Efficient error-correcting data structures

Victor Chen * Elena Grigorescu† Ronald de Wolf‡

April 13, 2009

Abstract

We are interested in constructing efficient data structures that still work (most of the time) when hit by a constant fraction of adversarial noise. Roughly speaking, by "efficient" we mean constructions that are simultaneously close to the optimal time and space for the noiseless case. Recently, de Wolf [20] introduced a model for this, called "error-correcting data structures," and studied the tradeoff between data structure length and efficiency of query answering (as measured by the number of bit-probes). Unfortunately, this tradeoff is quite bad in that model, and it is unlikely that one could construct error-correcting data structures that are simultaneously efficient in time and space, unless significant progress is made in improving this tradeoff for "locally decodable codes." In this paper we relax the requirements on error-correcting data structures: our model only requires that *most* queries are answered correctly, while for the remaining queries the decoder is allowed to claim ignorance. If there is no noise on the data structure, it should answer all queries correctly. Using the "relaxed locally decodable codes" of Ben-Sasson et al. [5] as a building block, we show that this relaxation allows us to construct efficient, near-optimal data structures for a number of fundamental data structure problems, including versions of the MEMBERSHIP, PREDECESSOR, and RANK problems.

*MIT CSAIL, victor@csail.mit.edu. Supported by NSF award CCF-0829672

†MIT CSAIL, elena_g@mit.edu. This work started when this author was visiting CWI in Summer 2008. Supported by NSF award CCF-0829672.

‡CWI Amsterdam, rdewolf@cwi.nl. Partially supported by a Vidi grant from the Netherlands Organization for Scientific Research (NWO), and by the European Commission under the Integrated Project Qubit Applications (QAP) funded by the IST directorate as Contract Number 015848.

1 Introduction

The area of data structures is one of the oldest and most basic parts of computer science, in theory as well as in practice. The underlying question is a time-space tradeoff: we are given a piece of data, and we would like to store it in a short, space-efficient data structure that at the same time allows us to quickly answer specific queries about the stored data. On one extreme, we could store the data by just storing a list of the correct answers to all possible queries. This is extremely time-efficient (you can immediately look up the correct answer without doing any computation), but usually takes much more space than the information-theoretic minimum. At the other extreme, we could just store a maximally compressed version of the data. This is as space-efficient as it gets, but probably not very good for quickly answering queries, since we would first have to undo the whole compression. A good data structure sits somewhere in the middle: it does not use much more space than the information-theoretic minimum, but stores the data in a structured way that enables efficient query-answering.

In general, a data structure problem is specified by a set D of *data items*, a set Q of *queries*, a set A of *answers*, and a function $f : D \times Q \to A$ which specifies the correct answer $f(x, q)$ of query q to data item x. A typical example is the s-OUT-OF-n MEMBERSHIP problem. Consider a universe $[n] = \{1, \ldots, n\}$ and some $s \ll n$. Given a set $S \subseteq [n]$ of at most s elements, we would like to store it in a compact representation that can answer "membership queries" efficiently, i.e., tell us whether or not $i \in S$ for a given $i \in [n]$. Formally $D = \{S : S \subseteq [n], |S| \leq s\}$, $Q = [n]$, and $A = \{0, 1\}$. The function is $\text{MEM}_{n,s}(S, i) = 1$ if $i \in S$, and $\text{MEM}_{n,s}(S, i) = 0$ if $i \notin S$. Since there are $\binom{n}{s}$ subsets of the universe of size s, and we need a different instantiation of the data structure for each such set, clearly $\log \binom{n}{s} \approx s \log n$ bits is the information-theoretic lower bound on the space our data structure needs (our logs are always to base 2). An easy way to achieve this is to store S in sorted order. If each number is stored in its own $\log n$-bit "cell", this data structure takes s cells, which is $s \log n$ bits. To answer a membership query we can do binary search on the list, which enables us to determine whether $i \in S$ in about $\log s$ "cell-probes" or $\log s \cdot \log n$ bit-probes. The length of this data structure is essentially optimal, but its number of probes is not. Fredman, Komlós, and Szemerédi [11] developed a famous hashing-based data structure that has length $O(s)$ cells (which is $O(s \log n)$ bits), but that only needs a *constant* number of cell-probes (which is $O(\log n)$ bit-probes). Buhrman, Miltersen, Radhakrishnan, and Venkatesh [7] took the final step, designing a data structure of length $O(s \log n)$ bits that answers queries with *only one bit-probe*. This is simultaneously optimal in terms of time (clearly, 1 bit-probe cannot be improved upon), and space (up to a constant factor). Unlike the earlier data structures, theirs has a randomized decoder and a small error probability (which they show is unavoidable). Many more data structure problems have been studied, and often their optimal time-space tradeoff is known. We refer to Miltersen's survey [16] for more details.

It is reasonable to assume that most practical implementations of data storage are susceptible to *noise*: over time some of the bits in the data structure may be flipped or erased by various accidental or malicious causes. This buildup of errors may cause the data structure to deteriorate to the point that most queries are not answered correctly any more. Accordingly, it is a natural task to design data structures that are not only efficient in space and time, but also continue to work when subjected to a certain amount of noise. Some efficient data structures have indeed been designed that can cope with noise in certain special cases, for instance for pointer-based data structures [1] and for models where a small amount of incorruptible memory is available [10, 12, 9, 6].

