Sketching in Adversarial Environments

Ilya Mironov[*] Moni Naor[†] Gil Segev[‡]

Abstract

We formalize a realistic model for computations over massive data sets. The model, referred to as the *adversarial sketch model*, unifies the well-studied sketch and data stream models together with a cryptographic flavor that considers the execution of protocols in "hostile environments", and provides a framework for studying the complexity of many tasks involving massive data sets.

In the adversarial sketch model several parties are interested in computing a joint function in the presence of an adversary that dynamically chooses their inputs. These inputs are provided to the parties in an on-line manner, and each party incrementally updates a compressed sketch of its input. The parties are not allowed to communicate, they do not share any secret information, and any public information they share is known to the adversary in advance. Then, the parties engage in a protocol in order to evaluate the function on their current inputs using only the compressed sketches.

In this paper we settle the complexity of two fundamental problems in this model: testing whether two massive data sets are equal, and approximating the size of their symmetric difference. For these problems we construct explicit and efficient protocols that are optimal up to poly-logarithmic factors. Our main technical contribution is an explicit and deterministic encoding scheme that enjoys two seemingly conflicting properties: incrementality and high distance, which may be of independent interest.

A preliminary version of this work appeared in *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 651-660, 2008.

[*]Microsoft Research, Silicon Valley Campus, 1065 La Avenida, Mountain View, CA 94043. Email: `mironov@microsoft.com`.

[†]Incumbent of the Judith Kleeman Professorial Chair, Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: `moni.naor@weizmann.ac.il`. Research supported in part by a grant from the Israel Science Foundation. Part of the work was done at Microsoft Research.

[‡]Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot 76100, Israel. Email: `gil.segev@weizmann.ac.il`. Part of the work was done at Microsoft Research.

1 Introduction

The past two decades have witnessed striking technological breakthroughs in information collection and storage capabilities. These breakthroughs allowed the emergence of enormous collections of data, referred to as massive data sets, such as the World Wide Web, Internet traffic logs, financial transactions, census data and many more. This state of affairs introduces new and exciting challenges in analyzing massive data sets and extracting useful information.

From a computational point of view, most of the traditional computational models consider settings in which the input data is easily and efficiently accessible. This is, however, usually not the case when dealing with massive data sets. Such data sets may either be stored on highly constrained devices or may only be accessed in an on-line manner without the ability to actually store any significant fraction of the data. In recent years several computational models which are suitable for computing over massive data sets have been developed, such as sketch and lossy compression schemes [13, 24], data stream computations [3, 21, 27], and property testing [26, 39].

Motivated by the challenges posed by computational tasks involving massive data sets, and by the existing approaches for modeling such tasks, we formalize a realistic model of computation which we refer to as the *adversarial sketch model*. This model can be seen as unifying the standard sketch model and the data stream model together with a cryptographic flavor that considers the execution of protocols in "hostile environments". The model under consideration provides a framework for studying the complexity of many fundamental and realistic problems that arise in the context of massive data sets. In what follows we briefly describe the standard sketch model and the data stream model, as well as our approach for modeling computations in hostile environments in this context.

The standard sketch model. In the standard sketch model the input is distributed among several parties. Each party runs a compression procedure to obtain a compact "sketch" of its input, and these sketches are then delivered to a referee. The referee has to compute (or to approximate) the value of a pre-determined function applied to the inputs of the parties by using only the sketches and not the actual inputs. The parties are not allowed to communicate with each other, but are allowed to share a random reference string which is chosen independently of their inputs. This string can be used, for example, to choose a random hash function that will be applied by each party to obtain a compressed sketch of its input. This model fits many scenarios in which a massive data set is partitioned and stored in a distributed manner in several locations. In each location a compressed sketch of the stored data is computed, and then sent to a central processing unit that uses only the sketches and not the actual data. The main performance criterion for protocols in this model is the size of the sketches. We note that this model is essentially the public-coin variant of the simultaneous communication model introduced by Yao [43].

The data stream model. In the data stream model the input is received as a one-way stream. Once an element from the stream has been processed it is discarded and cannot be retrieved unless it is explicitly stored in memory, which is typically small relative to the size of the data stream. The data stream model captures scenarios in which computations involve either massive data sets that are stored on sequential magnetic devices (for which one-way access is the most efficient access method), or on-line data which is continuously generated and not necessarily stored. The main performance criteria for algorithms in this model is the amount of storage they consume and the amount of time required for processing each element in the stream. For a more complete description of this model, its variants and the main results we refer the reader the surveys by Muthukrishnan [35] and by Babcock et al. [5].

The adversarial factor. In the standard sketch model described above, it is assumed that the parties share a random string, which is chosen independently of the inputs held by the parties[1]. In many real-life scenarios, however, it is not at all clear that such an assumption is valid. First, since the parties are assumed not to communicate with each other, this enforces the introduction of trust in a third party to set up the random string. In many situations such trust may not be available, and if the shared string is set up in an adversarial manner there are usually no guarantees on the behavior of the protocol. That is, there may be "bad" choices of the shared string that cause the protocol to fail with very high probability. Second, even when a truly random string is available, this string may be known to an adversary as well (and in advance), and serve as a crucial tool in attacking the system. For example, an adversary may be able to set the inputs of the parties after having seen the random string. Thus, when considering computations in a setting where the inputs of the parties may be adversarially chosen, it is usually not justified to assume independence between the shared random string and the inputs of the parties[2]. For these reasons we are interested in exploring the feasibility and efficiency of computations over massive data sets in hostile environments. In such environments the honest parties do not share any secret information, and any public information they share is known to the adversary in advance who may then set the inputs of the parties. Protocols designed in such a model have significant security and robustness benefits.

Sketching in adversarial environments. We consider a model with three participating parties: two honest parties, Alice and Bob, and an adversarial party[3]. Computation in this model proceeds in two phases. In the first phase, referred to the as the *sketch phase*, the adversarial party chooses the inputs of Alice and Bob. These inputs are sets of elements taken from a large universe $\mathcal{U}$, and provided to the honest parties in an on-line manner in the form of a sequence of insert and delete operations. Once an operation from the sequence has been processed it is discarded and cannot be retrieved unless explicitly stored. This phase defines the input sets $S_A \subseteq \mathcal{U}$ and $S_B \subseteq \mathcal{U}$ of Alice and Bob, respectively. During this phase the honest parties are completely isolated in the sense that (1) they are not allowed to communicate with each other, and (2) the sequence of operations communicated to each party is hidden from the other party. In addition, we assume that the honest parties do not share any secret information, and that any public information they share is known to the adversary in advance. In the second phase, referred to as the *interaction phase*, Alice and Bob engage in a protocol in order to compute (or approximate) a pre-determined function of their input sets.

When designing protocols in the adversarial sketch model we are mainly interested in the following performance criteria: (1) the amount of storage (i.e., the size of the sketches), (2) the update time during the sketch phase (i.e., the time required for processing each of the insert and delete operations), and (3) the communication and computation complexity during the interaction phase.

The most natural question that arises in this setting is to characterize the class of functions that can be computed or approximated in this model with sublinear sketches and poly-logarithmic

[1]In the data stream model, when dealing only with insertions, several deterministic algorithms are known, most notably those based on the notion of *core-sets* (see, for example, [2, 6]).

[2]Typical examples include: (1) Plagiarism detection – two parties wish to compute some similarity measure between documents. In this case the inputs (i.e., the documents) are chosen by the assumed plagiarizer. (2) Traffic logs comparison: two internet routers wish to compare their recent traffic logs. The inputs of the routers can be influenced by any party that can send packets to the routers.

[3]For concreteness we focus in this informal discussion on the simplest case where only two honest parties are participating in the computation. We note that the model naturally generalizes to any number of honest parties.

update time, communication and computation[4]. In the standard sketch model a large class of functions was shown to be computed or approximated with highly compressed sketches whose size is only poly-logarithmic in the size of the input. Therefore, one can ask the rather general question of whether the adversarial sketch model "preserves sublinearity and efficiency". That is, informally:

Is any function, computable in the standard sketch model with highly compressed sketches and poly-logarithmic update time, also computable in the adversarial sketch model with sublinear sketches and poly-logarithmic update time, communication and computation?

1.1 Our Contributions

In this paper we study the two fundamental problems of testing whether two massive data sets are equal, and approximating the size of their symmetric difference. For these problems we provide an affirmative answer to the above question. We construct *explicit and efficient* protocols with sketches of essentially optimal sublinear size, poly-logarithmic update time during the sketch phase, and poly-logarithmic communication and computation during the interaction phase. We settle the complexity, up to logarithmic factors, of these two problems in the adversarial sketch model.

Our main technical contribution, that serves as a building block of our protocols, is an explicit and deterministic encoding scheme that enjoys two seemingly conflicting properties: incrementality and high distance. That is, the encoding guarantees that (1) for any set S and element x the encodings of the sets $S \cup \{x\}$ and $S \setminus \{x\}$ can be easily computed from the encoding of S by modifying only a small number of entries, and (2) the encodings of any two distinct sets significantly differ with respect to a carefully chosen weighted distance. In addition, the scheme enables efficient (linear time) decoding. We believe that an encoding scheme with these properties can find additional applications, and may be of independent interest. In what follows we formally state our results[5].

Equality testing. An equality testing protocol in the adversarial sketch model is parameterized by the size N of the universe of elements from which the sets are taken, and by an upper bound K on the size of the sets to be tested[6]. Our construction provides an explicit protocol, and in addition a non-constructive proof for the existence of a protocol that enjoys slightly better guarantees[7]. We prove the following theorem:

Theorem 1.1. *In the adversarial sketch model, for every N, K and $0 < \delta < 1$ there exists a protocol for testing the equality of two sets of size at most K taken from a universe of size N with the following properties:*

1. *Perfect completeness: For any two sequences of insert and delete operations communicated to the parties that lead to the same set of elements, the parties always output* `Equal`*.*

[4]Various relaxations may be interesting as well, for instance, allowing rather high communication complexity.

[5]Our protocols have the property that, during the interaction phase, the amount of computation is linear in the amount of communication. Therefore, for simplicity, we omit the computation cost and only state the communication complexity.

[6]We note that the upper bound K on the size of the sets only imposes a restriction on the size of the sets at the end of the sketch phase. During the sketch phase the parties should be able to deal with sets of arbitrary size, and nevertheless the size of the sketches refers to their maximal size during the sketch phase. A possible adversarial strategy, for example, is to insert all the N possible elements and then to delete $N - K$ of them.

[7]The poly-logarithmic gap between the non-explicit and the explicit parameters is due to a poly-logarithmic gap between the optimal and the known explicit constructions of dispersers (see, for example, [41]). Any improved explicit construction of dispersers will, in turn, improve our explicit protocols.

2. *Soundness: For any two sequences of insert and delete operations communicated to the parties that do not lead to the same set of elements, the parties output* `Not Equal` *with probability*[8] *at least $1 - \delta$.*

3. *The size of the sketches, the update time during the sketch phase, and the communication complexity during the interaction phase are described below in Table 1.*

	Non-explicit protocol	**Explicit protocol**
Size of sketches	$O\left(\sqrt{K \cdot \log N \cdot \log(1/\delta)}\right)$	$\sqrt{K \cdot \mathrm{polylog}(N) \cdot \log(1/\delta)}$
Update time	$O(\log K \cdot \log N)$	$\mathrm{polylog}(N)$
Communication	$O\left(\left(\log^2 K + \log K \cdot \log\log N\right) \cdot \log(1/\delta)\right)$	$\mathrm{polylog}(N) \cdot \log(1/\delta)$

Table 1: The non-explicit and explicit parameters of the equality testing protocol.

