Erasure-Resilience vs. Tolerance to Errors

Sofya Raskhodnikova, Nithin Varma
Boston University
Goal: study of sublinear algorithms resilient to adversarial corruptions in the input

Focus: Property Testing Model

[Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]
A Sublinear-Time Algorithm

randomized algorithm

approximate answer

Quality of approximation

Resources
- number of queries
- running time
Algorithms Resilient to Erasures (or Errors)

- ≤ α fraction of the input is erased (or modified) adversarially before algorithm runs
- Algorithm does not know in advance what’s erased (or modified)

**sublinear algorithm**
Property Testing

Property Tester [Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]

Two objects are at distance $\varepsilon = \text{they differ in an } \varepsilon \text{ fraction of places}$
Property Testing with Erasures

**Property Tester** [Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]

- **YES**
- **far from YES**
- \( \varepsilon \)

Accept with probability \( \geq 2/3 \)

Don’t care

Reject with probability \( \geq 2/3 \)

**Erasure-Resilient Property Tester** [Dixit Raskhodnikova Thakurta Varma 16]

- Can be completed to **YES**
- Every completion is far from **YES**
- \( \varepsilon \)

Accept with probability \( \geq 2/3 \)

Don’t care

Reject with probability \( \geq 2/3 \)

\( \leq \alpha \) fraction of the input is erased adversarially

Two objects are at distance \( \varepsilon = \) they differ in an \( \varepsilon \) fraction of places
Property Testing with Errors

Property Tester [Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]

- Two objects are at distance $\varepsilon$ = they differ in an $\varepsilon$ fraction of places
- Don’t care Accept with probability $\geq 2/3$
- Reject with probability $\geq 2/3$

Tolerant Property Tester [Parnas Ron Rubinfeld 06]

- $\leq \alpha$ fraction of the input is wrong
- Don’t care Accept with probability $\geq 2/3$
- Reject with probability $\geq 2/3$

Two objects are at distance $\varepsilon$ = they differ in an $\varepsilon$ fraction of places
Relationships Between Models

Containments are strict:

- [Fischer Fortnow 05]: standard vs. error-tolerant
- [Dixit Raskhodnikova Thakurta Varma 16]: standard vs. erasure-resilient
- new: erasure-resilient vs. error-tolerant
Main Tool: Locally List Erasure-Decodable Codes

• Locally list decodable codes have been extensively studied
  [Goldreich Levin 89, Sudan Trevisan Vadhan 01, Gutfreund Rothblum 08, Gopalan Klivans Zuckerman 08, Ben-Aroya Efremenko Ta-Shma 10, Kopparty Saraf 13, Kopparty 15, Hemenway Ron-Zewi Wootters 17, Goi Kopparty Oliveira Ron-Zewi Saraf 17, Kopparty Ron-Zewi Saraf Wootters 18]

• Only errors, not erasures were previously considered
  – Not the case without the locality restriction
    [Guruswami 03, Guruswami Indyk 05]

• Can locally list decodable codes perform better with erasures than with errors?
A Locally List Erasure-Decodable Code

- An error-correcting code $C_n : \Sigma^n \rightarrow \Sigma^N$
- Parameters: $\alpha$ fraction of erasures, list size $\ell$ and $q$ queries.

- w.p. $\geq 2/3$, for every $x \in \Sigma^n$ with encoding $C_n(x)$ that agrees with $w$ on all non-erased bits, one of the algorithms $A_j$, given oracle access to $w$, implicitly computes $x$ (that is, $A_j(i) = x_i$);

- each algorithm $A_j$ makes at most $q$ queries to $w$. 

\begin{align*}
\text{Output} & \quad A_1 \quad A_2 \quad \ldots \quad A_\ell \\
\text{(\(\alpha, \ell, q\))-local list erasure-decoder} & \\
\end{align*}
Hadamard Code

Hadamard: \( \{0,1\}^k \rightarrow \{0,1\}^{2^k} \); Hadamard(\(x\)) = \((\langle x, y \rangle)_{y \in \{0,1\}^k}\)

<table>
<thead>
<tr>
<th>Type of Corruptions</th>
<th>Corruption Tolerance (\alpha)</th>
<th>Number of Queries</th>
<th>List Size</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Errors</td>
<td>(0 \leq \alpha &lt; 1/2)</td>
<td>(O\left(\frac{1}{(1/2 - \alpha)^2}\right))</td>
<td>(O\left(\frac{1}{(1/2 - \alpha)^2}\right))</td>
<td>[Goldreich Levin 89]</td>
</tr>
<tr>
<td>Erasures*</td>
<td>(0 \leq \alpha &lt; 1)</td>
<td>(O\left(\frac{1}{1 - \alpha}\right))</td>
<td>(O\left(\frac{1}{1 - \alpha}\right))</td>
<td>[new]</td>
</tr>
</tbody>
</table>

If fraction of errors is \(\geq 1/2\), impossible to decode Hadamard codes.