Recently, de Wolf [20] came up with a general model of so-called error-correcting data structures, which takes its treatment of errors from the area of error-correcting codes. The goal is to design a data structure, ideally with both small length and small number of bit-probes for query-answering, that still gives correct

answers whenever the data structure is corrupted by noise. The data structure is viewed as a bitstring, and we want to be able to deal with a (small) constant fraction δ of errors.[1] As is usual in error-correcting codes, we make a worst-case assumption: the noise is not probabilistic but *adversarial*, it could be placed in positions that make life as hard as possible. Formally, the definition from [20] is as follows:

Definition 1 (ECD)**.** Let $f : D \times Q \to A$ be a data structure problem. Let p be a positive integer, $\delta \in [0,1]$, and $\varepsilon \in [0,1/2]$. A (p,δ,ε)-*error-correcting data structure (ECD)* for f of length N is a map $\mathcal{E} : D \to \{0,1\}^N$ (the "encoder") for which there is a randomized algorithm $\mathcal{D}$ (the "decoder") with the following properties: for every $x \in D$, and every $w \in \{0,1\}^N$ at Hamming distance $\Delta(w, \mathcal{E}(x)) \leq \delta N$

1. $\mathcal{D}$ makes at most p probes to its oracle (i.e., to bits of w).
2. $\Pr[\mathcal{D}^w(q) = f(x,q)] \geq 1 - \varepsilon$ for every $q \in Q$.

This model generalizes the usual noise-free data structures (where $\delta = 0$) as well as error-correcting codes (where the data structure problem has only one possible query, namely to recover the encoded string). The definition also incorporates so-called *locally decodable codes* (LDCs), which are error-correcting data structures for the membership problem with $s = n$, i.e., where the possible data pieces are all n-bit strings and a query asks for the value of the ith bit of the encoded data.

This model of error-correcting data structures is fairly clean and general but has the severe drawback that the optimal time-space tradeoffs are much worse than in the noise-free model. For instance, de Wolf [20] shows that for the membership problem where the size of the set S is bounded by some $s \ll n$, the optimal length of a p-probe error-correcting data structure roughly equals (up to a $\log n$ factor) the length of the shortest p-probe LDC that encodes s-bit strings. All known constructions of LDCs with constant number of probes have superpolynomial length [22, 8], and this has been conjectured to be inevitable.[2] Hence it is unlikely that error-correcting data structures can simultaneously be error-correcting and efficient in both time and space. Since the membership problem is a special case of many other data structure problems, those other problems are subject to the LDC lower bounds as well.

In this paper we overcome this drawback by relaxing the requirements of the data structure a bit. We take our lead from the *relaxed* locally decodable codes introduced by Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan [5]. They relax the usual definition of an LDC by requiring the decoder to return the correct answer on *most* rather than all queries i. For the remaining queries it is also allowed to claim ignorance, i.e., to output a special symbol '$\perp$' interpreted as "don't know." For none of the queries, however, is it allowed to return an incorrect answer with large probability. Formally, their definition is as follows:

Definition 2 (RLDC)**.** Let p be a positive integer, $\delta \in [0,1]$, $\varepsilon \in [0,1/2]$, and $\rho \in [0,1]$. A $(p,\delta,\varepsilon,\rho)$-*relaxed locally decodable code (RLDC)*, mapping n information bits into an encoding of length N, is a map $\mathcal{E} : \{0,1\}^n \to \{0,1\}^N$ for which there is a randomized algorithm $\mathcal{D}$ with the following properties: for every $x \in \{0,1\}^n$, and every $w \in \{0,1\}^N$ at Hamming distance $\Delta(w, \mathcal{E}(x)) \leq \delta N$

1. $\mathcal{D}$ makes at most p probes to its oracle.
2. $\Pr[\mathcal{D}^w(i) \in \{x_i, \perp\}] \geq 1 - \varepsilon$ for every $i \in [n]$.

[1] We only consider bitflip-errors here, not erasures. Since the latter are easier to deal with than bit-flips, it suffices to design a data structure dealing with bitflip-errors.

[2] However, the best *proven* lower bounds on the length of p-probe LDCs (with fixed error-rate δ and error probability ε) are only of the slightly-superlinear form $n^{1+\Omega(1/p)}$ [13, 15, 21] (only for $p = 2$ the tight bound $2^{\Theta(n)}$ is known [15]). Efficient error-correcting data structures are thus not excluded by what we know, though they seem unlikely.

3. The set $G = \{i : \Pr[\mathcal{D}^w(i) = x_i] \geq 1 - \varepsilon\}$ has size at least ρn.

4. If $w = \mathcal{E}(x)$ then $G = [n]$.

In this definition 'G' refers to the "good" set, the set of queries that are answered correctly with high probability. Note that incorrect answers have probability at most ε, no matter whether $i \in G$ or not. Also note, by Condition 4, that if there is no noise then for every possible i we get the correct answer (w.h.p.). The special case $\rho = 1$ is the usual definition of a locally decodable code (LDC), which is due to Katz and Trevisan [13]. Relaxing the LDC-definition like this suddenly allows for very efficient codes: while all known constructions of LDCs with $O(1)$ bit-probes have superpolynomial length, Ben-Sasson et al. [5] managed to construct relaxed LDCs with $O(1)$ bit-probes of *nearly-linear* length. We remark that their construction, based on PCPs of proximity, is quite involved.

We can similarly relax the definition of an error-correcting data structure to obtain the main concept of this paper: *relaxed error-correcting data structure*, or RECD for short:

Definition 3 (RECD)**.** Let $f : D \times Q \to A$ be a data structure problem. Let p be a positive integer, $\delta \in [0, 1]$, $\varepsilon \in [0, 1/2]$, and $\rho \in [0, 1]$. A $(p, \delta, \varepsilon, \rho)$*-relaxed error-correcting data structure (RECD)* for f of length N is a map $\mathcal{E} : D \to \{0, 1\}^N$ for which there is a randomized algorithm $\mathcal{D}$ with the following properties: for every $x \in D$, and every $w \in \{0, 1\}^N$ at Hamming distance $\Delta(w, \mathcal{E}(x)) \leq \delta N$

1. $\mathcal{D}$ makes at most p probes to its oracle.

2. $\Pr[\mathcal{D}^w(q) \in \{f(x, q), \perp\}] \geq 1 - \varepsilon$ for every $q \in Q$.