A rather straightforward reduction of computations in the private-coin simultaneous communication model to computations in the adversarial sketch model (see Section 2) implies that the size the sketches in our protocols is essentially optimal (the following theorem is stated for protocols with constant error).

Theorem 1.2. *Any equality testing protocol in the adversarial sketch model requires sketches of size $\Omega\left(\sqrt{K \cdot \log(N/K)}\right)$.*

Approximating the size of the symmetric difference. We construct a protocol that enables two parties to approximate the size of the symmetric difference between their two input sets determined during the sketch phase. We prove the following theorem:

Theorem 1.3. *In the adversarial sketch model, for every N, K, $0 < \delta < 1$ and constant $0 < \rho \leq 1$, there exists a protocol for approximating the size of the symmetric difference between two sets of size at most K taken from a universe of size N with the following properties:*

1. *For any two sequences of insert and delete operations communicated to the parties that lead to sets with symmetric difference of size $\Delta_{\texttt{OPT}}$, the parties output $\Delta_{\texttt{APX}}$ such that*

$$\Pr\left[\Delta_{\texttt{OPT}} \leq \Delta_{\texttt{APX}} \leq (1+\rho)\Delta_{\texttt{OPT}}\right] > 1 - \delta \ .$$

2. *Sketches of size $O\left(\sqrt{K \cdot \log N} \cdot (\log\log K + \log(1/\delta))\right)$.*

3. *Update time $O(\log K \cdot \log N)$.*

4. *Communication complexity $O\left(\left(\log^2 K + \log K \cdot \log\log N\right) \cdot (\log\log K + \log(1/\delta))\right)$.*

As with the equality testing protocol, our construction provides an explicit protocol as well. The explicit protocol guarantees that $\Pr\left[\Delta_{\texttt{OPT}} \leq \Delta_{\texttt{APX}} \leq \mathrm{polylog}(N)\Delta_{\texttt{OPT}}\right] > 1 - \delta$, and the size of sketches, update time and communication complexity match those stated in Theorem 1.3 up to poly-logarithmic factors.

[8]The probability is taken only over the internal coin tosses of the honest parties.

Dealing with multisets and real-valued vectors. For simplicity, we stated the above results assuming that the inputs of the parties are sets. Both of our protocols, however, can in fact be used when the inputs of the parties are multisets containing at most K distinct elements each, and even real-valued vectors of length N with at most K non-zero entries. In these cases, the equality protocol determines whether the inputs are equal as multisets or as real-valued vectors, and the symmetric difference protocol approximates the number of distinct elements in the symmetric difference between the multisets or the number of non-zero entries in the difference between the vectors.

For concreteness, in this paper we present our protocols for multisets, and note that they can be easily adapted for real-valued vectors given an appropriate representation. When applied to multisets, an additional parameter of the problems under consideration is an upper bound, M, on the number of appearances of any element in each multiset. This generalization, however, hardly affects the performance of our protocols. The only change is that the size of sketches, the update time, the communication and the computation increase by a multiplicative factor of $\log M$.

1.2 Related Work

Due to the recent emergence of massive data sets, extensive work has been devoted to designing sketch-based algorithms for many tasks, such as estimating various similarity and distance measures, compressed data structures, histogram maintenance, and many more. We note that it is far beyond the scope of this paper to present an exhaustive overview of this ever-growing line of work. The reader may find Bar-Yossef's Ph.D. thesis [7] (and the many references therein) and the Handbook of Massive Data Sets [1] as sources of preliminary information and reference. We focus only on the main results that are relevant to our setting.

Gibbons and Matias [24] denoted as *synopsis data structures* any data structures that are substantively smaller than their base data sets. Their goal was to design data structures that support queries for massive data sets while minimizing the number of disk accesses. They constructed such data structures for estimating frequency moments, estimating the number of distinct elements, identifying frequent elements and maintaining histograms and quantiles. Broder et al. [13, 14] developed efficient sketching techniques to determine the syntactic similarity of documents. They defined a similarity measure known as *resemblance*, which was shown to be applicative in eliminating near-duplicates of web pages. Sketch computations were found useful in particular for approximate nearest-neighbor algorithms in high-dimensional spaces by Indyk and Motwani [30] and by Kushilevitz, Ostrovsky and Rabani [33]. Both of these approaches were based on compressed sketches for estimating various distance measures, such as the Hamming distance and L^p norms. Sketches for approximating such distance measures were also developed by Feigenbaum et al. [20, 21] and by Indyk and Woodruff [31]. Additional sketch-based approximations include an edit distance approximation due to Bar-Yossef, Jayram, Krauthgamer and Kumar [8], and Cosine similarity and earth mover distance due to Charikar [16]. We note that the above mentioned results rely on public randomness which enables highly compressed sketches (usually of poly-logarithmic size), which are much smaller than the possible sketches in the adversarial model (given the lower bound stated in Theorem 1.2).

Compressed sensing. Sketch computations are of key importance in *compressed sensing* [15, 18], a rapidly developing field of research in signal processing. Algorithms for compressed sensing receive as input a signal and output a short sketch of the signal, usually via a small set of non-adaptive linear measurements, that can be used to approximate the signal (see, for example, [15, 17, 25, 29]).

Compressed sensing is most effective when the signal can be well approximated using much fewer vectors from some fixed basis than the signal's nominal dimension.

Early results on compresses sensing showed that the set of measurements can be chosen from some distribution, and the reconstruction algorithm was guaranteed to be correct for any specific signal with high probability over the choice of the set of measurements. However, for any set of measurements it may be possible to choose a specific signal on which the reconstruction algorithm fails. Recently, sets of measurements with significantly stronger uniform recovery properties have been discovered. Specifically, it is possible to choose one set of measurements which is "good" (in the above sense) for all signals. Although this coincides with the approach underlying the adversarial sketch model, two desiderata of protocols in the adversarial sketch model are incompatible with existing compressed sensing mechanisms: poly-logarithmic update time and sublinear space requirement.

1.3 Overview of Our Techniques

In this section we provide an informal overview of the main techniques underlying our protocols. We focus here on the problem of testing whether two massive data sets are equal, as it already illustrates the main ideas. We first make an attempt to point out the difficulties of designing an efficient equality testing protocol in the adversarial sketch model by demonstrating that known solutions to several relaxations do not seem to extend to the model under consideration.

Relaxation 1: The standard sketch model. In the standard sketch model there are highly efficient solutions which take advantage of the fact that the parties share a random string which is chosen independently of their inputs. A very simple protocol proceeds as follow: The shared random string is a description of a randomly chosen function h from a family of pair-wise independent functions that map sets of size K to $\{0,1\}$. The sketch of each party consists of a single bit obtained by applying h to its input set[9]. Clearly, if the sets are equal then the sketches are equal as well, and if the sets are not equal then the sketches differ with probability $1/2$ over the choice of h. Such an approach was demonstrated by Blum et al. [11] (in the context of memory checking) to efficiently support incremental updates by using ϵ-biased hash functions [36] instead of pair-wise independent hash functions.

Relaxation 2: The private-coin simultaneous communication model. The above solution heavily relies on the shared random string available to the parties. When such a string is not available, but arbitrary access to the inputs during the sketch phase is allowed, the model is equivalent to the private-coin simultaneous communication model [32, 43] (this is exactly the standard sketch model without shared randomness). In this model the complexity of the equality function is well-studied and tight bounds are known. Babai and Kimmel [4] (generalizing Newman and Szegedy [37]) proved that in any equality testing protocol in this model it holds that $s \times t = \Omega(K)$ where s and t are the amount of communication sent to the referee by the parties when comparing two K-bit strings.

In the simultaneous communication model there is an explicit protocol that matches this lower bound. For simplicity we present the protocol for comparing two K-bit strings. Alice and Bob agree ahead of time on an error-correcting code $C : \{0,1\}^K \to \{0,1\}^{O(K)}$ for which any two distinct codewords differ on at least a $(1-\epsilon)$-fraction of their entries (for some constant $0 < \epsilon < 1$). Alice

[9]More specifically, the parties agree ahead of time on a canonical representation for sets, and apply h to the representations of their sets.

and Bob encode their inputs, and view the codewords as $O(\sqrt{K}) \times O(\sqrt{K})$ matrices. Alice sends a random row to the referee and Bob send a random column. The referee compares the bits at the intersection of these row and column, and outputs `Equal` if and only if they match. Clearly, if the inputs of the parties are equal then the bits in the intersection always match, and if the inputs are not equal then bits differ with probability at least $1 - \epsilon$.

Such a solution does not seem to extend to the adversarial sketch model in which the parties are given only restricted access to their inputs. The inputs are provided in an on-line manner and once an operation from the sequence has been processed it is discarded and cannot be retrieved. Therefore it does not seem possible for the parties to encode their inputs in an incremental manner (i.e., with only a small cost for *each* update operation) and still obtain codewords that differ on a constant fraction of their entries. Overcoming the inherent difficulty of constructing an *incremental encoding scheme with high distance* is the main idea underlying our protocols.

Relaxation 3: Comparing "close" sets. The final relaxation we consider is not a relaxation of the model, but a relaxation of the problem. Suppose that we are guaranteed that the symmetric difference between the inputs sets S_A and S_B of the honest parties is rather small. In this case there is a simple protocol with highly compressed sketches, very fast updates and low communication. Denote by ℓ the maximal size of the symmetric difference between the two sets. The honest parties agree ahead of time on a mapping T from the universe $\mathcal{U}$ of elements to vectors over some domain with the following simple property: For every ℓ distinct elements $x_1, \ldots, x_\ell \in \mathcal{U}$ the vectors $T(x_1), \ldots, T(x_\ell)$ are linearly-independent. As a concrete mapping, for example, we assume that $\mathcal{U} = [N]$ and use $T : [N] \to \mathrm{GF}(Q)^\ell$ defined by $T(x) = (1, x, \ldots, x^{\ell-1})$ for some prime Q in the range $[N, 2N]$. The sketch of a set S is given by $\sum_{x \in S} T(x)$, and is clearly easily updated given an on-line sequence of insert and delete operations that determine the set S. In the interaction phase the parties compare their sketches and output `Equal` if and only if the sketches are equal. Then, for equal sets the sketches are always equal. If the sets are not equal but the size of their symmetric difference is at most ℓ then the sketches always differ. The property of this solution is that the size of the sketch and the communication complexity are proportional to the size of the assumed symmetric difference between the sets and not to the size of the sets. We use this solution as a tool in our construction[10].

Conflicting properties: incrementality vs. high distance. Our main technical contribution is a deterministic encoding scheme that enjoys two seemingly conflicting properties:

- **Incrementality:** Given an encoding of a set $S \subseteq \mathcal{U}$ and an additional element $x \in \mathcal{U}$, the encodings of the sets $S \cup \{x\}$ and $S \setminus \{x\}$ can be easily computed from the encoding of S without having to recompute the entire encoding of the new set. By "easily computed" we mean that only a small number of modifications is required (for example, poly-logarithmic in the size of the encoding of S).

