

Efficient and Error-Correcting Data Structures

for membership and polynomial evaluation

Victor Chen¹ Elena Grigorescu², Ronald de Wolf³

¹MIT & Tsinghua Univ.

²MIT

³CWI

March 4, 2010

- 1 Introduction
- 2 Model of error-correction
- 3 The data structure problems

- a basic question in computer science:
store data in a space-efficient structure so that
queries about the data can be answered efficiently

- a basic question in computer science:
store data in a space-efficient structure so that
queries about the data can be answered efficiently
- a time-space tradeoff

- a basic question in computer science:
 - store data in a space-efficient structure so that queries about the data can be answered efficiently
- a time-space tradeoff
 - time: # bits probed from the storage to answer a query
 - space: # bits in the storage representation

Motivation

- vast literature on data structure

Motivation

- vast literature on data structure

some examples:

data set: a subset of integers

given x , is x in the collection of integers?

Motivation

- vast literature on data structure

some examples:

data set: a subset of integers

given x , what's the closest predecessor of x ?

Motivation

- vast literature on data structure

some examples:

data set: a subset of points in a Euclidean plane

given x , what's the nearest neighbor of x ?

Motivation

- vast literature on data structure

some examples:

data set: a univariate polynomial

given x , what's evaluation of this polynomial at x ?

- vast literature on data structure

some examples:

data set: a univariate polynomial

given x , what's evaluation of this polynomial at x ?

- these examples have efficient constructions

- vast literature on data structure

some examples:

data set: a univariate polynomial

given x , what's evaluation of this polynomial at x ?

- these examples have efficient constructions but malfunction in the presence of *noise*

- vast literature on data structure

some examples:

data set: a univariate polynomial

given x , what's evaluation of this polynomial at x ?

- these examples have efficient constructions but malfunction in the presence of *noise*
- Goal: construct error-correcting data structure for these problems

- vast literature on data structure

some examples:

data set: a univariate polynomial

given x , what's evaluation of this polynomial at x ?

- these examples have efficient constructions but malfunction in the presence of *noise*
- Goal: construct **error-correcting** data structure for these problems

- vast literature on data structure

some examples:

data set: a univariate polynomial

given x , what's evaluation of this polynomial at x ?

- these examples have efficient constructions but malfunction in the presence of *noise*
- Goal: construct **error-correcting** data structure for these **problems**

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A \quad D: \text{set of data items}$$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A \quad Q: \text{set of queries}$$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A \quad A: \text{set of answers}$$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -membership

- $x \in D$ if $x \in \{0, 1\}^n, |x| \leq s$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -membership

- $x \in D$ if $x \in \{0, 1\}^n, |x| \leq s$
- $Q = [n], A = \{0, 1\}$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -membership

- $x \in D$ if $x \in \{0, 1\}^n, |x| \leq s$
- $Q = [n], A = \{0, 1\}$
- $\text{MEM}_{n,s}(x, i) = x_i$.

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -polynomial evaluation

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -polynomial evaluation

- $g \in D$ if $g \in \mathbb{Z}_n[X]$ and $\deg(g) \leq s$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -polynomial evaluation

- $g \in D$ if $g \in \mathbb{Z}_n[X]$ and $\deg(g) \leq s$
- $Q, A = \mathbb{Z}_n$

The data structure problems

A data structure is represented by a function

$$f : D \times Q \rightarrow A$$

(s, n) -polynomial evaluation

- $g \in D$ if $g \in \mathbb{Z}_n[X]$ and $\deg(g) \leq s$
- $Q, A = \mathbb{Z}_n$
- $\text{POLYEVAL}_{n,s}(g, \alpha) = g(\alpha)$.

