R}(\ell) \notin \left[\frac{98}{100}, \frac{100}{98}\right]\right) \leq P(\ell \notin [a, b]) \leq \frac{1}{100}.
\]

Using the same approach as Lemma 4.3, we prove a similar result for the task of estimating the bias $p$ of a Bernoulli random variable up to an additive $\epsilon$. In this setting, we provide an asymptotic lower bound on the number of samples, where the asymptotics are meant for the limit $(p, \epsilon) \to 0$.

The following lemma is folklore and it is implied by Fact 2.2 in [4]. Here we prove it for the sake of completeness, without claiming any original contribution.

**Lemma 4.4.** Let $0 < \epsilon < p$ and set $q = p$ with probability $1/2$, and $q = p - \epsilon$, otherwise. Let $X_1, \ldots, X_m$ be a sequence of i.i.d. Bernoulli random variables with bias $q$, and let $\ell = \sum_{i=1}^m X_i \sim \text{Bin}(m, q)$. Define the ratio
\[
\mathcal{R}(\ell) = \frac{P(\ell \mid q = p)}{P(\ell \mid q = p - \epsilon)}.
\]
If $m = o(p/\epsilon^2)$ then
\[
P\left(\frac{98}{100} \leq \mathcal{R}(\ell) \leq \frac{100}{98}\right) \geq \frac{99}{100},
\]
for $p$ (and thus also $\epsilon$) small enough.

**Proof.** We follow the same scheme we adopted in the proof of Lemma 4.3. First, we compute $\mathcal{R}(\ell)$ explicitly
\[
\mathcal{R}(\ell) = \frac{p^\ell (1-p)^{m-\ell}}{(p-\epsilon)^\ell (1-p+\epsilon)^{m-\ell}} = \frac{(1 + \frac{\epsilon}{1-p})^{\ell-m}}{(1 - \frac{\epsilon}{p})^{\ell}}.
\]
We have $E[\ell \mid q = p] = mp$, $E[\ell \mid q = p - \epsilon] = m(p-\epsilon)$, $\text{Var}(\ell \mid q = p) \leq mp$, and $\text{Var}(\ell \mid q = p - \epsilon) \leq mp$. We define $a = m(p-\epsilon) - 10\sqrt{mp}$ and $b = mp + 10\sqrt{mp}$, and using Chebyshev inequality we have $P(\ell \in [a, b] \mid q = p) \geq 99/100$ and $P(\ell \in [a, b] \mid q = p - \epsilon) \geq 99/100$. Hence, $P(\ell \in [a, b]) \geq 99/100$. Notice that $|b - a| = m\epsilon + 20\sqrt{mp} = o(p/\epsilon)$ where the second equality holds by the assumption that $m = o(p/\epsilon^2)$. Now, we bound $\mathcal{R}(a)$ and $\mathcal{R}(b)$. Again, we notice that $\ell \mapsto \mathcal{R}(\ell)$ is an increasing function.

\[
\mathcal{R}(a) \cdot \frac{99}{100} \leq \mathcal{R}(a) \sum_{k \in [a, b]\cap\mathbb{Z}} P(\ell = k \mid q = p - \epsilon) 
\leq \sum_{k \in [a, b]\cap\mathbb{Z}} \mathcal{R}(k) P(\ell = k \mid q = p - \epsilon) 
\leq \sum_{k \in [a, b]\cap\mathbb{Z}} P(\ell = k \mid q = p) \leq 1
\]
and thus, $\mathcal{R}(a) \leq 100/99$. Analogously, we prove $\mathcal{R}(b) \geq 99/100$. For each $k < m$ we have
\[
\frac{\mathcal{R}(k + 1)}{\mathcal{R}(k)} = \frac{1 + \frac{\epsilon}{1-p}}{1 - \frac{\epsilon}{p}} \leq 1 + \frac{3\epsilon}{p}
\]
for \( p \) and \( \epsilon \) sufficiently small. Hence,

\[
R(b) \leq R(a) \cdot \left(1 + 3 \frac{p}{\epsilon}\right)^{\lceil|b-a|\rceil} \\
\leq \frac{100}{99} \cdot e^{3\frac{p}{\epsilon}|b-a|} \\
\leq \frac{100}{99} \cdot e^{o(1)} \leq \frac{100}{98}
\]

where the last inequality holds for \( p \) and \( \epsilon \) sufficiently small. Analogously, we prove \( R(a) \geq 98/100 \).