*An improvement in dependence on \(\alpha\) was suggested by Venkat Guruswami*
How does separating erasures from errors in local list decoding help with separating them in property testing?
3CNF Properties: Hard to Test, Easy to Decide

• Formula $\phi_n$ : 3CNF formula on $n$ variables, $\theta(n)$ clauses
• Property $P_{\phi_n} \subseteq \{0,1\}^n$: set of satisfying assignments to $\phi_n$

Theorem [Ben-Sasson Harsha Raskhodnikova 05]

For sufficiently small $\epsilon$, 
$\epsilon$-testing $P_{\phi_n}$ requires $\Omega(n)$ queries.

• $P_{\phi_n}$ decidable by a $O(n)$-size circuit.
Testing with Advice: PCPs of Proximity (PCPPs)

[Ben-Sasson Goldreich Harsha Sudan Vadhan 06, Dinur Reingold 06]

- If $x$ has the property, then $\exists \pi(x)$ for which verifier accepts.
- If $x$ is $\varepsilon$-far, then $\forall \pi(x)$ verifier rejects with probability $\geq 2/3$.

**Theorem**

Every property decidable with a circuit of size $m$ has PCPP with proof length $\tilde{O}(m)$ and constant query complexity.

3CNF properties have efficient PCPPs
Separating Property

- $x$ satisfies the hard 3CNF property
- $r$ is the number of repetitions (to balance the lengths of 2 parts)
- $\pi(x)$ is the proof on which the PCPP verifier accepts $x$
- **Idea:** Even if a 3/4 fraction of the encoding is erased, we can still locally list erasure-decode and test with constant query complexity.
- If 1/2 fraction of encoding has errors, cannot decode the proof.
- Need $\Omega(|x|) = \tilde{\Omega}(\log N)$ queries to tolerantly test.

\[ x^r \quad \text{Hadamard}(x \circ \pi(x)) \]

\[ n \cdot r \quad 2^{\tilde{\Theta}(n)} \]

\[ N = n \cdot r + 2^{\tilde{\Theta}(n)} \]
Bottom Line

The separating property is

• erasure-resiliently testable with a constant number of queries,
• but requires $\tilde{\Omega}(\log N)$ queries to tolerantly test.

Error-tolerant testing is harder than erasure-resilient testing in general.
Open Questions and Directions

• Constant-query, constant list size, local list erasure-decodable codes with better rate?
  – Will imply better separation.

• Erasure-resilient testers for specific properties: linearity, dictatorship, linear threshold functions...

• Erasure-resilience for other models of sublinear algorithms.

Thank you!
## Separating Property: Erasure-Resilient Testing

| $x^r$ | Hadamard($x \circ \pi(x)$) |

**Idea:** If a constant fraction (say, 1/4) of the encoding is preserved, we can locally list erasure-decode.

### Erasure-Resilient Tester

1. Locally list erasure-decode Hadamard to get a list of algorithms.
2. For each algorithm, check if:
   - the plain part is $x^r$ by comparing u.r. bits with the corresponding bits of the decoding of $x$
   - PCPP verifier accepts $x \circ \pi(x)$
3. Accept if, for some algorithm on the list, both checks pass.

**Constant query complexity.**
Separating Property: Hardness of Tolerant Testing

Idea: Reduce standard testing of 3CNF property to tolerant testing of the separating property.

- Given a string \( x \), we can simulate access to

\[
\begin{array}{c|c}
\chi^r & \text{Hadamard}(x \circ \pi(x)) \\
\end{array}
\]

- All-zero string is \( \text{Hadamard}(x \circ \pi(x)) \) with 1/2 of the encoding bits are erroneous!

- Testing 3CNF property requires \( \Omega(n) \) queries, where \( n = |x| \).

The input length for separating property is \( N \approx 2^{cn} \).

\[ \Omega(n) \approx \Omega(\log N) \text{ queries are needed.} \]