- **High distance:** For any two distinct sets $S_A, S_B \subseteq \mathcal{U}$ of size at most K, the encodings of the sets significantly differ.

Clearly, if we consider the Hamming distance between codewords then such an encoding scheme does not exist: take any set $S \subseteq \mathcal{U}$ of size less than K, and an element $x \notin S$. Then, on one hand the incrementality property implies that the encodings of S and of $S \cup \{x\}$ differ only on very few

[10]Similar ideas are well-known in coding theory and also found applications in various other settings (for example, public-key traitor tracing [12], amortized communication complexity [19], signal processing [42] and many more). The set of linearly independent vectors is usually derived from the parity-check matrix of a linear error-correcting code.

entries, while on the other hand the high distance property implies that they differ on a significant fraction of the entries.

In our construction we manage to circumvent this conflict by considering a more generalized *weighted* distance, in which different entries are associated with different weights. More specifically, we map each set $S \subseteq \mathcal{U}$ to a *logarithmic number* of codewords $C(S) = \{C_1^S, \dots, C_k^S\}$, and consider the following distance measure:

$$\mathrm{dist}\left(C(S_A), C(S_B)\right) = 1 - \prod_{i=1}^{k} \left(1 - \mathrm{d_H}\left(C_i^{S_A}, C_i^{S_B}\right)\right) \quad ,$$

where $\mathrm{d_H}$ denotes the normalized Hamming distance. This approach enables us to construct an incremental encoding scheme in which the above distance between any two distinct codewords is at least $1-\epsilon$, for constant ϵ. The challenge in constructing such an encoding is to minimize the number k of codewords while enabling incremental updates. The number k of codewords corresponds in our protocols to the communication complexity during the interaction phase. When $k = 1$, the above distance is the normalized Hamming distance and in this case, as discussed above, the encoding cannot be incremental. When k is rather large, the encoding may be incremental, but this will lead to high communication complexity in our protocols. In our encoding, we show that $k \approx \log K$ suffices in order to enjoy "the best of the two worlds": updates require only a poly-logarithmic number of modifications, and the normalized distance between any two distinct codewords at least is $1 - \epsilon$.

We make a concentrated effort to provide an *explicit* encoding without relying on public randomness. The construction is based on a certain form of unbalanced bipartite graphs with random-like properties, that can be easily shown to be satisfied by random graphs. Obtaining explicit constructions, however, is much more subtle. In many cases, one can replace random bipartite graphs with explicit constructions of extractors, as was very recently done by Indyk [29] in the context of explicit constructions for compressed sensing [15, 18]. Explicit constructions of extractors, however, are still rather far from optimal. In this paper we emphasize the importance of identifying the *minimal properties* needed for the constructions, and as a result of identifying these properties we manage to base our constructions on "weaker" objects – dispersers, instead of extractors. Explicit constructions of dispersers are more practical, and are optimal up to poly-logarithmic factors.

More specifically, the encoding scheme is based on unbalanced bipartite graphs, which we refer to as *bounded-neighbor dispersers*. Very informally, we use these graphs as a deterministic way of mapping elements into "buckets" such that any set of elements of certain size does not lead to too many "overflowing buckets". In each bucket we apply a local encoding that "resolves collisions" among the elements mapped to the bucket[11]. A formal description of our encoding scheme is provided in Section 3.

The equality testing protocol. Given such an incremental encoding scheme, our protocol proceeds as follows: During the sketch phase each party incrementally updates an encoding that corresponds to the set defined by the sequence of insert and delete operations it receives from the adversary. That is, at the end of the sketch phase the sketches held by the parties are the encodings of their inputs sets. In the interaction phase, the parties compare a few entries of their sketches, and output `Equal` if and only if they all match. The size of the sketches in this protocol, however, is not sublinear in the size of the sets. In order to overcome this difficulty, we follow the approach

[11]Ideas along these lines were also used by Moran et al. [34] for designing history-independent data structures and conflict resolution algorithms.

described above in the simultaneous communication model, and have each party store and update only a small random sample of the entries of its codeword. Such a small sample (of size square-root of the size of a codeword) will still allow the parties to compare a few random entries. This results in a very efficient protocol with sketches of size $\widetilde{O}(\sqrt{K})$, poly-logarithmic update time during the sketch phase and poly-logarithmic communication and computation during the interaction phase. Moreover, the above mentioned lower bound of Babai and Kimmel [4] implies that the size of the sketches in our protocol is essentially optimal. A formal description of the protocol is provided in Section 4.

1.4 Paper Organization

The remainder of this paper is organized as follows. In Section 2 we formally describe the adversarial sketch model and point out its relation to the private-coin simultaneous communication model (which yields the lower bound of Theorem 1.2). In Section 3 we describe the encoding scheme which serves as a building block of our protocols. In Section 4 we present an equality testing protocol in the adversarial sketch model, and in Section 5 we demonstrate that a similar approach can be refined and extended to approximate the size of the symmetric difference between two sets. In Section 6 we present constructions of bounded-neighbor dispersers that are used to instantiate our protocols. Section 7 provides some concluding remarks and open problems.

2 The Adversarial Sketch Model

In this section we formally describe the adversarial sketch model and its relation to the private-coin simultaneous communication model (which yields the lower bound of Theorem 1.2). The adversarial sketch model consists of several honest parties and an adversarial party. In this paper we consider an information theoretic setting, and do not impose any restrictions on the computational capabilities of the adversary. At the same time, however, our protocols are efficient and can be executed by probabilistic polynomial-time parties. For concreteness we focus on the simplest case where there are only two honest parties, Alice and Bob, and note that the model naturally generalizes to any number of honest parties. Computation in this model proceeds in two phases:

Sketch phase. In the first phase the adversarial party chooses the inputs of Alice and Bob. These inputs are sets (or, more generally, multisets) of elements taken from a large universe $\mathcal{U}$, and provided to the honest parties in an on-line manner in the form of a sequence of insert and delete operations. Once an operation from the sequence has been processed it is discarded and cannot be retrieved unless explicitly stored. This phase defines the inputs $S_A \subseteq \mathcal{U}$ and $S_B \subseteq \mathcal{U}$ of Alice and Bob, respectively. During this phase the honest parties are completely isolated in the sense that (1) they are not allowed to communicate with each other, and (2) the sequence of operations communicated to each party is hidden from the other party. In addition, we assume that the honest parties do not share any secret information, and that any public information they share is known to the adversary in advance.

Interaction phase. In the second phase Alice and Bob engage in a (possibly interactive) protocol in order to compute (or approximate) a pre-determined function of S_A and S_B. The adversary is not active during this phase.

When designing protocols in the adversarial sketch model we are mainly interested in the following performance criteria:

1. **Size of sketches:** The amount of storage required by the honest parties during the sketch phase. We note that these sketches are the only information about the inputs S_A and S_B that is available to Alice and Bob during the interaction phase.

2. **Update time:** The time required by the honest parties for processing each of the insert and delete operations. More specifically, at any point in time during the sketch phase, each honest party holds a sketch corresponding to the sequence of insert and delete operation communicated from the adversary so far. Given an additional insert or delete operation, the sketch is usually modified, and the update time is the time required for such a modification.

3. **Communication complexity:** The number of bits communicated by the honest parties during the interaction phase.

4. **Computation complexity:** The amount of computation required by the honest parties during the interaction phase.

5. **Round complexity:** The number of communication rounds during the interaction phase. The number of communication rounds in our protocols is at most 3, and therefore we omit this parameter when stating our results[12].

6. **Accuracy:** The probability that the computation is successful. In case of an approximation protocol, the approximation guarantee is considered as well.

Lower bounds via simultaneous communication complexity. As mentioned in Section 1.3, the private-coin simultaneous communication model is a relaxation of the adversarial sketch model. Any protocol in the adversarial sketch model can be transformed into a simultaneous protocol: the parties send their sketches to the referee, who then internally simulates the interaction phase. Therefore, any communication lower bound in the private-coin simultaneous communication model serves as a lower bound on the size of the sketches in the adversarial sketch model. This simple observation is used for establishing the optimality of the sketches in our protocols (up to poly-logarithmic factors), by using the following lower bound for equality [4, 37]: In any equality testing protocol for K-bit strings in the private-coin communication complexity model it holds that $s \times t = \Omega(K)$, where s and t are the number of bits sent to the referee by Alice and Bob, respectively.

3 An Incremental Encoding Scheme

In this section we present the encoding scheme that serves as the basis of our protocols. The scheme encodes multisets, and is parameterized as follows:

- N - the size of the universe $\mathcal{U}$ of elements. For simplicity we assume that $\mathcal{U} = [N]$.
- K - the maximal number of distinct elements in each multiset.
- M - the maximal number of appearances of each element in any multiset.

[12] More specifically, our protocols consist of 3 messages, where after the second message one of the parties already learns the result. This result is sent to the other party as the third message.

3.1 The Encoding

The encoding of a multiset $S \subseteq [N]$ consists of a sequence of codewords $C(S) = \{C_0^S, \ldots, C_{k+1}^S\}$, where $k = \lceil \log_{1+\rho} K \rceil$ for some constant $\rho > 0$, with the following properties[13]:

- **Incrementality:** Given an encoding $C(S) = \{C_0^S, \ldots, C_{k+1}^S\}$ of a multiset $S \subseteq \mathcal{U}$ and an additional element $x \in \mathcal{U}$, the encodings of the multisets $S \cup \{x\}$ and $S \setminus \{x\}$ can be computed from the encoding of S by modifying only a poly-logarithmic number of entries in each codeword C_i^S.

- **High distance:** For any two distinct multisets $A, B \subseteq \mathcal{U}$, each containing at most K distinct elements and no element appears more than M times, there exists an integer $0 \leq i \leq k+1$ for which the codewords C_i^A and C_i^B differ on a $(1-\epsilon)$-fraction of their entries (note that this in particular implies distance at least $1-\epsilon$ according to the metric discussed in Section 1.3).

Informally speaking, the scheme incorporates two encodings: a *global encoding* that maps each element $x \in [N]$ to several entries of each of the $k+2$ codewords, and a *local encoding* that applies to the sets of elements mapped to each entry. More specifically, given a multiset $S \subseteq [N]$, we denote by $S_{i,y} \subseteq S$ the *set* of elements in S that are mapped to entry y of the i-th codeword, denoted $C_i^S[y]$ (note that we consider $S_{i,y}$ as a set and not as a multiset). We construct a mapping with the following property: for any two distinct multisets $A, B \subseteq [N]$ that contain at most K distinct elements each, there exists an index i for which $1 \leq |A_{i,y} \Delta B_{i,y}| \leq \ell$ for a significant fraction of the entries y and for some integer ℓ (where Δ denotes symmetric difference). The local encoding will then guarantee that $C_i^A[y] \neq C_i^B[y]$ by using the solution for "close" sets described in Section 1.3 as relaxation 3.