3. The set $G = \{q : \Pr[\mathcal{D}^w(q) = f(x, q)] \geq 1 - \varepsilon\}$ has size at least $\rho|Q|$.

4. If $w = \mathcal{E}(x)$ then $G = Q$.

Note that an ECD is an RECD with $\rho = 1$, while an RLDC is exactly an RECD for MEMBERSHIP.

The main contribution of this paper is to put forward this new definition and show that the relaxation allows us to construct error-correcting data structures that are efficient. More specifically, for a number of basic data structure problems, such as versions of MEMBERSHIP, PREDECESSOR, RANK, and NEAREST NEIGHBOR, the time and space of our RECDs are quite close to the optimal time and space tradeoff in the *noiseless* case. Accordingly, at a relatively small overhead in time and space, one can protect oneself against a constant fraction of noise, while still answering all queries correctly (with high probability) in the noiseless case. Our main theorems can be stated informally as follows.

Theorem 1 (Informal)**.** *There exists an RECD for* $\text{MEM}_{n,s}$ *that answers membership queries by probing* $O(1)$ *bits and that has length* $O(s^{1+\eta} \log n)$*, where* η *can be set arbitrarily small.*

For arbitrary alphabet size we obtain a similar result.

Theorem 2 (Informal)**.** *Let* $f : D \times Q \to A$ *be a data structure problem. Then there exists an RECD for* f *that answers queries by probing* $O(\log |A|)$ *bit and that has length* $O(|Q|^{1+\eta} \log |A|)$*, where* η *can be set arbitrarily small.*

These results have immediate applications to constructing efficient relaxed error-correcting data structures for the versions of MEMBERSHIP, PREDECESSOR, RANK, and NEAREST NEIGHBOR.

2 Preliminaries

We use $[n]$ to denote $\{1, \ldots, n\}$, and often switch back and forth between subsets of $[n]$ and the n-bit strings that are the characteristic vectors of those subsets. For instance, sets $S \subseteq [n]$ of size at most s correspond to n-bit strings of Hamming weight at most s.

We next list a few basic data structure problems for which we later obtain efficient RECDs.

Membership: This is the most basic data structure, studied from different perspectives in the literature. Constructing an error-correcting data structure for membership is equivalent to constructing a locally decodable code (LDC). The construction of LDCs has received a lot of attention, and it has numerous applications and connections in various areas such as probabilistically checkable proofs [2], private information retrieval [22], or hardness reductions [3].

- ⋆ MEMBERSHIP: Given a subset of a universe of size n, determine if a given element i is in that set.
  $D = \{0,1\}^n$, $Q = [n]$, $A = \{0,1\}$, $\text{MEM}_n(x, i) = x_i$
- ⋆ s-OUT-OF-n MEMBERSHIP: This is the membership problem to sparse sets.
  $D = \{x \in \{0,1\}^n : |x| \leq s\}$, $Q = [n]$, $A = \{0,1\}$, $\text{MEM}_{n,s}(x, i) = x_i$

Predecessor: This is another common data structure problem, and very tight time/space trade-offs were obtained in [18, 4] for the noiseless case. The problem has been considered in various models, including cell-probe and RAM models, for both its static as well as dynamic variants. We also consider a weaker, decision version of the problem, where the queries have binary answers.

- ⋆ PREDECESSOR SEARCH: Given a subset of an ordered universe of size n, and a specific element, find the closest predecessor of that element in the set.
  $D = \{0,1\}^n$, $Q = [n]$, $A = \{0, \ldots, n\}$, $\text{PREDSEARCH}_n(x, i) = \max\{j : j < i, x_j = 1\}$ (where $\max(\emptyset) = 0$)
- ⋆ PREDECESSOR DECISION: Given a subset of an ordered universe of size n, and a specific element, decide if there is a predecessor of that element in the set.
  $D = \{0,1\}^n$, $Q = [n]$, $A = \{0,1\}$, $\text{PREDDEC}_n(x, i) = x_1 \vee \cdots \vee x_{i-1}$

Rank: Optimal bounds for this were exhibited in [17] in the noiseless case. In this work we derive some results for both the general problem, as well as for a restricted version.

- ⋆ RANK: Given a subset of an ordered universe of size n, and a specific element, find the rank of this element in the set.
  $D = \{0,1\}^n$, $Q = [n]$, $A = \{0, \ldots, n\}$, $\text{RANK}_n(x, i) = \sum_{j=1}^{i} x_i$
- ⋆ RESTRICTED BOUNDED RANK: Given a subset of size s of an ordered universe of size n, and a specific element, find the rank of this element if it is in the set.
  $D = \{0,1\}^n$, $Q = [n]$, $A = \{0, \ldots, s\}$, $\text{RRANK}_{n,s}(x, i) = \begin{cases} 0 & \text{if } x_i = 0 \\ \sum_{j=1}^{i} x_i & \text{if } x_i = 1. \end{cases}$

Nearest neighbor: Given a collection of points $\mathcal{X}$ in the Hamming cube of dimension d, and a specific point y, find a/the point in $\mathcal{X}$ that is closest to y in terms of Hamming distance.
$D = \{0,1\}^{2^d}$, $Q = \{0,1\}^d$, $A = \{0,1\}^d$, $\text{NEAR}_d(\mathcal{X}, y) = \arg\min_{x \in \mathcal{X}} \Delta(x, y)$

Polynomial evaluation: Evaluate a univariate polynomial at a specified element over a finite field. Nearly-optimal bounds for the noiseless POLYNOMIAL EVALUATION were recently obtained in [14].

The set D consists of all $x \in \{0,1\}^n$ which are bit representations of polynomials g_x in $\mathbb{F}[X]$ with $\mathbb{F} = \{0,1\}^m$ and $n = m \cdot (\deg(g_x) + 1)$. $Q = A = \mathbb{F}$, and $\text{POLYEVAL}_{n,m}(x, \alpha) = g_x(\alpha)$.