Summary of our work

Introduce a new model to study error-correction in data structure

Summary of our work

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

- *Membership: with constant time and near-optimal space*

Summary of our work

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

- *Membership: with constant time and near-optimal space*
- *Poly Evaluation: with “efficient” time and near-optimal space*

Summary of our work

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

- *Membership: with constant time and near-optimal space*
- *Poly Evaluation: with “efficient” time and near-optimal space*
 $\text{polylog } s \cdot \log n$

Summary of our work

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

- *Membership: with constant time and **near-optimal** space*
- *Poly Evaluation: with “efficient” time and **near-optimal** space*
 $\text{polylog } s \cdot \log n$

Summary of our work

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

- *Membership: with constant time and **near-optimal** space*
- *Poly Evaluation: with “efficient” time and **near-optimal** space $\text{polylog } s \cdot \log n$*

near-optimal: near-linear in the information-theoretic lower bound $s \log n$

Summary of our work

Introduce a new model to study error-correction in data structure

Theorem (C, Grigorescu, de Wolf)

Obtained error-correcting data structure for

- *Membership: with constant time and near-optimal space*
- *Poly Evaluation: with “efficient” time and near-optimal space*
 $\text{polylog } s \cdot \log n$

near-optimal: near-linear in the information-theoretic lower bound $s \log n$

drawback: construction is non-explicit

- 1 Introduction
- 2 Model of error-correction**
- 3 The data structure problems

- Aumann & Bender 96

Related work: pointer-based model

- Aumann & Bender 96
- a directed graph where nodes store data, and edges represent pointers

Related work: pointer-based model

- Aumann & Bender 96
- a directed graph where nodes store data, and edges represent pointers
e.g., linked lists, stacks, trees

- Aumann & Bender 96
- a directed graph where nodes store data, and edges represent pointers
e.g., linked lists, stacks, trees
- achieved fault-tolerant data structures with constant overhead

- Aumann & Bender 96
- a directed graph where nodes store data, and edges represent pointers
e.g., linked lists, stacks, trees
- achieved **fault-tolerant** data structures with constant overhead

- Aumann & Bender 96
- a directed graph where nodes store data, and edges represent pointers
 - e.g., linked lists, stacks, trees
- achieved **fault-tolerant** data structures with constant overhead
 - against adversarial, detectable faults

- Aumann & Bender 96
- a directed graph where nodes store data, and edges represent pointers
 - e.g., linked lists, stacks, trees
- achieved **fault-tolerant** data structures with constant overhead
 - against adversarial, detectable faults
 - not all data can be recovered

Related work: faulty RAM model

- introduced by Finocchi & Italiano 04

Related work: faulty RAM model

- introduced by Finocchi & Italiano 04
- many subsequent works with optimal, fault-tolerant (some dynamic) data structures

Related work: faulty RAM model

- introduced by Finocchi & Italiano 04
- many subsequent works with optimal, **fault-tolerant** (some dynamic) data structures
 - against adversarial, undetectable faults
 - hardware assumption: $O(1)$ locations are unaffected by faults
 - faults can occur inside decoding

Related work: faulty RAM model

- introduced by Finocchi & Italiano 04
- many subsequent works with optimal, fault-tolerant (some dynamic) data structures
 - against adversarial, undetectable faults
 - hardware assumption: $O(1)$ locations are unaffected by faults
 - faults can occur inside decoding

Related work: faulty RAM model

- introduced by Finocchi & Italiano 04
- many subsequent works with optimal, fault-tolerant (some dynamic) data structures
 - against adversarial, undetectable faults
 - hardware assumption: $O(1)$ locations are unaffected by faults
 - faults can occur inside decoding
- ex. for sorting n keys, with up to $\tilde{O}(\sqrt{n})$ faults,
 - the subsequence of *uncorrupted* keys can be sorted

Related work: faulty RAM model

- introduced by Finocchi & Italiano 04
- many subsequent works with optimal, **fault-tolerant** (some dynamic) data structures
 - against adversarial, undetectable faults
 - hardware assumption: $O(1)$ locations are unaffected by faults
 - faults can occur inside decoding
- ex. for sorting n keys, with up to $\tilde{O}(\sqrt{n})$ faults,
 - the subsequence of *uncorrupted* keys can be sorted
 - cannot guarantee performance on corrupted keys