Finally, we have

\[
P\left(\frac{R(\ell)}{n} \in \left[\frac{98}{100}, \frac{100}{98}\right]\right) \leq P(\ell \not\in [a, b]) \leq \frac{1}{100}.
\]

\[\square\]

4.1 Proportional Sampling

In this section, we assume, as we did in Section 2, that we can sample only proportionally. We prove that \( \Omega(\sqrt{n}/\epsilon) \) samples are necessary to estimate \( W \) with probability 2/3, thus PropEstimator is optimal up to a constant factor.

Deciding the number of samples at run-time. In all our lower bounds, we show that it is not possible that an algorithm takes \( m = o(T(n, \epsilon)) \) samples in the worst case and correctly approximates \( W \) with probability 2/3. All our proofs work by showing that the Bayes classifier has risk \( 1/2 - o(1) \).

Suppose now that we have an algorithm \( A \) that uses in expectation \( \mu(n, \epsilon) = o(T(n, \epsilon)) \) samples. We now define a classifier as follows. We run \( A \) and abort if it uses more than \( 20 \mu(n, \epsilon) \) samples.\(^9\) We return the answer given by \( A \) or an arbitrary value if we have aborted the algorithm. By the Markov’s inequality, the probability that we abort is at most \( 1/20 \). Our classifier has risk \( 1/3 + 1/20 < 2/5 \). Since any constant success probability greater than \( 1/2 \) is equivalent up to probability amplification, we also have a classifier with risk \( 1/3 \) that uses \( O(\mu(n, \epsilon)) = o(T(n, \epsilon)) \) samples. Since a Bayes classifier with such parameters does not exist (as we show) and Bayes classifiers are risk-optimal, this is a contradiction.

**Theorem 4.5.** In the proportional setting, there does not exist an algorithm that, for every instance \((U, w)\) with \(|U| = n\), takes \( m = o(\sqrt{n}/\epsilon) \) samples and returns a \((1 \pm \epsilon)\)-approximation of \( W \) with probability 2/3. This holds also when \( n \) is known to the algorithm.

**Proof.** As already proven, we may assume that the algorithm only gets the fingerprint \( F_m \) of the sample \( S \) of size \( m \), instead of \( S \) itself. In the rest of the proof, we separately consider two cases: \( \epsilon \geq 1/\sqrt{n} \) and \( \epsilon < 1/\sqrt{n} \).

**Case \( \epsilon \geq 1/\sqrt{n} \):** We first define the hard instance \((U, w)\). Define the random variable \( k \) as \( k = (1-\epsilon)n \) with probability 1/2 and \( k = n \) otherwise, then let \( U = \{a_1 \ldots a_n\} \) and

\[
w(a_i) = \begin{cases} 1 & \text{if } i \leq k \\ 0 & \text{otherwise.} \end{cases}
\]

Items with weight zero are never sampled while sampling proportionally while we are sampling uniformly from those with weight 1. Moreover \( m = o(\sqrt{n}/\epsilon) = o(n) \) for \( \epsilon \geq 1/\sqrt{n} \). Hence, this

\(^9\)Note that, while the algorithm does not know \( \mu(n, \epsilon) \), this is not an issue in this argument. The reason is that a classifier is defined as an arbitrary function from the set of possible samples and private randomness to the set of classes. This allows us to “embed” \( \mu \) into the classifier.
is exactly the settings of Lemma 4.3. If we let $F_m$ to be the fingerprint of the samples and define $\mathcal{R}(F_m)$ as in Lemma 4.3, we have

$$P\left(\frac{98}{100} \leq \mathcal{R}(F_m) \leq \frac{100}{98}\right) \geq \frac{99}{100}.$$ 

Applying Lemma 4.1 gives us the desired result.

**Case $\epsilon < \frac{1}{\sqrt{n}}$:** For convenience, we show an instance of size $n + 1$ instead of $n$. First, we define $s^2 = \min\left\{ \frac{\sqrt{n}}{\epsilon m}, \frac{n}{4}\right\}$ and notice that $s = o(1)$ and $s \leq \sqrt{n}/2$. Define the random variable $k$ as $k = n - s\sqrt{n}$ with probability $1/2$ and $k = n$ otherwise. Define the events $E_1 = \{k = n - s\sqrt{n}\}$ and $E_2 = \{k = n\}$. We construct $(U, w)$ so that $U = \{a_0 \ldots a_n\}$ and

$$w(a_i) = \begin{cases} \frac{s\sqrt{n}}{\epsilon} - n & \text{if } i = 0 \\ 1 & \text{if } 1 \leq i \leq k \\ 0 & \text{otherwise.} \end{cases}$$

Notice that our choice of $s$ together with $\epsilon < 1/\sqrt{n}$ guarantees $n - s\sqrt{n} \geq n/2$ and $s\sqrt{n}/\epsilon - n = o(n)$. We have that $W = \sum_{i=0}^{n} w(a_i)$ differs by more than a factor of $(1 + \epsilon)$ between events $E_1$ and $E_2$.