3 Queries with binary answers

In this section we consider the case where queries have binary answers, i.e., $A = \{0,1\}$. We provide efficient RECDs for some commonly studied data structure problems.

3.1 The general MEMBERSHIP problem

Our basic building block is the relaxed LDC of Ben-Sasson et al. [5] of nearly-linear length. We already mentioned this in the introduction and here state their result in more detail:

Theorem 3 (BGHSV [5])**.** *For every $\varepsilon \in (0, 1/2)$ and $\eta > 0$, there exist an integer p and positive constants c and τ, such that for every n and every $\delta \leq \tau$, there exists a $(p, \delta, \varepsilon, 1 - c\delta)$-RLDC mapping n bits into an encoding of length $O(n^{1+\eta})$.*

Equivalently, this is a $(p, \delta, \varepsilon, 1 - c\delta)$-RECD for MEM_n. Choosing $\eta > 0$ to be very small, the length $O(n^{1+\eta})$ is close to optimal (clearly, at least n is needed). By picking the error-rate δ a sufficiently small constant, we can set $\rho = 1 - c\delta$ (the fraction of queries in the good set G) very close to 1.

For an arbitrary data structure problem $f : D \times Q \to A$ with binary answer set A, we can construct an RECD with length only slightly larger than $|Q|$ and only a constant number of probes for each query. This can be achieved by writing down the answers to all the possible queries in Q and encoding this $|Q|$-bit string by the RLDC provided by Theorem 3.

Corollary 4. *Let $f : D \times Q \to \{0,1\}$ be a data structure problem. For every $\varepsilon \in (0, 1/2)$ and $\eta > 0$, there exist an integer p and positive constants c and τ, such that for every $\delta \leq \tau$, f has a $(p, \delta, \varepsilon, 1 - c\delta)$-RECD of length $O(|Q|^{1+\eta})$.*

Corollary 4 implies the existence of good RECDs in the case where $Q = [n]$. In particular, for the MEMBERSHIP and PREDECESSOR DECISION problems, we obtain nearly optimal RECDs.

Corollary 5 (MEMBERSHIP and PREDECESSOR DECISION)**.** *For every n, MEM_n and PRED_n have RECDs with $p = O(1)$ bit-probes and nearly-linear length $O(n^{1+\eta})$.*

The parameters are optimal up to a constant factor in the number of bit-probes and optimal up to a factor n^η in its length (clearly we need length at least n bits, since the answers to all queries jointly allow to reconstruct the data $x \in \{0,1\}^n$).

3.2 The sparse MEMBERSHIP problem

In many data structure applications the data is sparse. For instance in s-OUT-OF-n MEMBERSHIP we only care about storing sets of some size s much smaller than the universe size n. Since there at least $\binom{n}{s}$ different data items to encode, any data structure will need space $N \geq \log \binom{n}{s} \approx s \log n$. The RECD for MEMBERSHIP from the end of the last section is of course also an RECD for the sparse version, but its length $n^{1+\eta} \gg s \log n$ is far from optimal now. In this section we construct an RECD for s-OUT-OF-n MEMBERSHIP that is simultaneously close to optimal in time and space: it still uses only a constant number of bit-probes, but its length is only $O(s^{1+\eta} \log n)$ bits.

We will use the following one-probe (non-error-correcting) data structure of Buhrman et al. [7] and its properties, which we describe next.

Theorem 6 (BMRV [7])**.** *For every $\varepsilon \in (0, 1/2)$ and for every positive integers $s \leq n$, there is an $(1, 0, \varepsilon)$-ECD for $\text{MEM}_{n,s}$ of length $m = \frac{100}{\varepsilon^2} s \log n$ bits.*

Properties of the BMRV encoding: The encoding can be represented as a bipartite graph $\mathcal{G} = L \times R$ with $|L| = n$ left vertices and $|R| = m$ right vertices, and uniform left degree $d = \frac{\log n}{\varepsilon}$. $\mathcal{G}$ is an expander graph: for each set $S \subseteq L$ with $|S| \leq 2s$, its neighborhood satisfies $|N(S)| \geq \left(1 - \frac{\varepsilon}{2}\right)|S|d$. For each assignment of bits to the left vertices with at most s 1s (i.e., each $x \in \{0,1\}^n$ of weight $|x| \leq s$), the encoding specifies an assignment of bits to the right vertices (which is the m-bit encoding of x). For each $i \in [n]$ let $P_i = N(\{i\}) \subseteq [m]$. A crucial property of the encoding function $\mathcal{E}_{bmrv}$ is that for every x of weight $|x| \leq s$, if $y = \mathcal{E}_{bmrv}(x) \in \{0,1\}^m$ then $\Pr_{j \in P_i}[x_i = y_j] \geq 1 - \varepsilon$. Hence the decoder for this data structure can just probe a random index $j \in P_i$ and return the resulting bit y_j. Note that this construction is not error-correcting at all, since $|P_i|$ errors in the data structure suffice to erase all information about the ith bit of the encoded x.

By combining the BMRV encoding with the RLDC construction of Theorem 3, one easily obtains an $(O(1), \delta, \varepsilon, 1 - O(\delta))$-RECD for $\text{MEM}_{n,s}$ of length $O((s \log n)^{1+\eta})$. However, we can do better:

Theorem 7. *For every $\varepsilon \in (0, 1/2)$ and $\eta > 0$, there exist an integer p and positive constants c and τ, such that for all s and n, and every $\delta \leq \tau$, $\text{MEM}_{n,s}$ has a $(p, \delta, \varepsilon, 1 - \frac{s}{2n})$-RECD of length $O(s^{1+\eta} \log n)$.*

Note that the size of the good set G is at least $\rho n = n - \frac{s}{2}$. Hence corrupting a δ-fraction of the bits of the RECD could turn half of the correct 1-answers into "don't know," but not all. This factor $\frac{1}{2}$ can easily be reduced further.