- data representation viewed as a bit-string

Related work: locally decodable model (de Wolf 09)

- data representation viewed as a bit-string
- model generalizes the concept of locally decodable code (LDC)

Related work: locally decodable model (de Wolf 09)

- data representation viewed as a bit-string
- model generalizes the concept of locally decodable code (LDC)
 - LDC is equiv. to $\text{MEM}_{n,n}$

Related work: locally decodable model (de Wolf 09)

- data representation viewed as a bit-string
- model generalizes the concept of locally decodable code (LDC)
 - LDC is equiv. to $\text{MEM}_{n,n}$
 - tolerates adversarial, undetectable bit-error (up to a constant fraction)

Related work: locally decodable model (de Wolf 09)

- data representation viewed as a bit-string
- model generalizes the concept of locally decodable code (LDC)
 - LDC is equiv. to $\text{MEM}_{n,n}$
 - tolerates adversarial, undetectable bit-error (up to a constant fraction)
 - studied static data structures: membership, inner product

- data representation viewed as a bit-string
- model generalizes the concept of locally decodable code (LDC)
 - LDC is equiv. to $\text{MEM}_{n,n}$
 - tolerates adversarial, undetectable bit-error (up to a constant fraction)
 - studied static data structures: membership, inner product
 - requires every query to be answered successfully (whp)

Related work: locally decodable model (de Wolf 09)

- data representation viewed as a bit-string
- model generalizes the concept of locally decodable code (LDC)
 - LDC is equiv. to $\text{MEM}_{n,n}$
 - tolerates adversarial, undetectable bit-error (up to a constant fraction)
 - studied static data structures: membership, inner product
 - requires every query to be answered successfully (whp)

Drawback: known LDC constructions with $O(1)$ time have super-polynomial space

- similar to de Wolf's, but allows failure for a few queries

- similar to de Wolf's, but allows failure for a few queries
 - for *most* queries, decoder gives correct answers
 - for remaining queries, decoder either gives correct answer or declares “don't know”

- similar to de Wolf's, but allows failure for a few queries
 - for *most* queries, decoder gives correct answers
 - for remaining queries, decoder either gives correct answer or declares “don't know”
- generalization of relaxed LDC (RLDC) from the PCP literature

- similar to de Wolf's, but allows failure for a few queries
 - for *most* queries, decoder gives correct answers
 - for remaining queries, decoder either gives correct answer or declares “don't know”
- generalization of relaxed LDC (RLDC) from the PCP literature
 - Upshot: RLDC has near-optimal space

A formal defn

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

for every $x \in D$, w such that $\delta(E(x), w) \leq \delta$

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

for every $x \in D$, w such that $\delta(E(x), w) \leq \delta$

- D makes t bit-probes into w

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

for every $x \in D$, w such that $\delta(E(x), w) \leq \delta$

- D makes t bit-probes into w
- for every $q \in Q$, $\Pr[D(q) \in \{f(x, q), \perp\}] \geq 1 - \epsilon$

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

for every $x \in D$, w such that $\delta(E(x), w) \leq \delta$

- D makes t bit-probes into w
- for every $q \in Q$, $\Pr[D(q) \in \{f(x, q), \perp\}] \geq 1 - \epsilon$
- the set $G = \{q : \Pr[D(q) = f(x, q)] \geq 1 - \epsilon\}$
has size $\geq (1 - \lambda)|Q|$

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

for every $x \in D$, w such that $\delta(E(x), w) \leq \delta$

- D makes t bit-probes into w
- for every $q \in Q$, $\Pr[D(q) \in \{f(x, q), \perp\}] \geq 1 - \epsilon$
- the set $G = \{q : \Pr[D(q) = f(x, q)] \geq 1 - \epsilon\}$
has size $\geq (1 - \lambda)|Q|$
- if $w = E(x)$, then $G = Q$