Consider an element $a \in U$ sampled proportionally and define $p_i = P(a \neq a_0 | E_i)$ for $i = 1, 2$. Then,

$$p_2 = \frac{n}{n - s\sqrt{n} + w(a_0)} = \frac{\epsilon\sqrt{n}}{s},$$

$$p_1 = \frac{n - s\sqrt{n}}{n - s\sqrt{n} + w(a_0)} = \frac{p_2 - \epsilon}{1 - \epsilon}.$$

and $|p_1 - p_2| \leq \epsilon$. We perform $m$ samples in total and define the random variable $X$ counting the number of times items other than $a_0$ are sampled. We define $F_X$ as the fingerprint of the sampled items different from $a_0$. Given $F_X$, we may easily reconstruct the fingerprint of the whole sample by adding the tuple $(m - X, w(a_0))$. It thus holds $P(F_m | E_i) = P(F_X | E_i)$ for $i \in \{1, 2\}$.

Now we define the event $\mathcal{L} = \{X \leq 30E[X]\}$ and by Markov’s inequality $P(\mathcal{L}) \geq 29/30$. Moreover,

$$E[X] = m(p_1 + p_2)/2 \leq m(p_2 + \epsilon) = m(\epsilon\sqrt{n}/s + \epsilon) \leq 2m\epsilon\sqrt{n}/s = o\left(\frac{n}{s}\right).$$

Define $\epsilon' = s/\sqrt{n}$. Conditioning on $\mathcal{L}$, we have $X \leq 30E[X] = o(\sqrt{n}/\epsilon') = o(n)$. If we further condition on $X = x$ for some $x \leq 30E[X]$ and we look at $F_X$, we are exactly in the setting of Lemma 4.3. Therefore,

$$P\left(\frac{98}{100} \leq \frac{P(F_X | X = x, E_1)}{P(F_X | X = x, E_2)} \leq \frac{100}{98}\right) \geq \frac{99}{100}$$

and integrating over $\mathcal{L}$ we obtain

$$P\left(\frac{98}{100} \leq \frac{P(F_X | X, E_1)}{P(F_X | X, E_2)} \leq \frac{100}{98} \bigg| \mathcal{L}\right) \geq \frac{99}{100}. $$

Now, we will bound the ratio

$$\mathcal{R}(X) = \frac{P(X | E_1)}{P(X | E_2)}.$$ 

We have $X = \sum_{i=1}^{m} X_i$, where $X_i$ is an indicator for the $i$th sample not being equal to $a_0$. Therefore, $X | E_j \sim \text{Bin}(m, p_j)$ for $j = 1, 2$. It holds, $|p_1 - p_2| \leq \epsilon$. Because $m = o(\sqrt{n}/\epsilon)$ and by the definition
of $s$, we have $m = o(\sqrt{n}/(\varepsilon)) = o(p_2/\varepsilon^2)$. We can apply Lemma 4.4 and obtain

$$P\left(\frac{98}{100} \leq \mathcal{R}(X) \leq \frac{100}{98}\right) \geq \frac{99}{100}.$$  

Finally, we put the bounds together. We consider the ratio

$$\mathcal{R}(F_m) = \frac{P(F_m \mid E_1)}{P(F_m \mid E_2)} = \frac{P(F_X \mid E_1)}{P(F_X \mid E_2)} = \frac{P(X \mid E_1) \cdot P(F_X \mid X, E_1)}{P(X \mid E_2) \cdot P(F_X \mid X, E_2)}.$$  

By the union bound, along with $7/8 < (98/100)^2$, we get

$$P\left(\mathcal{R}(F_m) \notin \left[\frac{7}{8}, \frac{8}{7}\right]\right) \leq P\left(\mathcal{L}\right) + P\left(\frac{P(F_X \mid X, E_1)}{P(F_X \mid X, E_2)} \notin \left[\frac{98}{100}, \frac{100}{98}\right] \mid \mathcal{L}\right) \leq \frac{1}{30} + \frac{1}{100} + \frac{1}{100} \leq \frac{1}{15}.$$  

We can apply Lemma 4.1 and conclude the proof. \hfill \square

### 4.2 Sum Estimation in Hybrid Setting, Known $n$

In this section, we assume, as we did in Section 3, that we can sample both proportionally and uniformly. We will prove that $\Omega(\min(\sqrt{n}/\varepsilon^{4/3}, n))$ samples are necessary to estimate $W$ with probability $2/3$. This complexity is the minimum of the sample complexity of the HybridEstimator and (up to a logarithmic factor) the complexity of the standard coupon collector algorithm. Recall that for algorithms in the hybrid setting, we can assume that they take as input two separate fingerprints: the fingerprint of the proportional and the fingerprint of the uniform samples.