For each codeword C_i, we view the global encoding as a bipartite graph $G = (L, R, E)$, where the set of vertices on the left, L, is identified with the universe of elements $[N]$, and the vertices on the right, R, are identified with the entries of C_i. An element $x \in [N]$ is mapped to entry $C_i[y]$ if and only if the edge (x, y) exists. We are interested in bipartite graphs with the following property: For every set $S \subseteq [N]$ of size roughly K', at least $(1-\epsilon)$-fraction of the vertices $y \in R$ have at least one and at most ℓ neighbors from S. Now consider any two different multisets $A, B \subseteq [N]$ that contain at most K distinct elements each, and suppose that the number of distinct elements in the multiset $S = A \Delta B$ is roughly K'. Then, such a property of the bipartite graph corresponding to the i-th codeword implies that $1 \leq |A_{i,y} \Delta B_{i,y}| \leq \ell$ for at least $(1-\epsilon)$-fraction of the entries $C_i[y]$. We refer to such graphs as *bounded-neighbor dispersers*. Formally, we define:

Definition 3.1. *Let $G = (L, R, E)$ be a bipartite graph. For a set $S \subseteq L$ and an integer ℓ we denote by $\Gamma(S, \ell)$ the set of all vertices in R that have at least one and at most ℓ neighbors in S.*

Definition 3.2. *A bipartite graph $G = (L, R, E)$ is a $(K, \epsilon, \rho, \ell)$-*bounded-neighbor disperser *if for every $S \subseteq L$ such that $K \leq |S| < (1+\rho)K$, it holds that $|\Gamma(S, \ell)| \geq (1-\epsilon)|R|$.*

For such graphs we denote $|L| = N$, and in addition we assume that all the vertices on the left have the same degree D, which is called the left-degree of the graph. We discuss and provide constructions of bounded-neighbor dispersers in Section 6.

The local encoding uses the solution described in Section 1.3 that applies whenever we are guaranteed to compare multisets for which the number of distinct elements in their symmetric difference is rather small. Denote by ℓ the bound on the size of the symmetric difference provided

[13]For the equality testing protocol it is sufficient to only consider $\rho = 1$. For approximating the size of the symmetric difference it will be useful to consider the more general case.

by the global encoding. That is, for any two different multisets $A, B \subseteq [N]$ that contain at most K distinct elements each the global encoding guarantees that there exists an index i for which $1 \leq |A_{i,y} \Delta B_{i,y}| \leq \ell$ for a significant fraction of the entries y. The local encoding consists of a mapping T from the universe $\mathcal{U}$ of elements to vectors over some domain with the following property: For every ℓ distinct elements $x_1, \ldots, x_\ell \in \mathcal{U}$ the vectors $T(x_1), \ldots, T(x_\ell)$ are linearly-independent. As a concrete mapping, for example, we use $T : [N] \to \mathrm{GF}(Q)^\ell$ defined by $T(x) = (1, x, \ldots, x^{\ell-1})$ for some prime Q such that $\max\{N, M\} < Q \leq 2\max\{N, M\}$. The local encoding in each entry $C_i^S[y]$ is given by $\sum_{x \in S_{i,y}} \sharp(x, S) \cdot T(x)$, where $\sharp(x, S)$ denotes the number of appearances of x in S. This is clearly easily updated given an on-line sequence of insert and delete operations that determine the multiset S.

A formal description. Let $G_0, \ldots, G_{k+1}$ denote a sequence of bipartite graphs $G_i = (L = [N], R_i, E_i)$ with left-degree D_i. The graphs are constructed such that each G_i is a $(K_i = (1+\rho)^i, \epsilon, \rho, \ell)$-bounded-neighbor disperser for some constant $0 < \epsilon < 1$ and an integer $\ell \geq 1$. The encoding consists of a sequence of codewords $C_0, \ldots, C_{k+1}$ (initialized with all zero entries). Each codeword C_i is identified with the right side R_i of the bipartite graph G_i, and contains $|R_i|$ entries denoted by $C_i[1], \ldots, C_i[|R_i|]$. An additional tool in our construction is a mapping T from the universe $[N]$ to vectors over some domain that was described above.

Figure 1 describes the incremental update operations of the encoding. That is, given an encoding $\{C_0, \ldots, C_{k+1}\}$ of a multiset $S \subseteq [N]$ and an element $x \in [N]$, Figure 1 describes the required modifications to the codewords in order to obtain the encodings of $S \cup \{x\}$ and of $S \setminus \{x\}$. We note that the update operations naturally extend to deal with real-valued multiplicities of elements.

```
Insert($x, \{C_0, \ldots, C_{k+1}\}$):
1: for $i = 0$ to $k+1$ do
2:     for all neighbors $y$ of $x$ in the graph $G_i$ do
3:         $C_i[y] \leftarrow C_i[y] + T(x)$

Delete($x, \{C_0, \ldots, C_{k+1}\}$):
1: for $i = 0$ to $k+1$ do
2:     for all neighbors $y$ of $x$ in the graph $G_i$ do
3:         $C_i[y] \leftarrow C_i[y] - T(x)$
```

Figure 1: The insert and delete operations.

In the remainder of this section we state and prove some useful properties of the encoding scheme that will be used for analyzing our protocols. In the following lemma we show that the encoding of a multiset is independent of the sequence of operations that led to the multiset. That is, each multiset has a unique encoding.

Lemma 3.3. *Any two sequences of insert and delete operations that lead to the same multiset result in the same encoding.*

Proof. It is rather straightforward to verify that for any multiset $S \subseteq [N]$ and for any sequence of insert and delete operations that lead to S, the resulting encoding is as follows: For every codeword C_i and entry $y \in R_i$ it holds that

$$C_i[y] = \sum_{x \in S \cap \Gamma(y)} \sharp(x, S) \cdot T(x) \enspace ,$$

where $\Gamma(y) \subseteq [N]$ is the set of all neighbors of y in the graph G_i. ∎

The following lemma states that any two different multisets, each containing at most K distinct elements and no element appears more than M times, have significantly different encodings.

Lemma 3.4. *Fix any two distinct multisets $A, B \subseteq [N]$, each containing at most K distinct elements and no element appears more than M times. Denote by $C^A = \{C_0^A, \dots, C_{k+1}^A\}$ and by $C^B = \{C_0^B, \dots, C_{k+1}^B\}$ the encodings of A and B, respectively, and denote by d the number of distinct elements in $A \Delta B$. Then, for $i = \lfloor \log_{1+\rho} d \rfloor$, the codewords C_i^A and C_i^B differ on at least $(1-\epsilon)$-fraction of their entries.*

Proof. Fix $A, B \subseteq [N]$ as in the lemma, and let $C^A = \{C_0^A, \dots, C_{k+1}^A\}$ and $C^B = \{C_0^B, \dots, C_{k+1}^B\}$ denote their encodings. Denote by $S \subseteq [N]$ the set of distinct elements in the symmetric difference between A and B, and let $i = \lfloor \log_{1+\rho} |S| \rfloor$. Notice that this implies that $K_i \leq |S| < (1+\rho)K_i$. We now consider the set S as a set in the left side of the bipartite graph $G_i = ([N], R_i, E_i)$, and prove that for every $y \in \Gamma(S, \ell)$ it holds that $C_i^A[y] \neq C_i^B[y]$ (recall that $\Gamma(S, \ell)$ is the set of all elements $y \in R_i$ that have at least one and at most ℓ neighbors in S). The lemma then follows since G_i is $(K_i, \epsilon, \rho, \ell)$-bounded-neighbor disperser and therefore $|\Gamma(S, \ell)| \geq (1-\epsilon)|R_i|$.

Fix any $y \in \Gamma(S, \ell)$, and assume towards a contradiction that $C_i^B[y] = C_i^A[y]$. Denote by $a_1, \dots, a_{t_A}$ the distinct elements in the multiset $A \setminus B$ which are neighbors of y in G_i, and denote by $b_1, \dots, b_{t_B}$ the distinct elements in the multiset $B \setminus A$ which are neighbors of y in G_i. In addition, denote by $s_1, \dots, s_t$ the distinct elements in the multiset $A \cap B$ which are neighbors of y in G_i. Then,

$$C_i^A[y] = \sum_{j=1}^{t_A} \sharp(a_j, A \setminus B) \cdot T(a_j) + \sum_{j=1}^{t} \sharp(s_j, A \cap B) \cdot T(s_j)$$

$$C_i^B[y] = \sum_{j=1}^{t_B} \sharp(b_j, B \setminus A) \cdot T(b_j) + \sum_{j=1}^{t} \sharp(s_j, A \cap B) \cdot T(s_j)$$

and therefore

$$\sum_{j=1}^{t_A} \sharp(a_j, A \setminus B) \cdot T(a_j) = \sum_{j=1}^{t_B} \sharp(b_j, B \setminus A) \cdot T(b_j) \ . \tag{3.1}$$

Note that the fact $y \in \Gamma(S, \ell)$ implies by definition $1 \leq t_A + t_B \leq \ell$. The mapping T has the property that any ℓ distinct elements are mapped to linearly-independent vectors, and this contradicts Equation 3.1 (note that all the coefficients in the linear combination in Equation 3.1 are bounded by M, which is less than the characteristic of the field). Therefore $C_i^B[y] \neq C_i^A[y]$ for every $y \in \Gamma(S, \ell)$, and the lemma follows. ■

In Table 2 we describe the size of the codewords and the update time. The generic parameters are obtained directly from the parameters of the bounded-neighbor dispersers $G_0, \dots, G_{k+1}$. For simplicity, we assume in Table 2 that $M \leq N$, but note that this is not essential for our construction. The explicit and non-explicit parameters are obtained by instantiating these graphs with the non-explicit and explicit constructions from Theorem 6.1 and Corollary 6.6, respectively.

	Generic parameters	**Non-explicit**	**Explicit**
Codeword size	$\ell \log N \cdot \sum_{i=0}^{k+1} \|R_i\|$	$O(K \cdot \log^2 N)$	$K \cdot \mathrm{polylog}(N)$
Update time	$\ell \log N \cdot \sum_{i=0}^{k+1} D_i$	$O(\log K \cdot \log^2 N)$	$\mathrm{polylog}(N)$

Table 2: The codeword size and update time of the encoding scheme.

Finally, we note that in the implicit parameters (where the construction of bounded-neighbor dispersers guarantees $\ell = 1$) in the special case that $M = 1$ (i.e., we consider sets and not multisets), it is not necessary to use the mapping T described above over $\mathrm{GF}(Q)$. Instead, it is possible to only store the parity of the number of elements mapped to each entry of the codeword. This enables us to reduce the codeword size to $O(K \cdot \log(N/K))$ and the update time to $O(\log K \cdot \log N)$.

3.2 Enabling Fast Decoding

An additional property of the encoding scheme is that it can be augmented in a way that enables fast decoding, while increasing the size of the codewords and the update time by only a poly-logarithmic factor. By fast decoding we mean decoding in time $\widetilde{O}(K)$ instead of $O(N)$. Although our protocols in this paper do not take advantage of this property, an encoding scheme with the three properties of incrementality, high distance and fast decoding may be of independent interest and find additional applications beyond the setting of this paper. We note, however, that unlike the basic version of the encoding scheme, the decoding does not extend to arbitrary real-valued vectors, but only to real-valued vectors with non-negative entries.

In order to enable fast decoding we apply a more subtle local encoding. That is, each element $x \in [N]$ will be associated with a vector $T(x)$ such that the collection of vectors $\{T(x)\}_{x \in [N]}$ will have a slightly stronger property than required in the basic scheme. Recall that the encoding of a multiset $S \subseteq [N]$ is of the form $C(S) = \{C_0, \dots, C_{k+1}\}$. The global encoding maps each element $x \in [N]$ to several entries of each of the $k + 2$ codewords. Informally, we view each entry $C_i^S[y]$ as a bucket that contains the elements of S which are mapped to it by the global encoding. In each such bucket we apply a local encoding with the following two properties:

1. If S contains at most ℓ distinct elements that are mapped to the bucket, then we are able to reconstruct all of them together with their frequencies (i.e., the number of times that each of these elements appear in S).