Proof. We show the existence of such an RECD for $\varepsilon = .49$. By standard amplification techniques (i.e., $O(\log(1/\varepsilon))$ repetitions) we can reduce the error probability to any other ε. The idea is similar to the approach of [20], which divides the BMRV data structure into roughly $\log n$ disjoint blocks of roughly s bits each, and encodes such block separately with a p-probe LDC. We do something similar, using an RLDC instead of an LDC to encode each block, and need to use the expander property of the BMRV structure to show that ρ is close to 1.

Encoding. We start with a BMRV structure for encoding $n' = 20n$ bits with error probability $\frac{1}{10}$. Let $\mathcal{E}_{bmrv}$ be the encoder for a $(1, 0, \frac{1}{10})$-ECD for $\text{MEM}_{20n,s}$ of length $m = 10^4 s \log(20n)$ (from Theorem 6).

Claim 8 (from Section 2.3 of [20])**.** *We can partition the m bits into $b = 10 \log(20n)$ disjoint sets $B_1, \ldots, B_b$ of $s' = 10^3 s$ indices each, such that for each of the first n indices, there are at least $b/4$ sets k satisfying $|P_i \cap B_k| = 1$.*

We view an encoding $y \in \{0,1\}^m$ as the concatenation of b strings of s' bits each: $y = y_{B_1} \cdots y_{B_b}$. If there were no noise, it would suffice to pick a block B_k at random, and to probe and return one of the P_i-bits from y_{B_k}. In order to deal with noise, we will encode each of the blocks with a $(p, 10^5\delta, \frac{1}{100}, 1-c\delta)$-RLDC that encodes s' bits into $O(s'^{1+\eta})$ bits. For $p = O(1)$ and sufficiently small δ, such an RLDC exists by Theorem 3. Let $\mathcal{E}_{rldc}$ and $\mathcal{D}_{rldc}$ be its encoder and decoder, respectively. For $x \in \{0,1\}^n$ of weight $|x| \leq s$, the encoder $\mathcal{E}$ of our RECD for $\textsc{Mem}_{n,s}$ takes $\mathcal{E}_{bmrv}(x0^{19n}) = y_{B_1} \cdots y_{B_b}$ and encodes each block with the RLDC:

$$\mathcal{E}(x) = \mathcal{E}_{rldc}(y_{B_1}) \cdots \mathcal{E}_{rldc}(y_{B_b}).$$

The length of $\mathcal{E}(x)$ is $N = b \cdot O(s'^{1+\eta}) = O(s^{1+\eta} \log n)$.

Decoding. In order to recover x_i from a string $w \in \{0,1\}^N$ satisfying $\Delta(w, \mathcal{E}(x)) \leq \delta N$, the decoder $\mathcal{D}$ does the following on input i:

1. Pick a random $k \in [b]$ (i.e., a random set B_k).
2. If $|P_i \cap B_k| \neq 1$ then output a random bit. Else, suppose $P_i \cap B_k = \{j\}$ and run the decoder $\mathcal{D}_{rldc}(j)$ on the (possibly corrupted) encoding of the kth block. Output its answer.

Analysis. We now verify the 4 conditions of Definition 3. For Condition 1: since $\mathcal{D}_{rldc}(j)$ makes at most p probes, so does $\mathcal{D}(i)$.

For Condition 2, the intuition is that most blocks don't have much higher error-rate than the average (which is at most δ), hence we can probably recover y_j for a more-or-less random $j \in P_i$, which will probably equal x_i. To make this precise, by Markov's inequality, a randomly chosen block k has error-rate $> 10^5\delta$ with probability at most $\frac{1}{10^5}$. If the block we chose indeed has error-rate $\leq 10^5\delta$, and $P_i \cap B_k = \{j\}$, then with probability at least $\frac{99}{100}$, $\mathcal{D}_{rldc}$ outputs y_j or $\perp$. Let $\beta \geq \frac{1}{4}$ be the fraction of blocks such that $|P_i \cap B_k| = 1$. Then we obtain Condition 2:

$$\Pr[\mathcal{D}(i) \in \{x_i, \perp\}] \geq (1-\beta)\frac{1}{2} + \beta\frac{99}{100} - \frac{1}{10^5} > 0.624. \tag{1}$$

For Condition 3 we need to use the expander property of the BMRV structure. Let G_k be the indices in block B_k that are answered correctly with probability at least $\frac{99}{100}$. We showed above that a $(1 - \frac{1}{10^5})$-fraction of the blocks have error-rate at most $10^5\delta$, and by the properties of the RLDC for such k we have $|G_k| \geq (1-c\delta)|B_k|$. Set $A = \cup_{k \in [b]} B_k \backslash G_k$, then $|A| \leq c\delta m$. Intuitively, A contains the queries to bits of y where $\perp$ is a likely answer. Recall that the BMRV expander has left degree $d = 10\log(20n)$. Take δ small enough that $|A| < \frac{1}{40}sd$ (this determines the value τ of the theorem). For Condition 3, we need to show that for any such small set A, most queries $i \in [n]$ are answered correctly with probability at least 0.51. It suffices to show that for most i, most of the set P_i falls outside of A. Let $B(A) = \{i \in [n] : |N(\{i\}) \cap A| \geq \frac{d}{10}\}$ be the set of queries where P_i has a relatively large overlap with A. We show that if A is small then $B(A)$ is small:

Claim 9. *For every A with $|A| < \frac{1}{40}sd$, it is the case that $|B(A)| < \frac{s}{2}$.*