**Theorem 4.6.** In the hybrid setting, there does not exist an algorithm that, for every instance $(U, w)$ with $|U| = n$, takes $m = o(\min(\sqrt{n}/\varepsilon^{4/3}, n))$ proportional and uniform samples and returns a $(1 \pm \varepsilon)$-approximation of $W$ with probability $2/3$. This holds also when $n$ is known to the algorithm.

The proof of Theorem 4.6 closely mimics the proof of Lemma 4.3. Indeed, in order to “fool” the uniform samples, we pad our distribution with many 0-weight elements. Indeed, only $n^{2/3}/\varepsilon^{2/3}$ element will have non-zero weight. On those elements, we basically apply Lemma 4.3. Then, to bound the information obtained by sampling (uniformly) zero-weight elements we use Lemma 4.4.

**Proof.** It is enough to prove that for $\varepsilon \geq 8/\sqrt{n}$, any algorithm returning a $(1 \pm \varepsilon)$-approximation of $W$ with probability $2/3$ must take $\Omega(\sqrt{n}/\varepsilon^{4/3})$ samples. Indeed, if $\varepsilon < 8/\sqrt{n}$, a $(1 \pm \varepsilon)$-approximation is also a $(1 \pm 8/\sqrt{n})$-approximation, and then we have reduced to the case $\varepsilon \geq 8/\sqrt{n}$ where we know that $\Omega(\sqrt{n}/\varepsilon^{4/3}) = \Omega(n)$ samples are necessary. In either case we then need $\Omega(\min(\sqrt{n}/\varepsilon^{4/3}, n))$ samples.

Define the random variable $k$ as $k = (1 - \varepsilon)n^{2/3}/\varepsilon^{2/3}$ with probability $1/2$ and $k = n^{2/3}/\varepsilon^{2/3}$, otherwise. Define the events $E_1 = \{k = (1 - \varepsilon)n^{2/3}/\varepsilon^{2/3}\}$ and $E_2 = \{k = n^{2/3}/\varepsilon^{2/3}\}$. We construct $(U, w)$ so that $U = \{a_1 \ldots a_n\}$ and

$$w(a_i) = \begin{cases} 1 & \text{if } 1 \leq i \leq k \\ 0 & \text{otherwise.} \end{cases}$$

We have that $W = \sum_{i=1}^{n} w(a_i)$ differs by more than a factor of $(1 + \varepsilon)$ between events $E_1$ and $E_2$. Notice that $k \leq n/4$ as $\varepsilon \geq 8/\sqrt{n}$. Let $S_p$ be the multiset of proportional samples and $S_u$ the multiset of uniform samples. Let $T_1 = (S_p \cup S_u) \cap \{k + 1, \ldots, n\}$ and $T_2 = (S_p \cup S_u) \cap \{1, \ldots, k\}$. Let $F_1 = \{(c_i, w(a_i))\}_i$ be the fingerprint of $T_1$ and $F_2 = \{(c_i, w(a_i))\}_i$ be the fingerprint of $T_2$. We now argue that $(F_1, F_2)$ is sufficient for any algorithm, oblivious of the choice of $k$, to reconstruct
sample multisets \( S'_u \) and \( S'_p \) distributed as \( S_u \) and \( S_p \). We pick \(|F_i| - m\) items at random from \( F_i \) and let \( S'_u \) be the multiset of these items, together with all items in \( F_h \). We let \( S'_p \) be the multiset of the items left in \( F_i \). It is easy to verify that \( (S'_u, S'_p) \sim (S_u, S_p) \). Thus, we can assume that the algorithm is given \((F_i, F_h)\) as input, instead of the sample multisets \( S_u \) and \( S_p \).

Consider \( a \in U \) sampled uniformly, and define \( p_i = P(w(a) = 1 \mid E_i) \) for \( i = 1, 2 \). Then,

\[
p_1 = \frac{n^{2/3}/\varepsilon^{1/3}}{n} = \frac{1}{n^{1/3}\varepsilon^{2/3}},
\]

\[
p_2 = \frac{(1 - \varepsilon)n^{2/3}/\varepsilon^{1/3}}{n} = \frac{1}{n^{1/3}\varepsilon^{2/3}} - \frac{\varepsilon^{1/3}}{n^{1/3}}
\]

and defining \( \varepsilon' = \varepsilon^{1/3}/n^{1/3} \) we obtain \( p_2 = p_1 - \varepsilon' \). Let \( X = |S_u \cap \{1, \ldots, k\}| \). Since \( X \mid E_j \sim \text{Bin}(m, p_j) \) for \( j = 1, 2 \) and \( m = o(n^{1/3}/\varepsilon^{1/3}) = o(p_1/\varepsilon^2) \), we can apply Lemma 4.4 and obtain

\[
P \left( \frac{98}{100} \leq \frac{P(X \mid E_1)}{P(X \mid E_2)} \leq \frac{100}{98} \right) \geq \frac{99}{100}.
\]