2. We are able to identify whether S contains more than ℓ elements that are mapped to the bucket. That is, we can identify "overflowing buckets".

A encoding with these properties was explicitly constructed by Indyk [28, Section 3.1], where the length of each vector $T(x)$ is $\ell \cdot \mathrm{polylog}(N)$, and the reconstruction can be done in time $\ell \cdot \mathrm{polylog}(N)$. We note that although Indyk's construction encoded sets of elements, it can be easily extended for multisets as well. In addition, we note that the property of identifying overflowing buckets is the reason that prevents the decoding procedure to extend to arbitrary real-value vectors (see also the difference between *strong k-sets* and *weak k-sets* in [22]). We can overcome this difficulty by basing our constructions on extractors and not on dispersers, with techniques similar to those recently applied by Indyk [29]. This will result, however, with codewords of size $K \cdot 2^{O(\log^2 \log N)}$ and update time $2^{O(\log^2 \log N)}$.

In the remainder of this section we describe and analyze the decoding procedure given a local encoding with these properties.

The decoding. Given an encoding $C(S) = \{C_0, \dots, C_{k+1}\}$ of a multiset $S \subseteq [N]$ containing at most K distinct elements, the decoding proceeds as follows. We examine the codeword C_k (for the decoding process we do not need the codeword C_{k+1}), and identify all the buckets $C_k[y]$ that contain at least one and most ℓ distinct elements. If the fraction of such buckets is less than $1 - \epsilon$, then we continue to the next codeword C_{k-1}. Otherwise, we reconstruct all the elements from these buckets together with their frequencies. Denote by X the multiset of reconstructed elements. Now, we compute the encoding of the multiset $S \setminus X$ by using the delete operation from Figure 1. We repeat this process with C_k, and continue similarly until we go over all the codewords.

We now prove that the decoding scheme indeed reconstructs a multiset S from its encoding. For every $0 \leq i \leq k$ denote by S_i the multiset of unreconstructed elements when the procedure begins to process codeword C_i.

Lemma 3.5. *For every $0 \leq i \leq k$, the number of distinct elements in S_i is less than $(1+\rho)^{i+1}$.*

Proof. We prove the lemma by induction on i, starting from $i = k$. By the definition of the S_i's, we have that $S_k = S$, which has at most K distinct elements, and $K < (1+\rho)^{k+1} = (1+\rho)^{\lceil \log_{1+\rho} K \rceil + 1}$. Now assume that the number of distinct elements in S_i is less than $(1+\rho)^{i+1}$, and we prove that the number of distinct elements in S_{i-1} is less than $(1+\rho)^i$. Assume for contradiction that the number of distinct elements in S_{i-1} is at least $(1+\rho)^i$. Since $S_{i-1} \subseteq S_i$, then it is at most $(1+\rho)^{i+1}$. On one hand, the property of the graph G_i guarantees that the fraction of buckets in the codeword C_i that contain at least one and at most ℓ elements is at least $1-\epsilon$. On the other hand, the fact that we continued to C_{i-1} implies that the latter fraction is less than $1-\epsilon$, and this is a contradiction. Thus, the number of distinct elements in S_{i-1} is at most $(1+\rho)^i$. ■

The above lemma directly implies that the decoding procedure reconstructs the multiset S: By the time the procedure begins processing the final codeword C_0, at most a single element is not reconstructed, and this element is reconstructed from C_0. The following lemma settles the decoding time, when the bounded-neighbor dispersers are instantiated with the explicit construction from Corollary 6.6.

Lemma 3.6. *The decoding terminates in time $K \cdot \mathrm{polylog}(N)$.*

Proof. When processing each codeword C_i, Lemma 3.5 implies that the number of distinct unreconstructed elements is at most $(1+\rho)^{i+1}$. In each iteration there are two possible cases: In the first case, at least $(1-\epsilon)$-fraction of the buckets in C_i contain at least one and at most ℓ elements. In this case we reconstruct at least $\frac{(1-\epsilon)|R_i|}{D_i} = \frac{(1+\rho)^i}{\mathrm{polylog}(N)}$ distinct elements (recall that D_i is the left-degree of the graph G_i). Therefore, after at most $\mathrm{polylog}(N)$ such iterations, the number of distinct unreconstructed elements is at most $(1+\rho)^i$. In the second case, less than $(1-\epsilon)$-fraction of the buckets in C_i contain at least one and at most ℓ elements, and we continue to the next codeword C_{i-1}. In this case, Lemma 3.5 implies that the number of distinct unreconstructed elements is at most $(1+\rho)^i$. Therefore, the total number of iterations during the decoding process is $\mathrm{polylog}(N)$, and each such iteration can be done in time $K \cdot \mathrm{polylog}(N)$. ■

4 The Equality Testing Protocol

The incremental encoding scheme described in Section 3 serves as our main tool in designing an equality testing protocol in the adversarial sketch model. The following theorem summarizes the properties of our protocol. It is stated using the parameters of the sequence of graphs $G_0, \ldots, G_{k+1}$ used to construct the incremental encoding scheme in Section 3 (with parameter $\rho = 1$, or any other constant ρ). Theorem 1.1 follows[14] by instantiating the graphs with the bounded-neighbor dispersers from Theorem 6.1 and Corollary 6.6.

[14]We note that in the implicit parameters stated in Theorem 1.1 we are able to eliminate the $\log N$ factors from the generic parameters in case that $M = 1$,exactly as in the encoding scheme.

Theorem 4.1. *In the adversarial sketch model, for every N, K, M and $0 < \delta < 1$ there exist a protocol for testing the equality of two multisets from a universe of size N, each containing at most K distinct elements and no element appears more than M times, with the following properties:*

1. *Perfect completeness: For any two sequences of insert and delete operations communicated to the parties that lead to the same multiset of elements, the parties always output* `Equal`*.*

2. *Soundness: For any two sequences of insert and delete operations communicated to the parties that do not lead to the same multiset of elements, the parties output* `Not Equal` *with probability at least $1 - \delta$.*

3. *Sketches of size $O\left(\sqrt{\log \frac{1}{\delta}} \cdot \ell \log\left(\max\{N, M\}\right) \cdot \sum_{i=0}^{k+1} \sqrt{|R_i|}\right)$.*

4. *Worst-case update time $O\left(\ell \log\left(\max\{N, M\}\right) \cdot \sum_{i=0}^{k+1} D_i\right)$.*

5. *Communication complexity $O\left(\log \frac{1}{\delta} \cdot \left(\ell \log\left(\max\{N, M\}\right) \log K + \sum_{i=0}^{k+1} \log |R_i|\right)\right)$.*

First attempt. In the sketch phase, Alice and Bob incrementally encode the sequences of insert and delete operations they receive (using the encoding scheme from Section 3 with parameter $\rho = 1$) and obtain the encodings $C^A = \{C_0^A, \dots, C_{k+1}^A\}$ and $C^B = \{C_0^B, \dots, C_{k+1}^B\}$ of their multisets S_A and S_B, respectively. In the interaction phase, the honest parties compare a random entry from each of the codewords. More specifically, Alice picks $k+2$ uniformly distributed entries $y_0, \dots, y_{k+1}$ and sends the values $(y_0, C_0^A[y_0]), \dots, (y_{k+1}, C_{k+1}^A[y_{k+1}])$ to Bob. Bob compares these entries to the corresponding entries of his codewords, and if $C_i^B[y_i] = C_i^A[y_i]$ for every $0 \le i \le k+1$ then they output `Equal`. Otherwise, they output `Not Equal`.

Lemma 3.3 guarantees that the protocol is perfectly complete. That is, any two sequences of insert and delete operations that lead to the same multiset of elements induce the same representation, and therefore the parties will always output `Equal`. Lemma 3.4 guarantees the soundness of the protocol. That is, if Alice and Bob are given two sequences of operations that do not lead to the same multisets, then there exists an integer $0 \le i \le k+1$ for which the codewords C_i^A and C_i^B differ on at least $(1-\epsilon)$-fraction of their entries (recall that this is the distance property guaranteed by the encoding scheme). In this case the parties output `Not Equal` with probability at least $1 - \epsilon$.

The protocol. A drawback of the above protocol is that the size of the sketches (i.e., the size of the encodings C^A and C^B) is not sublinear in the size of the input sets S_A and S_B. To overcome this undesirable property, we modify the protocol as follows: The parties view each codeword C_i (corresponding to C_i^A and to C_i^B) as a square matrix. Prior to the sketch phase, Alice chooses uniformly distributed rows $a_0, \dots, a_{k+1}$ and Bob chooses uniformly distributed columns $b_0, \dots, b_{k+1}$. During the sketch phase Alice only stores and updates the entries of each codeword C_i^A along row a_i, and Bob only stores and updates the entries of each codeword C_i^B along column b_i. Notice that each party now stores only a square-root of the entries of each codeword, and this leads to sketches which are of size $\widetilde{O}(\sqrt{K})$. In the interaction phase, for every $0 \le i \le k+1$ Alice and Bob compare the entry at the intersection of row a_i and column b_i in the codewords C_i^A and C_i^B, and output `Equal` if and only if they all match.

Amplifying the success probability. The protocol described above guarantees that if Alice and Bob are given two sequences of operations that do not lead to the same sets, then they both output Not Equal with probability at least $1-\epsilon$ (where ϵ in our constructions of bounded neighbor-dispersers is some constant). More generally, given an accuracy parameter $0 < \delta < 1$, we would like to amplify the probability that they output Not Equal to $1 - \delta$. Naively, this can be achieved by having Alice and Bob compare $s = O\left(\frac{1}{1-\epsilon}\log\frac{1}{\delta}\right)$ uniformly and independently chosen entries of each codeword. This will require each party to store and update s rows or columns of each codeword, resulting in sketches of size $\widetilde{O}(\sqrt{K}\cdot\log(1/\delta))$. We now show that it is possible, however, to reduce this amount to only $\widetilde{O}\left(\sqrt{K\cdot\log(1/\delta)}\right)$. We partition each codeword C_i (corresponding to C_i^A and to C_i^B) to s disjoint parts of roughly equal size (the partition is part of the description of the protocol, and is known to the adversary in advance). Alice and Bob will compare a random entry from each such part. This will require each of them to store and update only $O(\sqrt{|C_i|/s})$ entries from each part, and therefore a total of $O(\sqrt{|C_i|s})$ entries from each codeword. This turns out to have essentially the same effect as comparing s independently chosen entries from each codeword, and results in sketches of size $\widetilde{O}\left(\sqrt{K\cdot\log(1/\delta)}\right)$. A formal proof of the properties of the protocol follows.