Proof. Suppose, by way of contradiction, that $B(A)$ contains a set S of size $s/2$. S is a set of left vertices in the underlying graph $\mathcal{G}$, while A is a set of right vertices. Since $|S| < 2s$ and $\mathcal{G}$ is an expander, its neighborhood satisfies

$$|N(S)| \geq (1 - \frac{1}{20})d|S|.$$

By construction, each vertex in S has at most $\frac{9}{10}d$ neighbors outside A. We can therefore upper bound the size of $N(S)$ as follows:

$$|N(S)| \leq |A| + \frac{9}{10}d|S| < \frac{1}{40}ds + \frac{9}{10}d|S| = \frac{1}{20}d|S| + \frac{9}{10}d|S| = (1 - \frac{1}{20})d|S|.$$

This is a contradiction, hence no such S exists and $|B(A)| < s/2$. □

Define $G = [n] \backslash B(A)$ and notice that $|G| > n - s/2$. It remains to show that each query $i \in G$ is answered correctly with probability > 0.51. We have

$$\begin{aligned} \Pr[\mathcal{D}(i) = \perp] &\leq \Pr[D \text{ probes a block with noise-rate} > 10^5\delta] + \\ &\quad \Pr[D \text{ probes a } j \in A] + \Pr[\mathcal{D}(i) = \perp \mid D \text{ probes a } j \notin A] \\ &\leq \frac{1}{10^5} + \frac{1}{10} + \frac{1}{100} < 0.111. \end{aligned}$$

Combining with Eq. (1), we have Condition 3 for all $i \in G$:

$$\Pr[\mathcal{D}(i) = x_i] = \Pr[\mathcal{D}(i) \in \{x_i, \perp\}] - \Pr[\mathcal{D}(i) = \perp] \geq 0.51.$$

Finally, Condition 4 follows from the corresponding condition of the RLDC. □

4 Queries with non-binary answers

For many natural problems, the answer set A is not binary. For instance, the problem of searching for a predecessor in an ordered list of n elements can be reformulated as $f : \{0,1\}^n \times [n] \to [n]$ where $f(x, i)$ is equal to $\max\{j : j < i, x_j = 1\}$ (where we define $\max(\emptyset) = 0$ to cover the case where i doesn't have a predecessor). Since the correct answers are strings of length $\ell \approx \log n$, for information-theoretic reasons the number of bit-probes is $\Omega(\log n)$. Using Theorem 3, we show how to achieve an RECD with $O(\log n)$ probes and length roughly $n^{1+\eta}$, for small $\eta > 0$, which is simultaneously close to optimal in time as well as space. More generally:

Theorem 10. *Let $f : D \times Q \to \{0,1\}^\ell$ be a data structure problem. For every $\varepsilon \in (0, 1/2)$ and $\eta > 0$, there exist an integer $p = O(\ell)$ and positive constants c and τ, such that for every $\delta \leq \tau$, f has a $(p, \delta, \varepsilon, 1 - c\delta)$-RECD of length $O((\ell|Q|)^{1+\eta})$.*

To prove Theorem 10, one needs to extend the proof of Corollary 4 as follows. Suppose we simply encode the $\ell|Q|$-bit string $\langle f(x,q)\rangle_{q \in Q}$ by an RLDC, and use the decoder of the RLDC to recover each of the ℓ bits of $f(x,q)$. Now it is possible that for each $q \in Q$, the decoder outputs some blank symbols $\perp$ for some of the bits of $f(x,q)$, and no query could be answered correctly. To circumvent this, we first encode each ℓ-bit string $f(x,q)$ with a good error-correcting code, then encode the entire string by the RLDC. Now if the decoder does not output too many errors or blank symbols among the bits of the error-correcting code for $f(x,q)$, we can recover it. We need a family of error-correcting codes with the following property, see e.g. page 668 in [19] for a reference.

Fact 11 ([19], Theorem 2.10, pg. 668)**.** *For every $\delta \in (0, 1/2)$ there exists $R \in (0,1)$ such that for all n, there exists a binary linear code of block length n, information length Rn, Hamming distance δn, such that the code can correct from e errors and s erasures, as long as $2e + s < \delta n$.*

Proof of Theorem 10. Fix $\varepsilon \in (0, 1/2)$.

Encoding. Let $\mathcal{E}_{ecc} : \{0,1\}^{\ell} \to \{0,1\}^{\ell'}$ be an asymptotically good binary error-correcting code (from Fact 11), with $\ell' = O(\ell)$ and relative distance δ_{ecc}, and decoder $\mathcal{D}_{ecc}$. By Theorem 3, for every $\varepsilon_{rldc} > 0$ there exist $c_{rldc}, \tau_{rldc} > 0$ such that for every $\delta \leq \tau_{rldc}$, there is a $(O(1), \delta, \varepsilon_{rldc}, 1 - c_{rldc}\delta)$-RLDC that encodes $\ell'|Q|$ bits in $O((\ell'|Q|)^{1+\eta}) = O((\ell|Q|)^{1+\eta})$ bits. Let $\mathcal{E}_{rldc}$ and $\mathcal{D}_{rldc}$ denote its encoder and decoder, respectively. We construct an RECD for f as follows. Define the encoder $\mathcal{E} : D \to \{0,1\}^N$, where $N = O((\ell' \cdot |Q|)^{1+\eta})$, as

$$\mathcal{E}(x) = \mathcal{E}_{rldc}\left(\langle\, \mathcal{E}_{ecc}(f(x,q))\,\rangle_{q \in Q}\right).$$

Decoding. The decoder $\mathcal{D}$, with input $q \in Q$ and oracle access to $w \in \{0,1\}^N$, is defined as

1. For each $j \in [\ell']$, let $r_j = \mathcal{D}_{rldc}\left((q-1)\ell' + j\right)$ and set $r = r_1 \ldots r_{\ell'} \in \{0, 1, \perp\}^{\ell'}$.
2. If the number of blank symbols $\perp$ in r is at least $\frac{\ell'}{8}$, then output $\perp$. Else, output $\mathcal{D}_{ecc}(r)$.