A formal defn

$f : D \times Q \rightarrow A$ has a $(t, \delta, \epsilon, \lambda)$ -error-correcting d.s.

if there exist encoder E and decoder D such that

for every $x \in D$, w such that $\delta(E(x), w) \leq \delta$

- D makes t bit-probes into w
- for every $q \in Q$, $\Pr[D(q) \in \{f(x, q), \perp\}] \geq 1 - \epsilon$
- the set $G = \{q : \Pr[D(q) = f(x, q)] \geq 1 - \epsilon\}$
has size $\geq (1 - \lambda)|Q|$
- if $w = E(x)$, then $G = Q$

this talk: $\epsilon, \delta, \lambda$ are positive constants

- 1 Introduction
- 2 Model of error-correction
- 3 The data structure problems**

RLDC: basic building block

- RLDC: error-correcting d.s. for $\text{MEM}_{n,n}$

RLDC: basic building block

- RLDC: error-correcting d.s. for $\text{MEM}_{n,n}$
- Thm (BGHSV): for every t , there exists a RLDC making t probes and has space $n^{1+1/t}$.

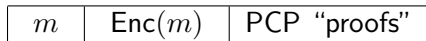
RLDC: basic building block

- RLDC: error-correcting d.s. for $\text{MEM}_{n,n}$
- Thm (BGHSV): for every t , there exists a RLDC making t probes and has space near-linear in n .

RLDC: basic building block

- RLDC: error-correcting d.s. for $\text{MEM}_{n,n}$
- Thm (BGHSV): for every t , there exists a RLDC making t probes and has space near-linear in n .
- Construction is existential, based on PCP machinery

sends message m into three pieces:



Basic principle: Compose RLDC with an appropriate noiseless d.s.

Basic principle: Compose RLDC with an **appropriate** noiseless d.s.

pseudorandom property

The decoder D is pseudorandom if for a random $q \in Q$, the bits probed by D also “behave” as if these are chosen uniformly at random.

Basic principle: Compose RLDC with an appropriate noiseless d.s.

pseudorandom property

The decoder D is pseudorandom if for a random $q \in Q$, the bits probed by D also “behave” as if these are chosen uniformly at random.

- meta-theme: noiseless d.s. with a pseudorandom decoder can be made error-correcting
 - ex. membership, poly. evaluation

Basic principle: Compose RLDC with an appropriate noiseless d.s.

pseudorandom property

The decoder D is pseudorandom if for a random $q \in Q$, the bits probed by D also “behave” as if these are chosen uniformly at random.

- meta-theme: noiseless d.s. with a pseudorandom decoder can be made error-correcting
ex. membership, poly. evaluation
- many d.s. do not have this property
e.g. those involving tree traversals

- information-theoretic lower bounds: $s \log n$ space, 1 bit-probe

Membership: overview

- information-theoretic lower bounds: $s \log n$ space, 1 bit-probe
- summary of noiseless constructions:

	Space	Time
[FKS]	$O(s \log n)$	$O(\log n)$
[BMRV]	$O(s \log n)$	1

Membership: overview

- information-theoretic lower bounds: $s \log n$ space, 1 bit-probe
- summary of noiseless constructions:

	Space	Time
[FKS]	$O(s \log n)$	$O(\log n)$
[BMRV]	$O(s \log n)$	1

- Trivial attempt at error correction
 - compose BMRV with RLDC encoding
 - near-linear in $O(s \log n)$ space, $O(1)$ time

Membership: our contribution

- a **modified** construction with $O(s^{1+\epsilon} \log n)$ space, $O(1)$ bit-probes

Membership: our contribution

- a modified construction with $O(s^{1+\epsilon} \log n)$ space, $O(1)$ bit-probes
- analysis relies on pseudorandomness of BMRV

fraction of lost data $\frac{s}{100n}$

Membership: our contribution

- a modified construction with $O(s^{1+\epsilon} \log n)$ space, $O(1)$ bit-probes
- analysis relies on pseudorandomness of BMRV