Proof of Theorem 4.1. The perfect completeness follows directly from the canonical representation of the encoding scheme (see Lemma 3.3). Now fix two sequences of insert and delete operations that do not lead to the same multiset of elements, and denote by S_A and S_B the resulting multisets (each containing at most K distinct elements and no element appears more than M times). We show that the parties output Not Equal with probability at least $1-\delta$. Denote by $C^A = \{C_0^A, \dots, C_{k+1}^A\}$ and by $C^B = \{C_0^B, \dots, C_{k+1}^B\}$ the encodings of S_A and S_B, respectively, and denote by d the number of distinct elements in $S_A \Delta S_B$. Then, for $i = \lfloor \log_{1+\rho} d \rfloor$, Lemma 3.4 states that the codewords C_i^A and C_i^B differ on at least $(1-\epsilon)$-fraction of their entries. These codewords are partitioned to $s = \Theta\left(\frac{1}{1-\epsilon}\log\frac{1}{\delta}\right)$ disjoint parts of roughly equal size, and from each part the parties compare a single entry. Denote by $\gamma^{(1)}, \dots, \gamma^{(s)}$ the fraction of differing entries in each part. Then, the probability that Alice and Bob output Equal is given by

$$\prod_{j=1}^{s}\left(1-\gamma^{(j)}\right) \le \exp\left(\sum_{j=1}^{s}\gamma^{(j)}\right) \le \exp\left((1-\epsilon)s\right) \le \delta \ .$$

Each codeword C_i contains $|R_i|$ entries, from which each party stores and updates $O(\sqrt{s|R_i|})$ entries, where $s = O\left(\log\frac{1}{\delta}\right)$. The content of each entry is an element of $\mathrm{GF}(Q)^\ell$ where $Q \le 2\max\{N, M\}$. Therefore the sketches are of size $O\left(\sqrt{\log\frac{1}{\delta}}\cdot\ell\log\left(\max\{N,M\}\right)\cdot\sum_{i=0}^{k+1}\sqrt{|R_i|}\right)$.

The update time is bounded by the update time of the encoding scheme (although the parties only need to update a small fraction of each codeword). The update time of the encoding scheme is at most $O\left(\ell\log\left(\max\{N,M\}\right)\cdot\sum_{i=0}^{k+1}D_i\right)$ since each insert and delete operation affects only D_i entries of each codeword C_i.

During the interaction phase, the parties compare $s = O\left(\log\frac{1}{\delta}\right)$ entries from each table. More specifically, one party sends the indices of the rows or columns that it holds for each codeword, and the other party replies with the elements in the intersection of the corresponding rows or columns. Therefore, the communication and computation during the interaction phase are bounded by $O\left(\log\frac{1}{\delta}\cdot\left(\ell\log\left(\max\{N,M\}\right)\log K + \sum_{i=0}^{k+1}\log|R_i|\right)\right)$. ■

5 Symmetric Difference Approximation

We present a protocol for approximating the size of the symmetric difference of two massive data sets in the adversarial sketch model. We note that the protocol can be applied also when the inputs of the parties are multisets, and in this case it approximates the number of distinct elements in the symmetric difference.

The approximation ratio of our protocol depends on the properties of the graphs $G_0, \ldots, G_{k+1}$ used to construct the incremental encoding scheme in Section 3, and stating the result requires introducing the following notation. Recall that each $G_i = (L = [N], R_i, E_i)$ is a $(K_i, \epsilon, \rho, \ell)$-bounded-neighbor disperser with left-degree D_i. We let $r = \max_{0 \leq i \leq k+1} \frac{K_i D_i}{(1-\epsilon)|R_i|}$. Theorem 6.1 shows that such graphs exist with $r \approx 1$, and Corollary 6.6 provides an explicit construction with $r = \mathrm{polylog}(N)$. We prove the following theorem:

Theorem 5.1. *In the adversarial sketch model, for every N, K, M, $0 < \delta < 1$ and constant $0 < \rho \leq 1$ there exist a protocol for approximating the number of distinct elements in the symmetric difference between multisets from a universe of size N, each containing at most K distinct elements and no element appears more than M times, with the following properties:*

1. *For any two sequences of insert and delete operations communicated to the parties that lead to multisets with symmetric difference that contains $\Delta_{\mathtt{OPT}}$ distinct elements, the parties output $\Delta_{\mathtt{APX}}$ such that*
$$\Pr\left[\Delta_{\mathtt{OPT}} \leq \Delta_{\mathtt{APX}} \leq (1+\rho)^3 r \Delta_{\mathtt{OPT}}\right] > 1 - \delta \ .$$

2. *Sketches of size $O\left(\left(\log\log K + \log\frac{1}{\delta}\right) \cdot \ell \log\left(\max\{N, M\}\right) \cdot \sum_{i=0}^{k+1} \sqrt{|R_i|}\right)$.*

3. *Worst-case update time $O\left(\ell \log\left(\max\{N, M\}\right) \cdot \sum_{i=0}^{k+1} D_i\right)$.*

4. *Communication $O\left(\left(\log\log K + \log\frac{1}{\delta}\right) \cdot \left(\ell \log\left(\max\{N, M\}\right) \log K + \sum_{i=0}^{k+1} \log|R_i|\right)\right)$.*

The protocol. In the sketch phase, Alice and Bob incrementally encode the sequences of insert and delete operations they receive (using the encoding scheme from Section 3 with parameter ρ), and obtain the encodings $C^A = \{C_0^A, \ldots, C_{k+1}^A\}$ and $C^B = \{C_0^B, \ldots, C_{k+1}^B\}$ of their multisets S_A and S_B, respectively. In the interaction phase, the parties compare $s = \Theta\left(\frac{1}{(1-\epsilon)\rho^2} \log \frac{\log_{1+\rho} K}{\delta}\right)$ independently chosen and uniformly distributed entries from each of the $k+2$ codewords. For every $0 \leq i \leq k+1$ denote by d_i the number of differing entries out of the s entries that the parties compare from the i-th codeword. The parties output $\Delta_{\mathtt{APX}} = (1+\rho)^{i+1}$ for the maximal i for which $d_i \geq \frac{1}{2}\left(1 + \frac{1}{1+\rho}\right)(1-\epsilon)s$. If there is no such i, then the parties output $\Delta_{\mathtt{APX}} = 0$.

In order to reduce the size of the sketches, we observe once again that the parties are not required to store and update their entire codewords. The parties view each codeword C_i (corresponding to C_i^A and to C_i^B) as a square matrix, and store and update only the entries that correspond to s random rows or columns from each codeword

In the following two lemmata we prove that the approximation guarantee is as claimed in Theorem 5.1.

Lemma 5.2. *For any two sequences of operations received by the parties, it holds that $\Delta_{\mathtt{OPT}} \leq \Delta_{\mathtt{APX}}$ with probability at least $1 - \delta/2$.*

Proof. Fix any two multisets $S_A, S_B \subseteq [N]$, each containing at most K distinct elements and no element appears more than M times. Denote by $C^A = \{C^A_0, \ldots, C^A_{k+1}\}$ and $C^B = \{C^B_0, \ldots, C^B_{k+1}\}$ their encodings, respectively. If $\Delta_{\texttt{OPT}} = 0$ (that is, $S_A = S_B$) then the encoding scheme (see Lemma 3.3) guarantees that $C^A = C^B$, and therefore the parties always output $\Delta_{\texttt{APX}} = 0$. Now suppose that $\Delta_{\texttt{OPT}} \geq 1$, and let $i_{\texttt{OPT}} = \left\lfloor \log_{1+\rho} \Delta_{\texttt{OPT}} \right\rfloor$. Lemma 3.4 states that the codewords $C^A_{i_{\texttt{OPT}}}$ and $C^B_{i_{\texttt{OPT}}}$ differ on a $(1-\epsilon)$-fraction of their entries. Alice and Bob compare $s = \Omega\left(\frac{1}{(1-\epsilon)\rho^2} \log \frac{1}{\delta}\right)$ random entries, and therefore a standard Chernoff bound implies that with probability at least $1 - \delta/2$, it holds that $d_i \geq \frac{1}{2}\left(1 + \frac{1}{1+\rho}\right)(1-\epsilon)s$. Therefore,

$$\Delta_{\texttt{APX}} \geq (1+\rho)^{i_{\texttt{OPT}}+1} = (1+\rho)^{\left\lfloor \log_{1+\rho} \Delta_{\texttt{OPT}} \right\rfloor + 1} \geq (1+\rho)^{\log_{1+\rho} \Delta_{\texttt{OPT}}} = \Delta_{\texttt{OPT}} \ .$$

■

Lemma 5.3. *For any two sequences of operations received by the parties, it holds that* $\Delta_{\texttt{APX}} \leq (1+\rho)^3 r \Delta_{\texttt{OPT}}$ *with probability at least* $1 - \delta/2$.

Proof. Fix any two multisets $S_A, S_B \subseteq [N]$, each containing at most K distinct elements and no element appears more than M times, and let $S = S_A \Delta S_B$. Denote by $C^A = \{C^A_0, \ldots, C^A_{k+1}\}$ and $C^B = \{C^B_0, \ldots, C^B_{k+1}\}$ their encodings, respectively. As in the proof of the previous lemma, if $\Delta_{\texttt{OPT}} = 0$ then the parties always output $\Delta_{\texttt{APX}} = 0$. Now suppose that $\Delta_{\texttt{OPT}} \geq 1$, and let $i_{\texttt{OPT}} = \left\lfloor \log_{1+\rho} \Delta_{\texttt{OPT}} \right\rfloor$. We argue that for sufficiently large $i > i_{\texttt{OPT}}$, the expected value of d_i is at most $\frac{1}{1+\rho} \cdot (1-\epsilon)s$. In this case a Chernoff bound guarantees that $d_i < \frac{1}{2}\left(1 + \frac{1}{1+\rho}\right)(1-\epsilon)s$ with probability at least $1 - \delta/(2\log_{1+\rho} K)$ (which suffices for a union bound over all the sufficiently large i's).

Let $i = i_{\texttt{OPT}} + j$ for some $j \geq 1$. We now consider the set $S \subseteq [N]$ as a set in the left side of the bipartite graph $G_i = ([N], R_i, E_i)$. As in the proof of Lemma 3.3, for every $y \in R_i \setminus \Gamma(S)$, it holds that $C^A_i[y] = C^B_i[y]$ (since no elements from the symmetric difference are mapped to such entries). Therefore,

$$\begin{aligned}
\mathrm{E}\left[d_i\right] &\leq \frac{|\Gamma(S)|}{|R_i|} \cdot s \\
&\leq \frac{\Delta_{\texttt{OPT}} D_i}{|R_i|} \cdot s \\
&\leq \frac{(1+\rho)^{i_{\texttt{OPT}}+1} D_i}{|R_i|} \cdot s \\
&= \frac{(1+\rho)^i D_i}{(1+\rho)^{j-1}|R_i|} \cdot s \\
&= \frac{K_i D_i}{(1+\rho)^{j-1}|R_i|} \cdot s \\
&= \frac{1}{(1+\rho)^{j-1}} \cdot \frac{K_i D_i}{(1-\epsilon)|R_i|} \cdot (1-\epsilon)s \ .
\end{aligned}$$

Therefore, for every $j \geq 2 + \log_{1+\rho} r$ it holds that

$$\begin{aligned} \mathrm{E}\left[d_{i_{\texttt{OPT}}+j}\right] &\leq \frac{1}{(1+\rho)^{j-1}} \cdot r \cdot (1-\epsilon)s \\ &\leq \frac{1}{(1+\rho)r} \cdot r \cdot (1-\epsilon)s \\ &= \frac{1}{1+\rho} \cdot (1-\epsilon)s \ . \end{aligned}$$

When setting $s = \Omega\left(\frac{1}{(1-\epsilon)\rho^2} \log \frac{\log_{1+\rho} K}{\delta}\right)$, a Chernoff bound implies that

$$\Pr\left[d_{i_{\texttt{OPT}}+j} \geq \frac{1}{2}\left(1 + \frac{1}{1+\rho}\right)(1-\epsilon)s\right] \leq \frac{\delta}{2\log_{1+\rho} K} \ ,$$

for every such j. This implies that with probability at least $1 - \delta/2$ it holds that

$$\begin{aligned} \Delta_{\texttt{APX}} &< (1+\rho)^{i_{\texttt{OPT}}+2+\log_{1+\rho} r+1} \\ &= (1+\rho)^3 r \cdot \Delta_{\texttt{OPT}} \ . \end{aligned}$$

■

In what follows we conclude the proof of Theorem 5.1, and then prove Theorem 1.3 by instantiating the protocol with the bounded-neighbor dispersers from Theorem 6.1 and Corollary 6.6.