Analysis. Fix an $x \in D$ and $w \in \{0,1\}^N$ such that $\Delta(w, \mathcal{E}(x)) \leq \delta N$, where $\delta \leq \tau_{rldc}$. We need to argue the above encoding and decoding satisfies the four conditions of Definition 3. For the first condition, since $\mathcal{D}_{rldc}$ makes $O(1)$ probes and $\mathcal{D}$ runs this ℓ' times, $\mathcal{D}$ makes $O(\ell') = O(\ell)$ probes into w.

We now show $\mathcal{D}$ satisfies Condition 2. Fix $q \in Q$. We want to show $\Pr[\mathcal{D}^w(q) \in \{f(x,q), \perp\}] \geq 1 - \varepsilon$. By Theorem 3, for each $j \in [\ell']$, with probability at most ε_{rldc}, $r_j = f(x,q)_j \oplus 1$. So on expectation, for at most a ε_{rldc}-fraction of the indices j, $r_j = f(x,q)_j \oplus 1$. By Markov's inequality, with probability at least $1 - \varepsilon$, the number of indices j such that $r_j = f(x,q)_j \oplus 1$ is at most $\frac{\varepsilon_{rldc}}{\varepsilon} \cdot \ell'$. If the number of $\perp$ symbols in r is at least $\frac{\ell'}{8}$ then $\mathcal{D}$ outputs $\perp$, so assume the number of $\perp$ symbols is less than $\frac{\ell}{8}$. Those $\perp$'s are viewed as erasures in the codeword $\mathcal{E}_{ecc}(f(x,q))$. So if $2\frac{\varepsilon_{rldc}}{\varepsilon} + \frac{1}{8} \leq \delta_{ecc}$, then by Fact 11, $\mathcal{D}_{ecc}$ will correct these errors and erasures and output $f(x,q)$.

For Condition 3, we show there exists a large subset G of q's satisfying $\Pr[\mathcal{D}^w(q) = f(x,q)] \geq 1 - \varepsilon$. Let $y = \langle\, \mathcal{E}_{ecc}(f(x,q))\,\rangle_{q \in Q}$, which is a $\ell'|Q|$-bit string. Call an index i in y "bad" if it does not satisfy the inequality in Condition 3 of the RLDC, i.e., $\Pr[\mathcal{D}^w_{rldc}(i) = y_i] < 1 - \varepsilon$. By Theorem 3, at most a $c_{rldc}\delta$-fraction of the indices in y are bad. Let α be a positive constant, to be chosen later. Call $q \in Q$ "bad" if more than an α-fraction of the bits in $\mathcal{E}_{ecc}(f(x,q))$ are bad. By Markov's inequality, at most a $\frac{c_{rldc}\delta}{\alpha}$-fraction of all Q are bad. Define G to be the set of q's that are not bad, then $|G| \geq (1 - \frac{c_{rldc}\delta}{\alpha})|Q|$.

We now show that each $q \in G$ satisfies the inequality in Condition 3 of Definition 3. On expectation, for at most a $(\alpha + (1-\alpha)\varepsilon_{rldc})$-fraction of the ℓ' indices in r, we have $r_j \neq f(x,q)_j$. Hence by Markov's inequality, with probability at least $1 - \varepsilon$, for at most a $\frac{1}{\varepsilon}(\alpha + (1-\alpha)\varepsilon_{rldc})$-fraction of the indices in r, we have $r_j \neq f(x,q)_j$. If $\frac{2}{\varepsilon}(\alpha + (1-\alpha)\varepsilon_{rldc}) \leq \delta_{ecc}$, then by Fact 11, $\mathcal{D}_{ecc}(r)$ will output $f(x,q)$.

Condition 4 follows using the corresponding condition in the definition of an RLDC. Hence, we can conclude there exists $\tau > 0$ such that for every $\delta \leq \tau$, $\mathcal{E}$ and $\mathcal{D}$ form an $(O(\ell), \delta, \varepsilon, 1 - c_{rldc}\delta/\alpha)$-RECD. To finish the proof, one can set for instance $\varepsilon_{rldc} = \frac{\varepsilon}{8}$, $\alpha = \frac{\varepsilon}{14}$, and $\delta_{ecc} > \frac{3}{8}$ to satisfy the previous constraints. □

Applying Theorem 10, we obtain efficient constructions for several data structure problems. As mentioned before, these parameters are close to optimal in time and space, even in the noiseless case.

Corollary 12 (PREDECESSOR SEARCH, RANK, NEAREST NEIGHBOR, POLYNOMIAL EVALUATION)**.** *For every $\eta > 0$, the data structure problems* PREDSEARCH_n *and* RANK *have RECDs of length $O(n^{1+\eta})$ and $O(\log n)$ bit-probes,* NEAR_d *has an RECD of length $O(2^{d(1+\eta)})$ and $O(d)$ bit-probes, and* $\text{POLYEVAL}_{n,\log n}$ *has an RECD of length $O(n^{1+\eta})$ with $O(\log n)$ bit-probes.*

Theorem 10 can be improved when the set of non-zero answers is $o(n)$.

Theorem 13. *Let $f : D \times Q \to \{0,1\}^\ell$ be a data structure problem. Let $s = \max_{x \in D} |\{q : f(x,q) \neq 0\}|$. If $s = o(n)$, then for every $\varepsilon \in (0, \frac{1}{2})$ and $\eta > 0$, there exist an integer $p = O(\ell)$ and $\tau > 0$, such that for every $\delta \leq \tau$, f has a $(p, \delta, \varepsilon, 1 - \frac{s}{2|Q|})$-RECD of length $O((\ell s)^{1+\eta} \log \ell |Q|)$.*

Proof sketch. Use an $\text{MEM}_{O(\ell|Q|),O(\ell s)}$ encoder (from Theorem 7) instead of $\mathcal{E}_{rldc}$ in the proof of Theorem 10. □

Applying Theorem 13, we obtain an efficient construction for RESTRICTED BOUNDED RANK.