BMRV

- a bipartite expander: n left nodes, $\approx s \log n$ right nodes

Membership: our contribution

- a modified construction with $O(s^{1+\epsilon} \log n)$ space, $O(1)$ bit-probes
- analysis relies on pseudorandomness of BMRV

BMRV

- a bipartite expander: n left nodes, $\approx s \log n$ right nodes
- encoding: 0, 1 assignment to the right side

Membership: our contribution

- a modified construction with $O(s^{1+\epsilon} \log n)$ space, $O(1)$ bit-probes
- analysis relies on pseudorandomness of BMRV

BMRV

- a bipartite expander: n left nodes, $\approx s \log n$ right nodes
- encoding: 0, 1 assignment to the right side
- decoding: for $i \in [n]$, pick a random neighbor

Membership: our contribution

- a modified construction with $O(s^{1+\epsilon} \log n)$ space, $O(1)$ bit-probes
- analysis relies on **pseudorandomness** of BMRV

BMRV

- a bipartite expander: n left nodes, $\approx s \log n$ right nodes
- encoding: 0, 1 assignment to the right side
- decoding: for $i \in [n]$, pick a random neighbor
- edges from a large left subset cannot be localized in a small right subset

Polynomial evaluation: overview

- Given a polynomial $g \in \mathbb{Z}_n[X]$, $\deg(g) \leq s$

Polynomial evaluation: overview

- Given a polynomial $g \in \mathbb{Z}_n[X]$, $\deg(g) \leq s$
- information-theoretic lower bounds: $s \log n$ space, $\log n$ bit-probes

Polynomial evaluation: overview

- Given a polynomial $g \in \mathbb{Z}_n[X]$, $\deg(g) \leq s$
- information-theoretic lower bounds: $s \log n$ space, $\log n$ bit-probes
- Trivial constructions:

	Space	Time
trivial 1	$n \log n$	$\log n$
trivial 2	$s \log n$	$s \log n$

Polynomial evaluation: overview

- Given a polynomial $g \in \mathbb{Z}_n[X]$, $\deg(g) \leq s$
- information-theoretic lower bounds: $s \log n$ space, $\log n$ bit-probes
- Trivial constructions:

	Space	Time
trivial 1	$n \log n$	$\log n$
trivial 2	$s \log n$	$s \log n$

Polynomial evaluation: overview

- Given a polynomial $g \in \mathbb{Z}_n[X]$, $\deg(g) \leq s$
- information-theoretic lower bounds: $s \log n$ space, $\log n$ bit-probes
- Trivial constructions:

	Space	Time
trivial 1	$n \log n$	$\log n$
trivial 2	$s \log n$	$s \log n$

- [Miltersen]: If $\log n \geq s$, trivial 2 is essentially optimal in cell-probe

Polynomial evaluation: overview

- Given a polynomial $g \in \mathbb{Z}_n[X]$, $\deg(g) \leq s$
- information-theoretic lower bounds: $s \log n$ space, $\log n$ bit-probes
- Trivial constructions:

	Space	Time
trivial 1	$n \log n$	$\log n$
trivial 2	$s \log n$	$s \log n$

- [Miltersen]: If $\log n \geq s$, trivial 2 is essentially optimal in cell-probe
- [Kedlaya+Umans]: near-linear in $O(s \log n)$ space and $O(\text{polylog } s \log n)$ time

Theorem (CRT)

Let P be a collection of distinct primes. Then $m < \prod_{p \in P} p$ is uniquely specified by its residue $[m]_p$.

Theorem (CRT)

Let P be a collection of distinct primes. Then $m < \prod_{p \in P} p$ is uniquely specified by its residue $[m]_p$.