Conditioning on \( X = x \) for some \( x = 1 \ldots m \), \( F_i \) represents a sample of \( x + |S_p| \) items drawn uniformly from a set of cardinality \( k \), so we are in the setting of Lemma 4.3. Moreover, we have

\[
|F_i| \leq |S_p| + |S_u| = o\left(\frac{\sqrt{n}}{\varepsilon^{4/3}}\right) = o\left(\frac{n^{2/3}/\varepsilon^{2/3}}{\varepsilon}ight).
\]

Hence, Lemma 4.3 holds and we have

\[
P \left( \frac{98}{100} \leq \frac{P(F_i \mid X = x, E_1)}{P(F_i \mid X = x, E_2)} \leq \frac{100}{98} \right) \geq \frac{99}{100}
\]

and integrating over \( x = 1 \ldots m \) we have

\[
P \left( \frac{98}{100} \leq \frac{P(F_i \mid X, E_1)}{P(F_i \mid X, E_2)} \leq \frac{100}{98} \right) \geq \frac{99}{100}.
\]

Similarly, we have that \( |F_h| \leq |S_u| = o(\sqrt{n}/\varepsilon^{4/3}) = o(\sqrt{n}/\varepsilon) \) where the second inequality holds because we are assuming \( \varepsilon > 8/\sqrt{n} \). Moreover, conditioning on \( X = x \) for some \( x = 1 \ldots m \), \( F_h \) represent a sample of \( |S_u| - x \) items drawn uniformly from a set of size \( n - k \). It holds \( n - k \geq 3n/4 \), and \( n - k \) thus differs by at most a factor \( 1 - \varepsilon \) between the two events \( E_1, E_2 \). Again, we are in the right setting to apply Lemma 4.3, and integrating over \( x = 1 \ldots m \) gives

\[
P \left( \frac{98}{100} \leq \frac{P(F_h \mid X, E_1)}{P(F_h \mid X, E_2)} \leq \frac{100}{98} \right) \geq \frac{99}{100}.
\]

We are now ready to put everything together. Note that \( F_i \) and \( F_h \) are independent once conditioned on \((E_1, X)\) or \((E_2, X)\). Define the ratio

\[
\mathcal{R}(F_i, F_h) = \frac{P((F_i, F_h) \mid E_1)}{P((F_i, F_h) \mid E_2)} = \frac{P(X \mid E_1) \cdot P(F_i \mid X, E_1) \cdot P(F_h \mid X, E_1)}{P(X \mid E_2) \cdot P(F_i \mid X, E_2) \cdot P(F_h \mid X, E_2)}.
\]

Using the union bound, along with \( 7/8 < (98/100)^2 \), we get

\[
P \left( \mathcal{R}(F_i, F_h) \notin \left[ \frac{7}{8}, \frac{8}{7} \right] \right) \leq P \left( \frac{P(X \mid E_1)}{P(X \mid E_2)} \notin \left[ \frac{98}{100}, \frac{100}{98} \right] \right) + P \left( \frac{P(F_i \mid X, E_1)}{P(F_i \mid X, E_2)} \notin \left[ \frac{98}{100}, \frac{100}{98} \right] \right) + P \left( \frac{P(F_h \mid X, E_1)}{P(F_h \mid X, E_2)} \notin \left[ \frac{98}{100}, \frac{100}{98} \right] \right) \leq \frac{1}{100} + \frac{1}{100} + \frac{1}{100} < \frac{1}{15}.
\]

We can apply Lemma 4.1 and conclude the proof.

\[\square\]
4.3 Sum Estimation in Hybrid Setting, Unknown $n$

We now prove a lower bound for the hybrid setting, in case the algorithm does not know $n$.

**Theorem 4.7.** In the hybrid setting, there does not exist an algorithm that, for every instance $(U, \omega)$, takes $m = o(\min(\sqrt{n}/\varepsilon, n))$ samples and returns a $(1 \pm \varepsilon)$-approximation of $W$ with probability $2/3$. This holds also when the algorithm is provided with an advice $\tilde{n}$ such that $(1 - \varepsilon)n \leq \tilde{n} \leq n$.