Proof of Theorem 5.1. Lemmata 5.2 and 5.3 settle the approximation guarantee. It remains to upper bound the size of sketches, the update time during the sketch phase and the communication complexity during the interaction phase.

Each codeword C_i contains $|R_i|$ entries, from which each party stores and updates $O(s\sqrt{|R_i|})$ entries, where $s = \Theta\left(\log\log K + \log \frac{1}{\delta}\right)$. The content of each entry is an element of $\mathrm{GF}(Q)^\ell$, where $Q \leq 2\max\{N, M\}$. Therefore, the size of the sketches in the protocol is bounded by $O\left(\left(\log\log K + \log \frac{1}{\delta}\right) \cdot \ell \log\left(\max\{N, M\}\right) \cdot \sum_{i=0}^{k+1} \sqrt{|R_i|}\right)$.

The update time is bounded by the update time of the encoding scheme (although the parties only need to update a small fraction of each codeword). The update time of the encoding scheme is at most $O\left(\ell \log\left(\max\{N, M\}\right) \cdot \sum_{i=0}^{k+1} D_i\right)$ since each insert and delete operation affects only D_i entries of each codeword C_i.

During the interaction phase, the parties compare $s = \Theta\left(\log\log K + \log \frac{1}{\delta}\right)$ from each table. More specifically, one party sends the indices of the rows or columns that it holds for each codeword, and the other party replies with the elements in the intersection of the corresponding rows or columns. Therefore the communication complexity during the interaction phase is $O\left(\left(\log\log K + \log \frac{1}{\delta}\right) \cdot \left(\ell \log\left(\max\{N, M\}\right) \log K + \sum_{i=0}^{k+1} \log |R_i|\right)\right)$. ■

Proof of Theorem 1.3. Given N, K and constant $0 < \rho \leq 1$, Theorem 6.1 shows the graphs $G_0, \ldots, G_{k+1}$ can be chosen such that each $G_i = (L = [N], R_i, E_i)$ is a $(K_i = (1+\rho)^i, \epsilon, \rho, \ell)$-bounded-neighbor disperser with left-degree $D_i = O(\log(N/K_i))$, $|R_i| = O(K_i \log(N/K_i))$, $\ell = 1$, constant ϵ and $r \leq (1+\rho)^2$. In this case applying Theorem 5.1 with $\rho/31$ implies that

$$\Delta_{\texttt{OPT}} \leq \Delta_{\texttt{APX}} \leq (1+\rho/31)^5 \Delta_{\texttt{OPT}} \leq (1+\rho)\Delta_{\texttt{OPT}} \ .$$

The parameters of the bounded-neighbor dispersers together with Theorem 5.1 imply the following:

- The sketches are of size $O\left(\sqrt{K \log N} \cdot \left(\log\log K + \log \frac{1}{\delta}\right) \cdot \log\left(\max\{N, M\}\right)\right)$.
- Update time $O\left(\log K \cdot \log N \cdot \log\left(\max\{N, M\}\right)\right)$.
- Communication complexity $O\left(\left(\log\log K + \log \frac{1}{\delta}\right) \cdot \log\left(\max\{N, M\}\right) \cdot \log K\right)$.

As with the implicit parameters of the encoding scheme stated in Table 2, since the construction of bounded-neighbor dispersers guarantees $\ell = 1$, in the special case that $M = 1$ (i.e., we consider sets and not multisets) it is not necessary to use the mapping T described above over $\mathrm{GF}(Q)$. Instead, it is possible to only store the parity of the number of elements mapped to each entry of the codeword. This enables us to eliminate the $\log\left(\max\{N, M\}\right)$ factors from the above parameters.

Finally, when instantiating the protocol with the explicit construction of bounded-neighbor dispersers from Corollary 6.6, we have that $r = \mathrm{polylog}(N)$ and therefore

$$\Delta_{\mathtt{OPT}} \leq \Delta_{\mathtt{APX}} \leq \mathrm{polylog}(N)\Delta_{\mathtt{OPT}} \ .$$

In addition, the explicit construction guarantees that the size of the sketches, the update time and the communication match those of the implicit construction within poly-logarithmic factors. ■

6 Constructions of Bounded-Neighbor Dispersers

Given N, K and ρ we are interested in constructing a $(K, \epsilon, \rho, \ell)$-bounded-neighbor disperser $G = (L = [N], R, E)$, such that ϵ, ℓ and $|R|$ are minimized. We first present a non-constructive proof of the existence of a bounded-neighbor disperser with essentially optimal parameters: a constant ϵ, $\ell = 1$ and $|R| = O(K \log(N/K))$. Then, we provide an explicit construction of bounded-neighbor dispersers, by showing that any disperser [40] is also a bounded-neighbor disperser for some parameters[15].

6.1 A Non-Constructive Proof

We prove the following theorem:

Theorem 6.1. *For every N, $K \leq N$ and constant $0 < \rho \leq 1$, there exists a $(K, \epsilon = 1 - \frac{\rho}{(1+\rho)^4}, \rho, \ell = 1)$-bounded-neighbor disperser $G = (L, R, E)$, with $|L| = N$, left-degree $D = O(\log(N/K))$, and $|R| = \left\lceil \frac{(1+\rho)^2}{\rho} KD \right\rceil$.*

Lemma 6.2. *Let X denote the number of bins that contain exactly one ball, when at least n and at most $(1+\rho)n$ balls are placed independently and uniformly at random in $m = \left\lceil \frac{(1+\rho)^2 n}{\rho} \right\rceil$ bins, for some $0 < \rho \leq 1$. Then,*

$$\Pr\left[X < \frac{n}{(1+\rho)^2}\right] < \exp\left(-\frac{\rho^2}{2(1+\rho)^3} \cdot n\right) \ .$$

Proof. For every $1 \leq i \leq n$, denote by X_i the Boolean random variable that equals 1 if and only if the i-th ball is placed in a bin that does not contain any other balls. Then $X \geq \sum_{i=1}^{n} X_i$ since the

[15]A similar observation was used by Moran et al. [34] who defined the notion of *bounded-neighbor expanders*, and showed that it can be satisfied by any disperser with certain parameters. Our graphs have slightly weaker properties, and this enables more efficient constructions. That is, the parameters of dispersers are better preserved.

number of balls is at least n. At most $(1+\rho)n$ balls are placed in $m = \left\lceil \frac{(1+\rho)^2 n}{\rho} \right\rceil$ bins, therefore for every $\vec{u} \in \{0,1\}^{n-1}$ and for every $1 \le i \le n$,

$$\begin{aligned}\Pr\left[X_i = 1 \mid (X_1, \dots, X_{i-1}, X_{i+1}, \dots, X_n) = \vec{u}\right] &\ge 1 - \frac{(1+\rho)n}{m} \\ &\ge \frac{1}{1+\rho} \ .\end{aligned}$$

Let $Y_1, \dots, Y_n$ denote n independent and identically distributed Boolean random variables with the property that $\Pr[Y_1 = 1] = 1/(1+\rho)$, and let $Y = \sum_{i=1}^{m} Y_i$. A standard coupling argument shows that for every $t > 0$ it holds that $\Pr[X < t] \le \Pr[Y < t]$. Therefore, by applying a Chernoff bound for Y, we obtain

$$\begin{aligned}\Pr\left[X < \frac{n}{(1+\rho)^2}\right] &\le \Pr\left[Y < \frac{1}{1+\rho} \cdot \frac{n}{1+\rho}\right] \\ &\le \exp\left(-\frac{1}{2} \cdot \frac{n}{1+\rho} \cdot \left(1 - \frac{1}{1+\rho}\right)^2\right) \\ &= \exp\left(-\frac{\rho^2}{2(1+\rho)^3} \cdot n\right) \ .\end{aligned}$$

■

Proof of Theorem 6.1. Fix N, $K \le N$ and constant $0 < \rho \le 1$. We apply a standard probabilistic argument and show that with positive probability a random bipartite graph satisfies the required property. Let $G = (L, R, E)$ be a random bipartite graph with $|L| = N$, left-degree $D = O(\log(N/K))$ and $|R| = \left\lceil \frac{(1+\rho)^2}{\rho} KD \right\rceil$ (that is, for every vertex $x \in L$ we choose D neighbors independently and uniformly at random from R). Lemma 6.2 implies that for every set $S \subseteq L$ such that $K \le S < (1+\rho)K$, the probability over the choice of the edges of the graph that $|\Gamma(S,1)| < \frac{KD}{(1+\rho)^2}$ is at most $\exp(-cKD)$, for some constant $c > 0$. Therefore, the probability that there exists a set $S \subseteq L$ such that $K \le S < (1+\rho)K$ and for which $|\Gamma(S,1)| < \frac{KD}{(1+\rho)^2}$ is upper bounded by

$$\sum_{i=K}^{\lfloor (1+\rho)K \rfloor} \binom{N}{i} \exp\left(-cKD\right) < 1 \ ,$$

for an appropriate choice of $D = O(\log(N/K))$. In particular, there exists a choice of edges for which for every set $S \subseteq L$ such that $K \le S < (1+\rho)K$ it holds that

$$|\Gamma(S,1)| \ge \frac{KD}{(1+\rho)^2} \ge (1-\epsilon)|R| \ .$$

■

6.2 An Explicit Construction

We provide an explicit construction of bounded-neighbor dispersers by showing that any disperser is a bounded-neighbor disperser for some parameters. We again emphasize the importance of basing our protocols on dispersers and not on extractors: Whereas the existing explicit constructions of extractors are rather far from optimal, the existing explicit constructions of disperser are optimal up to poly-logarithmic factors. This yields a significant performance differences.

Dispersers [40] are combinatorial objects with many random-like properties. Dispersers can be viewed as functions that take two inputs: a string that is not uniformly distributed, but has some randomness; and a shorter string that is completely random, and output a string whose distribution is guaranteed to have a large support. Dispersers have found many applications in computer science, such as simulation with weak sources, deterministic amplification, and many more (see [38] for a comprehensive survey). We now formally define dispersers, and then show that any disperser is a bounded-neighbor disperser for some parameters.