Consider eval. table of g over \mathbb{Z} :

$g(a_1)$	$g(a_2)$	\dots	$g(a_n)$
----------	----------	---------	----------

Theorem (CRT)

Let P be a collection of distinct primes. Then $m < \prod_{p \in P} p$ is uniquely specified by its residue $[m]_p$.

Consider eval. table of g over \mathbb{Z} :

$g(a_1)$	$g(a_2)$	\dots	$g(a_n)$
----------	----------	---------	----------

- maximum evaluation of g is n^{s+2}

Theorem (CRT)

Let P be a collection of distinct primes. Then $m < \prod_{p \in P} p$ is uniquely specified by its residue $[m]_p$.

Consider eval. table of g over \mathbb{Z} :

$g(a_1)$	$g(a_2)$	\dots	$g(a_n)$
----------	----------	---------	----------

- maximum evaluation of g is n^{s+2}
- Take P to be the first $\log n^{s+2}$ primes

Theorem (CRT)

Let P be a collection of distinct primes. Then $m < \prod_{p \in P} p$ is uniquely specified by its residue $[m]_p$.

Consider eval. table of g over \mathbb{Z} :

$g(a_1)$	$g(a_2)$	\dots	$g(a_n)$
----------	----------	---------	----------

- maximum evaluation of g is n^{s+2}
- Take P to be the first $\log n^{s+2}$ primes
- reduced polynomial $g_p := g \pmod p$

Theorem (CRT)

Let P be a collection of distinct primes. Then $m < \prod_{p \in P} p$ is uniquely specified by its residue $[m]_p$.

Consider eval. table of g over \mathbb{Z} :

$g(a_1)$	$g(a_2)$	\dots	$g(a_n)$
----------	----------	---------	----------

- maximum evaluation of g is n^{s+2}
- Take P to be the first $\log n^{s+2}$ primes
- reduced polynomial $g_p := g \pmod p$
- Store, for each $p \in P$, eval. table of g_p

Complications from encoding eval. tables by RLDC:

Error-correction with reduced polynomials

Complications from encoding eval. tables by RLDC:

- each table entry is over a non-binary alphabet

Complications from encoding eval. tables by RLDC:

- each table entry is over a non-binary alphabet

solution: a RLDC for large alphabet (code concatenation)

Complications from encoding eval. tables by RLDC:

- each table entry is over a non-binary alphabet
 solution: a RLDC for large alphabet (code concatenation)
- CRT reconstruction needs all residues to be correct

Complications from encoding eval. tables by RLDC:

- each table entry is over a non-binary alphabet
solution: a RLDC for large alphabet (code concatenation)
- CRT reconstruction needs all residues to be correct
solution: use a larger set of primes (CRT codes)

Complications from encoding eval. tables by RLDC:

- each table entry is over a non-binary alphabet
 solution: a RLDC for large alphabet (code concatenation)
- CRT reconstruction needs all residues to be correct
 solution: use a larger set of primes (CRT codes)

The extra redundancies do not affect the asymptotics

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

multi-linear extension

- write $s = d^m$, $d = \text{polylog } s$
- for $i \in [s]$, write $i = (i_0, i_1, \dots, i_{s-1})$ in base d

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

multi-linear extension

- write $s = d^m$, $d = \text{polylog } s$
- for $i \in [s]$, write $i = (i_0, i_1, \dots, i_{m-1})$ in base d
- $\psi_{s,m} : \mathbb{Z}_n[X] \rightarrow \mathbb{Z}_n[X_0, \dots, X_{m-1}]$
sends X^i to $X_0^{i_0} \cdots X_{m-1}^{i_{m-1}}$;

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

multi-linear extension

- write $s = d^m$, $d = \text{polylog } s$
- for $i \in [s]$, write $i = (i_0, i_1, \dots, i_{m-1})$ in base d
- $\psi_{s,m} : \mathbb{Z}_n[X] \rightarrow \mathbb{Z}_n[X_0, \dots, X_{m-1}]$
sends X^i to $X_0^{i_0} \cdots X_{m-1}^{i_{m-1}}$; extends multi-linearly

