Separating Erasures and Errors in Sublinear Algorithms

Sofya Raskhodnikova*, **Nithin Varma***,
Noga Ron-Zewi#

*Boston University #University of Haifa

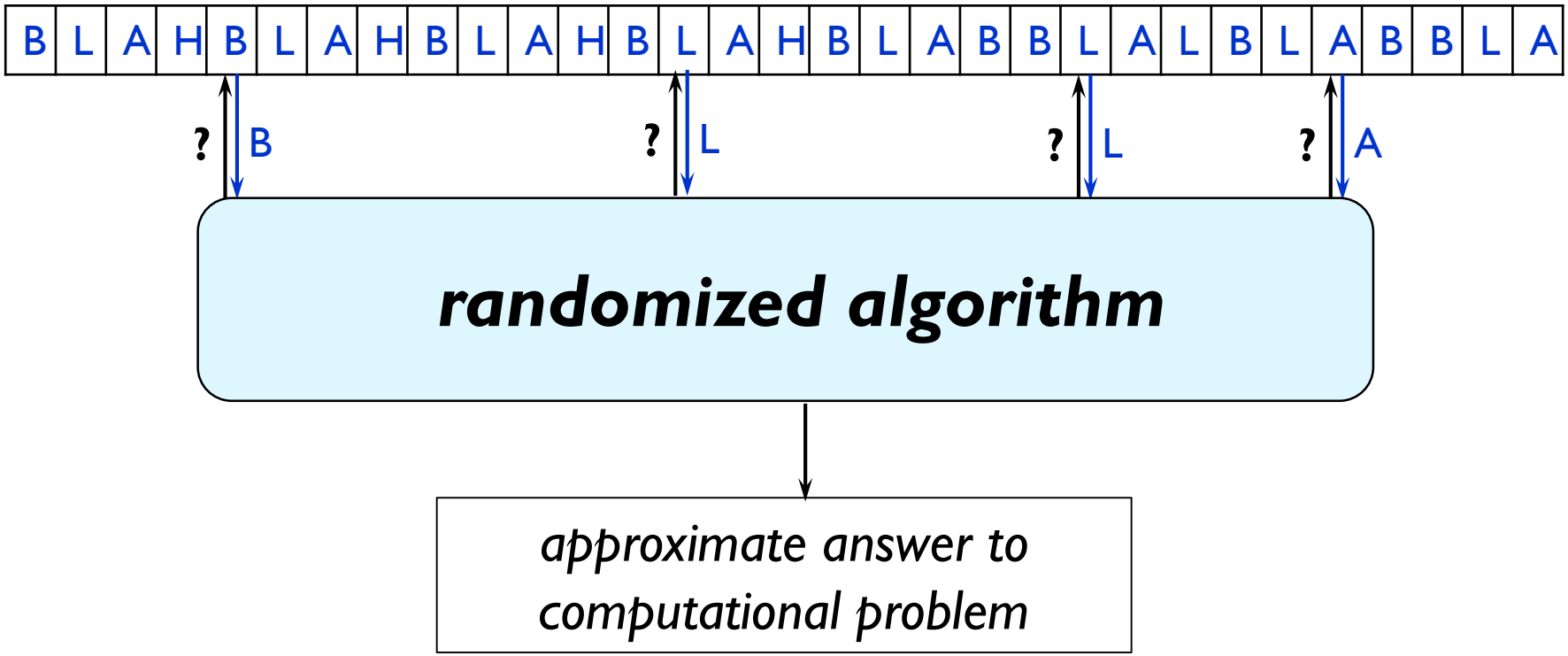
BOSTON UNIVERSITY

Goal: study of sublinear algorithms resilient to adversarial corruptions in the input