**Proof.** Employing the same argument as in Theorem 4.6, it is sufficient to prove that for $\varepsilon \geq 1/\sqrt{n}$ a lower bound of $\Omega(\sqrt{n}/\varepsilon)$ holds.

Consider the instance $(U, \omega)$ where $\omega(a) = 1$ for each $a \in U$ and we set $|U| = n$ with probability $1/2$ and $|U| = (1 - \varepsilon)n$, otherwise. Providing a $(1 \pm \varepsilon)$-approximation of $W$ is equivalent to distinguishing between the two cases. On this instance, sampling uniformly and proportionally is the same. Therefore, we are in the setting of Lemma 4.3. Combining Lemmas 4.3 and 4.1 like in the proofs above, we get that no classifier can distinguish between $|U| = n$ and $|U| = (1 - \varepsilon)n$ with probability $2/3$ using $o(\sqrt{n}/\varepsilon)$ samples. \hfill \square

5 COUNTING EDGES IN A GRAPH

In this section, we show an algorithm that estimates the average degree of a graph $G = (V, E)$ in the model in which we are allowed to perform random vertex queries, random edge queries, and degree queries. Recall that a random vertex query returns a uniform sample from $V$, a random edge queries returns a uniform sample from $E$, and a degree queries returns $\deg(v)$ when we provide $v \in V$ as argument. In this section, we denote the number of vertices and edges with $n$ and $m$ respectively.

Here, we show that HARMONICESTIMATOR from Section 3, can be adapted to estimate the average degree $d$. It achieves a complexity of $O(m \log \log n / n' \varepsilon^2 + n / n' \varepsilon^2)$ in expectation, where $n'$ is the number of non-isolated vertices. This is very efficient when there are few isolated vertices and the graph is sparse. Moreover, the only way we use sampling of vertices is to estimate the number of non-isolated vertices. Therefore, if we assume that there are no isolated vertices in the graph, it is sufficient to only be able to uniformly sample edges.

Our approach is similar to that of [12] but the authors in the article do not prove bounds on the time complexity. Moreover, their algorithm only works when there are no isolated vertices.

**Theorem 5.1.** Given a graph $G = (V, E)$, consider a model that allows (1) random vertex queries, (2) random edge queries, and (3) degree queries. In this model, there exists an algorithm that, with probability at least $2/3$, returns a $(1 \pm \varepsilon)$-approximation of $d$ of the average degree $d = 2m/n$. This algorithm performs $O(m \log \log d / n' \varepsilon^2 + n / n' \varepsilon^2)$ queries in expectation, where $n'$ is the number of non-isolated vertices.

**Proof.** We first show an algorithm that is given $\hat{d}$ such that $d \leq \hat{d}$, has time complexity $O(\hat{d} n / \varepsilon^2 n')$, and is correct with probability $2/3$. We define a sum estimation problem $(U, \omega)$. We set the universe to be $U = V$ and for each vertex $v \in U$, we set its weight $\omega(v) = \deg(v)$. Then $W = \sum_{a \in U} \omega(a) = 2m$ and $W/n = d$. Sampling an edge uniformly at random and picking one of its endpoints at random corresponds to sampling a vertex proportionally to its weight. Moreover, we can sample vertices uniformly. Therefore, we are able to implement both queries of the hybrid setting. We run HARMONICESTIMATOR($\varepsilon, \hat{d}, 1$). By Lemma 3.1, it returns with probability at least $2/3$ a $1 \pm \varepsilon$-approximation of $d$, and its sample complexity is $O(\hat{d} n / \varepsilon^2 n' + n / \varepsilon^2 n')$ since what is called $p$ in Lemma 3.1 is now $n'/n$.