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

multi-linear extension

- write $s = d^m$, $d = \text{polylog } s$
- for $i \in [s]$, write $i = (i_0, i_1, \dots, i_{s-1})$ in base d
- $\psi_{s,m} : \mathbb{Z}_n[X] \rightarrow \mathbb{Z}_n[X_0, \dots, X_{m-1}]$
sends X^i to $X_0^{i_0} \cdots X_{m-1}^{i_{m-1}}$; extends multi-linearly
- max. value of $\psi(g)$ is $\approx n^{dm}$

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

multi-linear extension

- write $s = d^m$, $d = \text{polylog } s$
 - for $i \in [s]$, write $i = (i_0, i_1, \dots, i_{m-1})$ in base d
 - $\psi_{s,m} : \mathbb{Z}_n[X] \rightarrow \mathbb{Z}_n[X_0, \dots, X_{m-1}]$
sends X^i to $X_0^{i_0} \cdots X_{m-1}^{i_{m-1}}$; extends multi-linearly
-
- max. value of $\psi(g)$ is $\approx n^{dm}$
 - for $a \in \mathbb{Z}_n$, $g(a) = \psi(g)([a]_n, [a^d]_n, \dots, [a^{d^{m-1}}]_n)$

Reducing the bit-probe complexity

Culprit: max. value of g over Z is $\approx n^s$, too high

multi-linear extension

- write $s = d^m$, $d = \text{polylog } s$
- for $i \in [s]$, write $i = (i_0, i_1, \dots, i_{s-1})$ in base d
- $\psi_{s,m} : \mathbb{Z}_n[X] \rightarrow \mathbb{Z}_n[X_0, \dots, X_{m-1}]$
sends X^i to $X_0^{i_0} \cdots X_{m-1}^{i_{m-1}}$; extends multi-linearly
- max. value of $\psi(g)$ is $\approx n^{dm}$
- for $a \in \mathbb{Z}_n$, $g(a) = \psi(g)([a]_n, [a^d]_n, \dots, [a^{d^{m-1}}]_n)$
- use eval. tables of reduced polynomials of $\psi(g)$

Polynomial evaluation: summary

- reducing space to $s \log n$:
 solution: recurse and apply CRT for a second time

Polynomial evaluation: summary

- reducing space to $s \log n$:
 solution: recurse and apply CRT for a second time
- final encoding: eval. tables of reduced multivariate polynomials
 protected by layers of coding: CRT, concatenation, repetition

Polynomial evaluation: summary

- reducing space to $s \log n$:
 solution: recurse and apply CRT for a second time
- final encoding: eval. tables of reduced multivariate polynomials
 protected by layers of coding: CRT, concatenation, repetition
- decoding analysis: exploits CRT

Polynomial evaluation: summary

- reducing space to $s \log n$:
 - solution: recurse and apply CRT for a second time
- final encoding: eval. tables of reduced multivariate polynomials
 - protected by layers of coding: CRT, concatenation, repetition
- decoding analysis: exploits CRT

pseudorandomness

Because of the one-to-one CRT reconstruction, to evaluate g on a random entry, random entries in the reduced polynomials are read.

Open problems

- space-efficient, error-correcting representation for other data structures?

Open problems

- space-efficient, error-correcting representation for other data structures?
 - e.g., nearest neighbor, predecessor search
 - reducing error probability of RLDC to be sub-constant?

Open problems

- space-efficient, error-correcting representation for other data structures?
 - e.g., nearest neighbor, predecessor search
 - reducing error probability of RLDC to be sub-constant?
- a constructive encoding for RLDC?

Open problems

- space-efficient, error-correcting representation for other data structures?
 - e.g., nearest neighbor, predecessor search
 - reducing error probability of RLDC to be sub-constant?
- a constructive encoding for RLDC?
- exist data structures that are succinct *and* error-correcting?