Focus: Property Testing Model

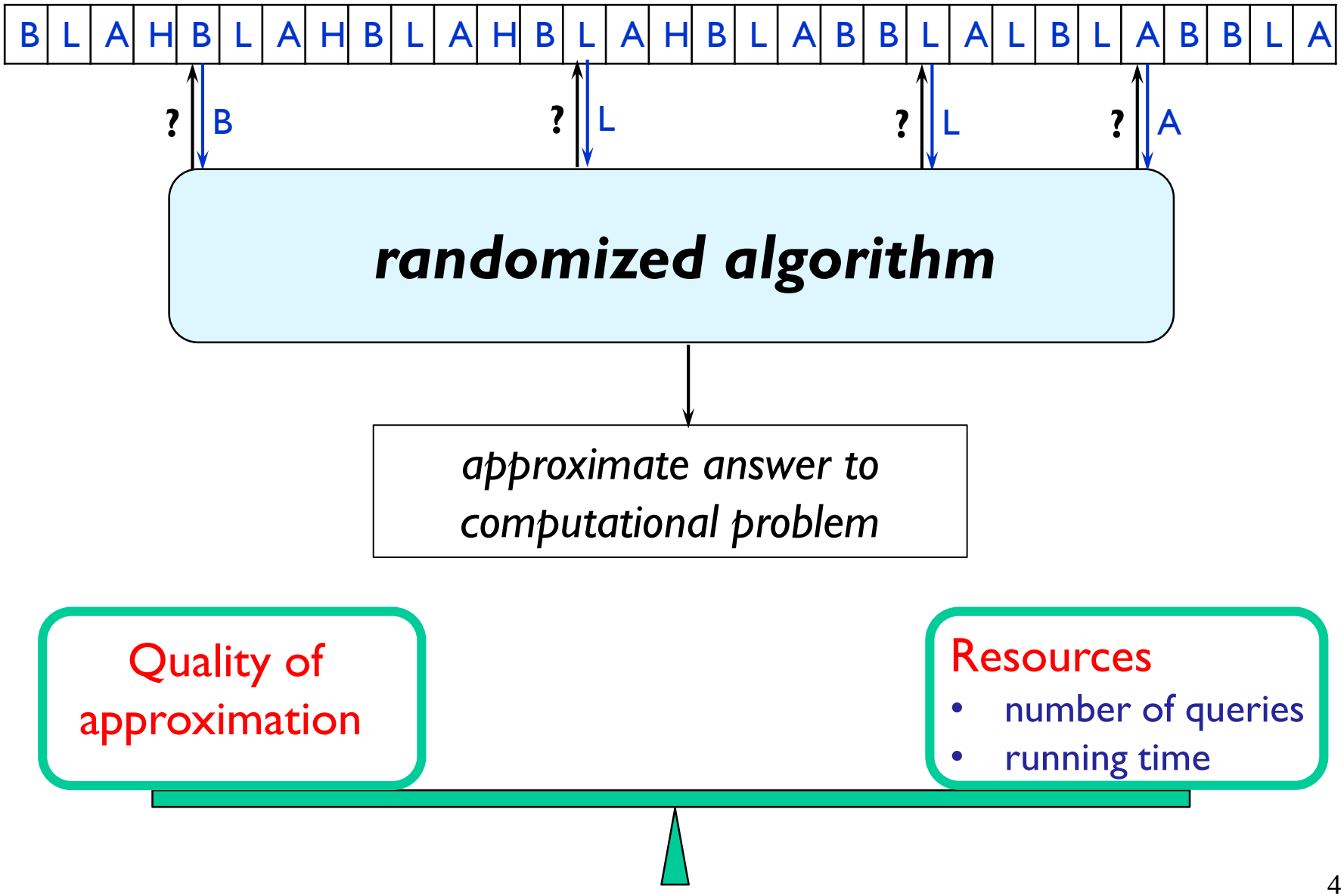
[Rubinfeld Sudan 96, Goldreich Goldwasser Ron 98]