Corollary 14 (RESTRICTED BOUNDED RANK)**.** *For every $\eta > 0$, every positive integer n, and $s = o(n)$, the data structure problem $\text{RRANK}_{n,s}$ has an RECD of length $O(s^{1+\eta} \log n)$ and $O(\log s)$ bit-probes. This is close to optimal.*

5 Conclusion

We presented a relaxation of the notion of error-correcting data structures recently proposed in [20]. While the earlier definition does not allow data structures that are both error-correcting and efficient in time and space (unless an unexpected breakthrough for constant-probe LDCs happens), the present relaxed definition does allow this. We pay for that in permitting the decoder to claim ignorance on a small fraction of the possible queries, but that seems a reasonable price to pay.

The efficient data structures we presented followed quite easily from the (highly non-trivial) relaxed locally decodable codes of Ben-Sasson et al. [5]. We feel the contribution of the present paper lies not so much in its technical content, but in giving a more practical version of the definition of [20]. This opens up many questions: there are many data structure problems in the literature for which we would like to find efficient (relaxed) error-correcting data structures.

In particular, consider the problems RANK and PREDECESSOR in the sparse case, encoding an s-element set S of a universe of size n. If $s = O(\log n)$, one can trivially obtain a RECD of size $O(s \log n)$ with $O(\log^2 n)$ bit-probes: just write down S as a string of $s \log n$ bits and encode it with a good error-correcting code, and read the entire encoding when queried for an index. A very interesting open question is to exhibit an RECD of almost optimal length, say $O(s^{1+\eta} \log n)$ bits, that answers queries with the optimal number of bit-probes (which is $O(\log s)$ for RANK and $O(\log n)$ for PREDECESSOR).

At some point we might even start to care about the various constant factors hidden in our results, with a view to actual implementations and applications—both data structures and error-correcting codes are eminently practical areas, so it would not be surprising if their common generalization eventually turned out to be of practical importance as well.

References

[1] Y. Aumann and M. Bender. Fault-tolerant data structures. In *Proceedings of 37th IEEE FOCS*, pages 580–589, 1996.

[2] L. Babai, L. Fortnow, L. Levin, and M. Szegedy. Checking computations in polylogarithmic time. In *Proceedings of 23rd ACM STOC*, pages 21–31, 1991.

[3] L. Babai, L. Fortnow, N. Nisan, and A. Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.

[4] P. Beame and F. E. Fich. Optimal bounds for the predecessor problem. In *Proceedings of 31st ACM STOC*, pages 295–304, 1999.

[5] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. Vadhan. Robust PCPs of proximity, shorter PCPs and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, 2006. Earlier version in STOC'04.

[6] G. Brodal, R. Fagerberg, I. Finocchi, F. Grandoni, G. Italiano, A. Jørgenson, G. Moruz, and T. Mølhave. Optimal resilient dynamic dictionaries. In *Proceedings of 15th European Symposium on Algorithms (ESA)*, pages 347–358, 2007.

[7] H. Buhrman, P. B. Miltersen, J. Radhakrishnan, and S. Venkatesh. Are bitvectors optimal? *SIAM Journal on Computing*, 31(6):1723–1744, 2002. Earlier version in STOC'00.

[8] K. Efremenko. 3-query locally decodable codes of subexponential length. In *Proceedings of 41st ACM STOC*, 2009.

[9] I. Finocchi, F. Grandoni, and G. Italiano. Resilient search trees. In *Proceedings of 18th ACM-SIAM SODA*, pages 547–553, 2007.

[10] I. Finocchi and G. Italiano. Sorting and searching in the presence of memory faults (without redundancy). In *Proceedings of 36th ACM STOC*, pages 101–110, 2004.

[11] M. Fredman, M. Komlós, and E. Szemerédi. Storing a sparse table with $O(1)$ worst case access time. *Journal of the ACM*, 31(3):538–544, 1984.

[12] A. G. Jørgenson, G. Moruz, and T. Mølhave. Resilient priority queues. In *Proceedings of 10th International Workshop on Algorithms and Data Structures (WADS)*, volume 4619 of *Lecture Notes in Computer Science*, 2007.

[13] J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of 32nd ACM STOC*, pages 80–86, 2000.

[14] K. S. Kedlaya and C. Umans. Fast modular composition in any characteristic. In *Proceedings of 49th IEEE FOCS*, pages 146–155, 2008.

[15] I. Kerenidis and R. de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *Journal of Computer and System Sciences*, 69(3):395–420, 2004. Earlier version in STOC'03. quant-ph/0208062.

[16] P. B. Miltersen. Cell probe complexity - a survey. Invited paper at *Advances in Data Structures* workshop. Available at Miltersen's homepage, 1999.

[17] M. Pătraşcu. Succincter. In *Proceedings of 49th IEEE FOCS*, pages 305–313, 2008.

[18] M. Pătraşcu and M. Thorup. Time-space trade-offs for predecessor search. In *Proceedings of 38th ACM STOC*, pages 232–240, 2006. See also arXiv:0603043.

[19] V. S. Pless, W. C. Huffman, and R. A. Brualdi, editors. *Handbook of Coding Theory, Vol.1*. Elsevier Science, New York, NY, USA, 1998.

[20] R. de Wolf. Error-correcting data structures. In *Proceedings of 26th Annual Symposium on Theoretical Aspects of Computer Science (STACS'2009)*, pages 313–324, 2009. cs.DS/0802.1471.

[21] D. Woodruff. New lower bounds for general locally decodable codes. Technical report, ECCC Report TR07–006, 2006.

[22] S. Yekhanin. Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM*, 55(1), 2008. Earlier version in STOC'07.