---

10Recall that a vertex is isolated if it has degree 0.
It remains to get rid of the need for advice $\hat{\theta}$. We use the standard technique of performing a geometric search. See, for example, [10] for more details. We initialize $\hat{\theta} = 1$ and in each subsequent iteration, we double $\hat{\theta}$. Let $K$ be a large enough constant. In each iteration, we run $K \log \log \hat{\theta}$ independent copies of $\text{HARMONICESTIMATOR}(\epsilon, \hat{\theta}, 1)$ and denote with $d_1 \ldots d_{K \log \log \hat{\theta}}$ the returned estimates. We define $\hat{d}$ as the median of $d_1 \ldots d_{K \log \log \hat{\theta}}$. We say that the $d_i$ succeeds if both the following hold: (i) $d_i \geq d/20$, (ii) $\hat{\theta} < d$ or $d_i$ is a $(1 \pm \epsilon)$-approximation of $d$. Otherwise, we say that $d_i$ fails. We extend this definition to $\hat{d}$. Thanks to Lemma 3.1, $d_i$ fails with probability $\leq 1/3 + 1/20$. By a standard argument based on the Chernoff bound, for $K$ large enough, we have that $\hat{d}$ fails with probability at most $2/(\pi \log^2(2\hat{\theta}))$. Denote with $E_j$ the event that $\hat{d}$ succeeds at iteration $j$ (i.e., when $\hat{\theta} = 2^{j-1}$). Define $E = \bigcap_{j \geq 0} E_j$. Union bound gives $P(E) \geq 1 - \frac{2}{\pi} \sum_{j > 0} \frac{1}{j^2} = 2/3$. We stop our algorithm when $\hat{d} \leq \hat{\theta}/20$ and return $\hat{d}$. Conditioning on $\{\hat{d} \leq \hat{\theta}/20\} \cap E$, we have $\hat{\theta}/20 \geq \hat{d} \geq d/20$. This implies that $\hat{\theta} \geq \hat{d}$ and hence $\hat{d}$ is a $(1 \pm \epsilon)$-approximation of $d$. Since $P(E) \geq 2/3$, we have that with probability $2/3$ the returned value is a $(1 \pm \epsilon)$-approximation of $d$.

One iteration of our algorithm has time complexity $O(\frac{\hat{\theta} n \log \log \hat{\theta}}{\epsilon^3 n \epsilon^2})$. We argue that the expected complexity is dominated by the first iteration in which $\hat{\theta} \geq 40d$. The time complexity of each additional iteration (conditioned on being executed) increases by a factor $2 + o(1)$. Each additional iteration is executed only if the previous one resulted in an estimate $\hat{d} > \hat{\theta}/20 \geq 2d$. This happens only when $\hat{d}$ is not a $(1 \pm \epsilon)$-approximation of $d$, and assuming a correct advice $\hat{\theta} \geq d$ this happens outside of $E$. Therefore, we execute each additional iteration with probability $\leq 1/3$. Since the time spend in each iteration increases by a factor $2 + o(1)$ while the probability of executing the iteration decreases by a factor of 3, the expected complexity contributed by each additional iteration for $\hat{\theta} \geq 40 d$ decreases by a factor of $3 + o(1)$. Therefore, the expected complexity is dominated (up to a constant factor) by the first execution with $\hat{\theta} \geq 40d$. If $d \geq 1$, then in this iteration, we have $\hat{\theta} = \Theta(d)$. The time complexity is then $O(\frac{m \log \log d}{\epsilon^3 n \epsilon^2})$. If $d < 1$, then it holds $\theta = O(1)$ in this execution. The complexity is then $O(\frac{n}{n \epsilon^2})$. This gives total time complexity of $O(\frac{m \log \log d}{n \epsilon^2} + \frac{n}{n \epsilon^2})$. \hfill \Box

6 OPEN PROBLEMS
We believe there are many interesting open problems related to our work. We now give a non-comprehensive list of questions that we think would give more understanding of weighted sampling and its applications.

More efficient algorithm for spacial classes of inputs. Are there some large classes of inputs for which it is possible to get a more efficient algorithm? Can the problem be parameterized by some additional parameters apart from $n, \epsilon$ (e.g., empirical variance) that tend to be small in practice?

Different sampling probabilities. Are there settings where one may efficiently sample with probability depending on the value of an item but not exactly proportional? Could this be used to give a general algorithm for estimating the sum $W$? An example of such a result is [5] where the authors show an efficient algorithm for estimating the average degree of a graph when sampling vertices with probabilities proportional to $\frac{m}{n} + d(v)$.

Get a complete understanding of edge counting. The complexity of the problem of approximately counting edges in a graph is understood in terms of $n, m$ in the setting where we can only sample...
vertices uniformly at random. What is the complexity of counting edges when we allow only random edge queries? What if both random edge and random vertex queries are allowed? As we show, it may be useful to parameterize the problem by the fraction of vertices that are not isolated. What is the complexity of the problem under such parameterization?

ACKNOWLEDGMENTS

We are grateful to Rasmus Pagh for his advice. We would like to thank our supervisor, Mikkel Thorup, for helpful discussions and his support. We would like to thank Talya Eden for her advice regarding related work.