A Sublinear Algorithm

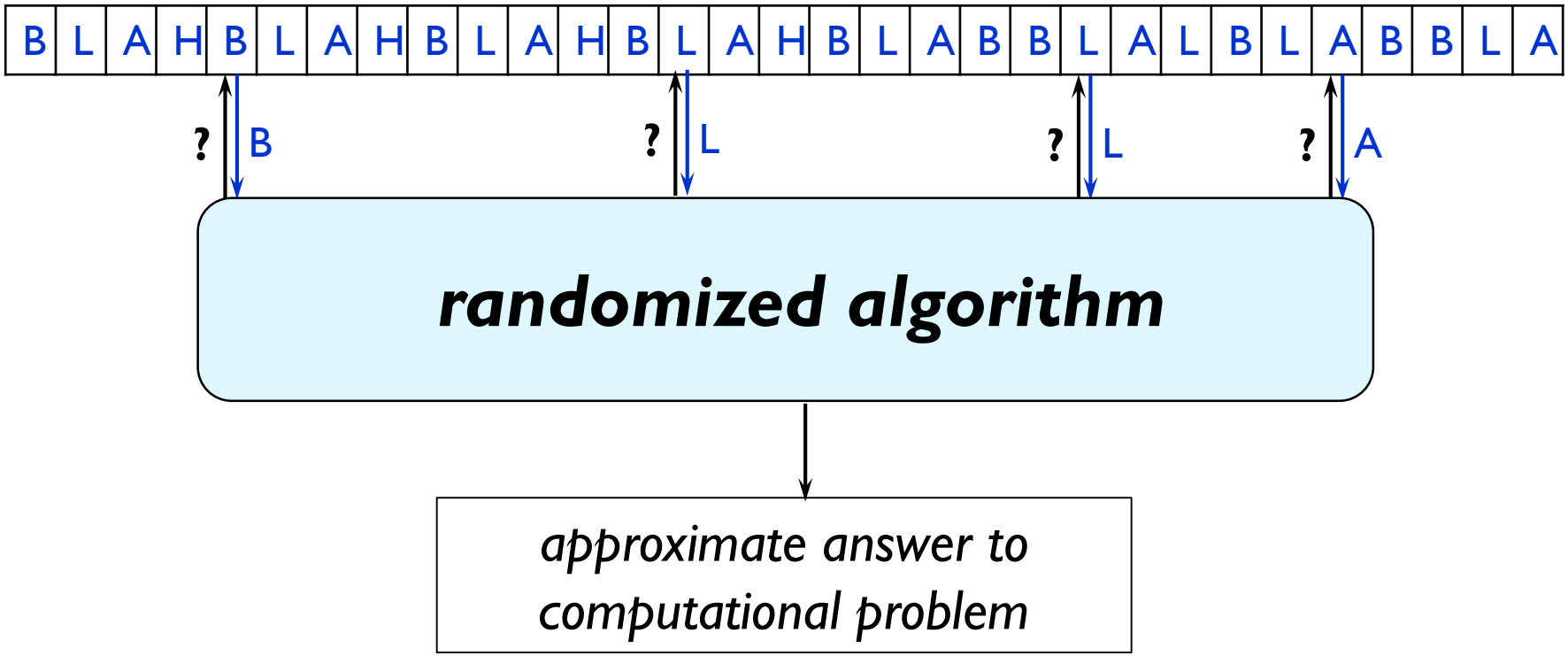


Time complexity or Query complexity sublinear in input length

A Sublinear Algorithm

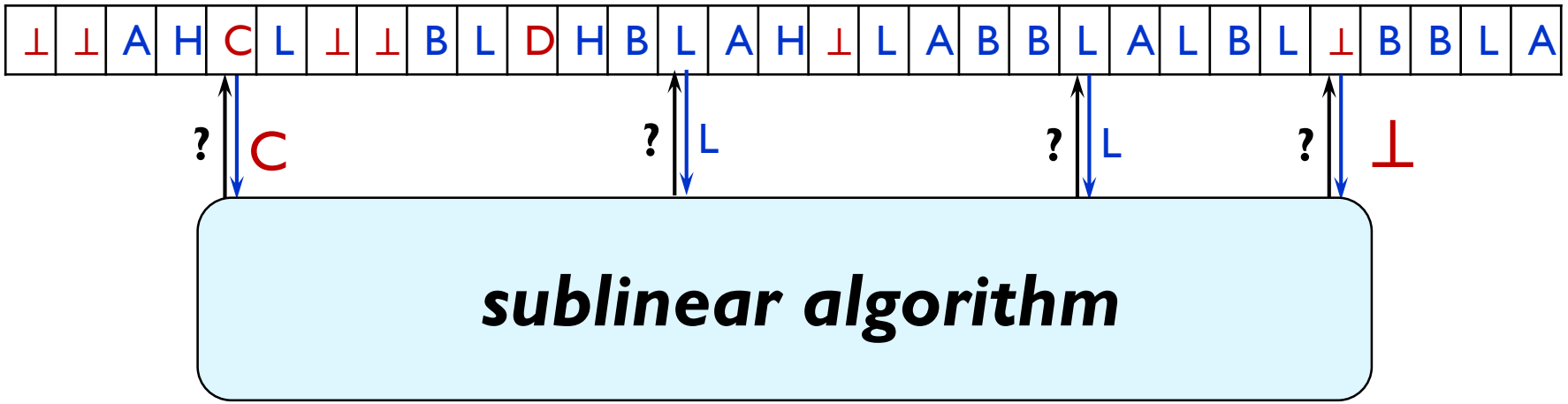


A Sublinear Algorithm



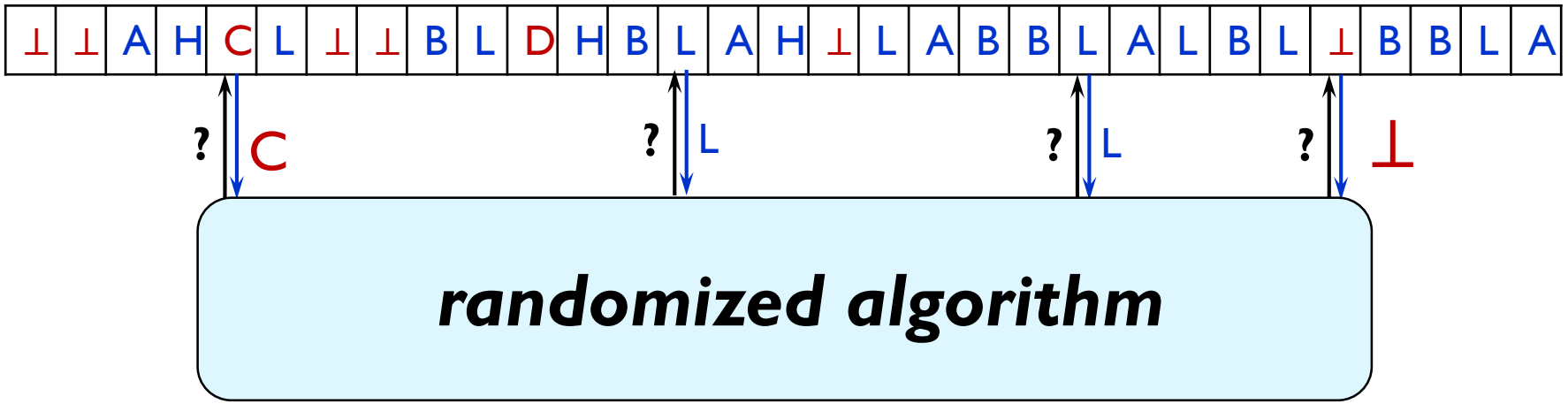
Is it always reasonable to assume that the input is corruption-free?

Corruption-Resilient Sublinear Algorithms