Definition 6.3. *A bipartite graph $G = (L, R, E)$ is a (K, ϵ)-disperser if for every $S \subseteq L$ of size at least K, it holds that $|\Gamma(S)| \geq (1-\epsilon)|R|$, where $\Gamma(S)$ denotes the set of neighbors of the vertices in S.*

Lemma 6.4. *Any (K, ϵ)-disperser $G = (L, R, E)$ with left-degree D is a $(K, \epsilon', \rho = 1, \ell)$-bounded-neighbor disperser, for $\epsilon' = \frac{1+\epsilon}{2}$ and $\ell = \left\lceil \frac{4DK}{(1-\epsilon)|R|} \right\rceil$.*

Proof. Let $G = (L, R, E)$ be a (K, ϵ)-disperser with left-degree D, and define $\epsilon' = \frac{1-\epsilon}{2}$ and $\ell = \frac{4DK}{(1-\epsilon)|R|}$. We show that for every set $S \subseteq L$ such that $K \leq |S| < 2K$, it holds that $|\Gamma(S, \ell)| \geq (1-\epsilon')|R|$. First, since G is a (K, ϵ)-disperser and $|S| \geq K$, then $|\Gamma(S)| \geq (1-\epsilon)|R|$. Now consider the subgraph $G' = (S, \Gamma(S), E')$, where E' are all the outgoing edges of S. The assumption $|S| \leq 2K$ implies that there are at most $2DK$ edges in G', and therefore the average degree of the vertices of $\Gamma(S)$ in G' is at most

$$\frac{|E'|}{|\Gamma(S)|} \leq \frac{2DK}{(1-\epsilon)|R|} \leq \frac{\ell}{2} \ .$$

This implies that at least $|\Gamma(S)|/2$ vertices in $\Gamma(S)$ have degree at most ℓ in G'. Therefore,

$$|\Gamma(S, \ell)| \geq \frac{|\Gamma(S)|}{2} \geq \frac{1-\epsilon}{2} \cdot |R| = (1-\epsilon') \cdot |R| \ .$$

■

Lemma 6.4 can be instantiated, for example, with the following disperser construction of Ta-Shma, Umans and Zuckerman [41].

Theorem 6.5 ([41])**.** *For every n, k, and constant $\epsilon > 0$, there exists an efficiently computable $(K = 2^k, \epsilon)$-disperser $G = (L, R, E)$, with $|L| = N = 2^n$, $|R| = \Theta(K/\log^3(N))$ and left-degree $D = \mathrm{polylog}(N)$.*

Corollary 6.6. *For every n and k there exists an efficiently computable $(K = 2^k, \epsilon = 3/4, \rho = 1, \ell)$-bounded-neighbor disperser $G = (L, R, E)$, with $|L| = N = 2^n$, $|R| = \Theta(K/\log^3(N))$, left-degree $D = \mathrm{polylog}(N)$ and $\ell = \mathrm{polylog}(N)$.*

7 Concluding Remarks and Open Problems

Relying on computational assumptions. In this paper we considered the adversarial sketch model in the information-theoretic setting (i.e., we did not impose any restrictions on the computational capabilities of the adversary). It is reasonable to assume, however, that the adversary is polynomially bounded in any realistic setting. It will be interesting to explore whether computational assumptions can significantly improve the efficiency of protocols in the adversarial sketch model. For example, the existence of incremental collision resistant hash functions [9, 10] implies an equality testing protocol with highly compressed sketches which dramatically circumvents the lower bound stated in Theorem 1.2 in the computational setting. A major drawback of existing constructions of such hash functions is that they either rely on a random oracle, or are inefficient (more specifically, the construction of Bellare, Goldreich and Goldwasser [9] can be proved secure without a random oracle, but in this case the size of the description of each hash function is too large to be used in practice – linear in the number of input blocks)[16].

In addition, for the problem of approximating the size of the symmetric difference we are not aware of any protocol in the computational setting that similarly improves our protocol. It would be very interesting to take advantage of computational assumptions and construct such a protocol with highly compressed sketches.

Preserving sublinearity and efficiency without a shared random string. As discussed in Section 1, the most natural question that arises in the context of the adversarial sketch model is to characterize the class of functions that can be computed or approximated in this model with sublinear sketches and poly-logarithmic update time, communication and computation. In particular, we have asked whether the adversarial sketch model "preserves sublinearity and efficiency" of problems from the standard sketch model.

In this paper we provided an affirmative answer for the problems of testing whether two massive data sets are equal, and approximating the size of their symmetric difference. It would be interesting to consider other distances and similarity measures that can be efficiently approximated in the standard sketch model (See Section 1.2). For example, an intriguing measure, due to its application in eliminating near-duplicates of web pages is the resemblance measure [14], defined as

$$r(S,T) = \frac{|S \cap T|}{|S \cup T|} \ .$$

There are highly compressed sketches for estimating the resemblance between two sets using a collection of min-wise independent permutations [13]. It is not clear, however, that without shared randomness this technique can result in sketches that can be updated in an efficient incremental manner. It would be interesting to construct an efficient protocol for approximating resemblance in the adversarial sketch model.

Compressed sensing: explicit reconstruction of sparse signals. Indyk [29] showed that any signal of length N with at most K non-zero entries can be compressed (and efficiently reconstructed) using a fixed set of $K \cdot 2^{O(\log^2 \log N)}$ non-adaptive linear measurements. Indyk's construction of the set of measurements is explicit and is based on unbalanced bipartite graphs, similar to the ideas underlying our encoding scheme. However, his construction requires extractors and not dispersers, and this leads to the $2^{O(\log^2 \log N)}$ factor. It would be interesting to find explicit compressed sensing algorithms while relying on dispersers instead of extractors, and this may lead to a set of measurements of size $K \cdot \text{polylog}(N)$, which is optimal up to poly-logarithmic factors.

Acknowledgments

We thank Ziv Bar-Yossef, Robert Krauthgamer, Danny Segev and the anonymous referees for their comments.

[16]An additional construction that does not rely on random oracles can be based on the techniques of Gennaro, Halevi and Rabin [23] that require hash functions that output prime numbers, but this again does not result in an efficient construction.

References

[1] J. Abello, P. M. Pardalos, and M. G. C. Resende, editors. *Handbook of Massive Data Sets*. Kluwer Academic Publishers, 2002.

[2] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan. Geometric approximation via core sets. *Combinatorial and Computational Geometry - MSRI Publications*, pages 1–30, 2005.

[3] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Journal of Computer and System Sciences*, 58(1):137–147, 1999.

[4] L. Babai and P. G. Kimmel. Randomized simultaneous messages: Solution of a problem of Yao in communication complexity. In *Proceedings of the 12th Annual IEEE Conference on Computational Complexity*, pages 239–246, 1997.

[5] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proceedings of the 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 1–16, 2002.

[6] M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core-sets. In *Proceedings on 34th Annual ACM Symposium on Theory of Computing*, pages 250–257, 2002.

[7] Z. Bar-Yossef. *The Complexity of Massive Data Set Computations*. PhD thesis, University of California at Berkeley, 2002.

[8] Z. Bar-Yossef, T. S. Jayram, R. Krauthgamer, and R. Kumar. Approximating edit distance efficiently. In *Proceedings of the 45th Symposium on Foundations of Computer Science*, pages 550–559, 2004.

[9] M. Bellare, O. Goldreich, and S. Goldwasser. Incremental cryptography: The case of hashing and signing. In *Advances in Cryptology - CRYPTO '94*, pages 216–233, 1994.

[10] M. Bellare and D. Micciancio. A new paradigm for collision-free hashing: Incrementality at reduced cost. In *Advances in Cryptology - EUROCRYPT '97*, pages 163–192, 1997.

[11] M. Blum, W. S. Evans, P. Gemmell, S. Kannan, and M. Naor. Checking the correctness of memories. *Algorithmica*, 12(2/3):225–244, 1994.

[12] D. Boneh and M. K. Franklin. An efficient public key traitor tracing scheme. In *Advances in Cryptology - CRYPTO '99*, pages 338–353, 1999.

[13] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *Journal of Computer and System Sciences*, 60(3):630–659, 2000.

[14] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8-13):1157–1166, 1997.

[15] E. J. Candès and T. Tao. Near-optimal signal recovery from random projections: Universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406–5425, 2006.

[16] M. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 380–388, 2002.

[17] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for compressed sensing. In *Proceedings of the 13th International Colloquium on Structural Information and Communication Complexity*, pages 280–294, 2006.

[18] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[19] T. Feder, E. Kushilevitz, M. Naor, and N. Nisan. Amortized communication complexity. *SIAM Journal on Computing*, 24(4):736–750, 1995.

[20] J. Feigenbaum, Y. Ishai, T. Malkin, K. Nissim, M. J. Strauss, and R. N. Wright. Secure multiparty computation of approximations. *ACM Transactions on Algorithms*, 2(3):435–472, 2006.

[21] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan. An approximate L^1-difference algorithm for massive data streams. *SIAM Journal on Computing*, 32(1):131–151, 2002.

[22] S. Ganguly and A. Majumder. Deterministic k-set structure. In *Proceedings of the 25th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 280–289, 2006.

[23] R. Gennaro, S. Halevi, and T. Rabin. Secure hash-and-sign signatures without the random oracle. In *Advances in Cryptology - EUROCRYPT '99*, pages 123–139, 1999.

[24] P. B. Gibbons and Y. Matias. Synopsis data structures for massive data sets. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 909–910, 1999.

[25] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. One sketch for all: fast algorithms for compressed sensing. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 237–246, 2007.

[26] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45(4):653–750, 1998.

[27] M. R. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on data streams. In *External memory algorithms*, pages 107–118. American Mathematical Society, 1999.

[28] P. Indyk. Explicit constructions of selectors and related combinatorial structures, with applications. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 697–704, 2002.

[29] P. Indyk. Explicit constructions for compressed sensing of sparse signals. In *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 30–33, 2008.

[30] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the 30th Annual ACM Symposium on the Theory of Computing*, pages 604–613, 1998.

[31] P. Indyk and D. P. Woodruff. Polylogarithmic private approximations and efficient matching. In *Proceedings of the 3rd Theory of Cryptography Conference*, pages 245–264, 2006.

[32] I. Kremer, N. Nisan, and D. Ron. On randomized one-round communication complexity. *Computational Complexity*, 8(1):21–49, 1999.

[33] E. Kushilevitz, R. Ostrovsky, and Y. Rabani. Efficient search for approximate nearest neighbor in high dimensional spaces. *SIAM Journal on Computing*, 30(2):457–474, 2000.

[34] T. Moran, M. Naor, and G. Segev. Deterministic history-independent strategies for storing information on write-once memories. In *Proceedings of the 34th International Colloquium on Automata, Languages and Programming*, pages 303–315, 2007.

[35] S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.

[36] J. Naor and M. Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, 1993.

[37] I. Newman and M. Szegedy. Public vs. private coin flips in one round communication games. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing*, pages 561–570, 1996.

[38] N. Nisan and A. Ta-Shma. Extracting randomness: A survey and new constructions. *Journal of Computer and System Sciences*, 58(1):148–173, 1999.

[39] R. Rubinfeld and M. Sudan. Robust characterizations of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

[40] M. Sipser. Expanders, randomness, or time versus space. *Journal of Computer and System Sciences*, 36(3):379–383, 1988.

[41] A. Ta-Shma, C. Umans, and D. Zuckerman. Lossless condensers, unbalanced expanders, and extractors. *Combinatorica*, 27(2):213–240, 2007.

[42] H. S. Witsenhausen and A. D. Wyner. Interframe coder for video signals. U.S. patent number 4,191,970, 1980.

[43] A. C. Yao. Some complexity questions related to distributive computing. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing*, pages 209–213, 1979.