REFERENCES

[1] Maryam Aliakbarpour, Amartya Shankha Biswas, Themis Gouleakis, John Peebles, Ronitt Rubinfeld, and Anak Yodpinyanee. 2018. Sublinear-time algorithms for counting star subgraphs via edge sampling. *Algorithmica* 80, 2 (2018), 668–697. DOI: https://doi.org/10.1007/s00453-017-0287-3

[2] Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. 2019. A simple sublinear-time algorithm for counting arbitrary subgraphs via edge sampling. In *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)* (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 124), Avrim Blum (Ed.), Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany. 6:1–6:20. https://doi.org/10.4230/LIPIcs.ITCS.2019.6

[3] Clément Canonne and Ronitt Rubinfeld. 2014. Testing probability distributions underlying aggregated data. In *Proceedings of the Automata, Languages, and Programming*, Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias (Eds.), Springer, Berlin, 283–295.

[4] Clément L. Canonne. 2022. *Topics and Techniques in Distribution Testing*. Now Publishers.

[5] Anirban Dasgupta, Ravi Kumar, and Tamas Sarlos. 2014. On estimating the average degree. In *Proceedings of the 23rd International Conference on World Wide Web (WWW ’14)*, Association for Computing Machinery, New York, NY, USA, 795–806. DOI: https://doi.org/10.1145/2566486.2568019

[6] Talya Eden, Dana Ron, and C. Seshadhri. 2017. Sublinear time estimation of degree distribution moments: The degeneracy connection. *Leibniz International Proceedings in Informatics, LIPIcs*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing. DOI: https://doi.org/10.4230/LIPIcs.ICALP.2017.7

[7] Talya Eden, Dana Ron, and C. Seshadhri. 2017. Sublinear time estimation of degree distribution moments: The degeneracy connection. In *Leibniz International Proceedings in Informatics, LIPIcs*, Vol. 80. Schloss Dagstuhl- Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing. https://doi.org/10.4230/LIPIcs.ICALP.2017.7

[8] Uriel Feige. 2006. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SIAM Journal on Computing* 35 (2006), 964–984. DOI: https://doi.org/10.1137/S0097539704447304

[9] Hendrik Fichtenberger, Mingze Gao, and Pan Peng. 2020. Sampling arbitrary subgraphs exactly uniformly in sublinear time. In *Proceedings of the 47th International Colloquium on Automata, Languages, and Programming (ICALP 2020)*. Artur Czumaj, Anuj Dawar, and Emanuela Merelli (Eds.), Leibniz International Proceedings in Informatics (LIPIcs), Vol. 168, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 45:1–45:13. DOI: https://doi.org/10.4230/LIPIcs.ICALP.2020.45

[10] Oded Goldreich and Dana Ron. 2006. Approximating average parameters of graphs. In *Proceedings of the Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*. Josep Diaz, Klaus Jansen, José D. P. Rolim, and Uri Zwick (Eds.), Springer, Berlin, 363–374.

[11] D. G. Horvitz and D. J. Thompson. 1952. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association* 47, 260 (1952), 663–685. Retrieved from http://www.jstor.org/stable/2280784

[12] Liran Katzir, Edo Liberty, and Oren Somekh. 2011. Estimating sizes of social networks via biased sampling. In *Proceedings of the 20th International Conference on World Wide Web (WWW ’11)*, Association for Computing Machinery, New York, NY, USA, 597–606. DOI: https://doi.org/10.1145/1963405.1963489

[13] Richard J. Lipton, Jeffrey F. Naughton, Donovan A. Schneider, and S. Seshadri. 1993. Efficient sampling strategies for relational database operations. *Theoretical Computer Science* 116, 1 (1993), 195–226. DOI: https://doi.org/10.1016/0304-3975(93)90224-H

[14] Rajeev Motwani, Rina Panigrahy, and Ying Xu. 2007. Estimating sum by weighted sampling. In *Proceedings of the 34th International Conference on Automata, Languages and Programming (ICALP ’07)*. Springer-Verlag, Berlin, 53–64.

[15] Krzysztof Onak and Xiaorui Sun. 2018. Probability-revealing samples. In *Proceedings of the AISTATS*. ACM Trans. Algor., Vol. 20, No. 3, Article 27. Publication date: June 2024.
[16] C. Seshadhri. 2015. A simpler sublinear algorithm for approximating the triangle count. arXiv preprint arXiv:1505.01927 (2015).

[17] Amartya Shankha Biswas, Talya Eden, and Ronitt Rubinfeld. 2021. Towards a decomposition-optimal algorithm for counting and sampling arbitrary motifs in sublinear time. In Random.

[18] Jakub Tětek and Mikkel Thorup. 2022. Edge sampling and graph parameter estimation via vertex neighborhood accesses. In Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, 1116–1129.

[19] Osamu Watanabe. 2005. Sequential sampling techniques for algorithmic learning theory. Theoretical Computer Science 348, 1 (2005), 3–14. DOI: https://doi.org/10.1016/j.tcs.2005.09.003

Received 6 April 2022; revised 21 July 2023; accepted 18 February 2024