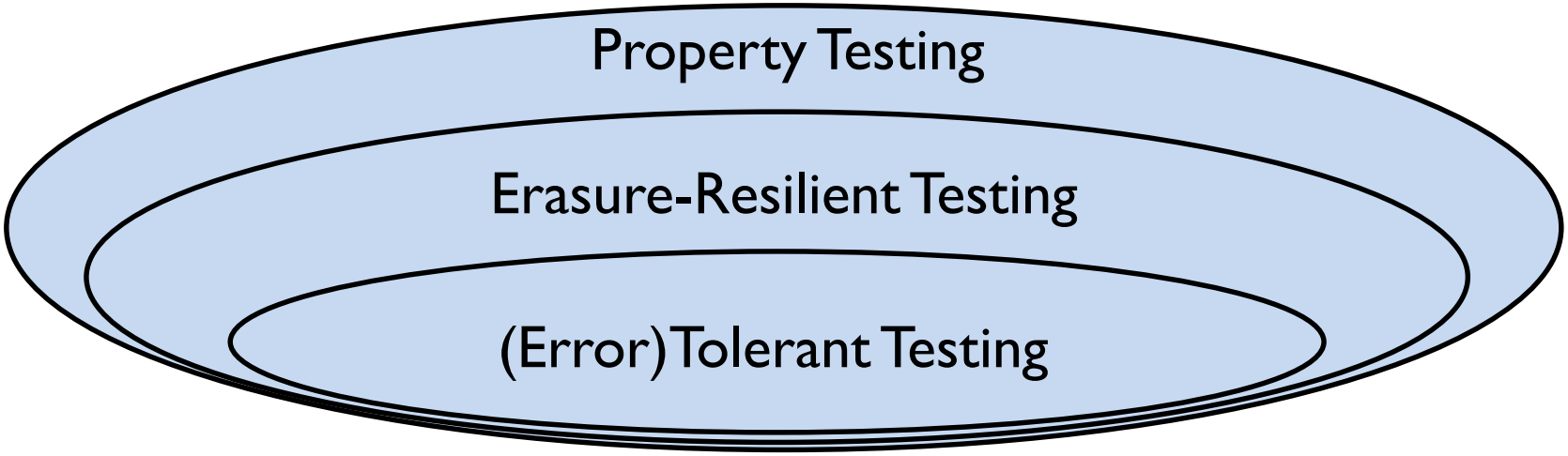
- Some fraction of the input could be erased or modified adversarially before algorithm runs.
- Algorithm does not know in advance what is erased/modified.
- Erasures can be identified at query time. Not the case with errors.

Corruption-Resilience: Motivation



- **Errors** -- **modified by an adversary**, or **noisy**
- **Erasures** -- **erased by an adversary**, or **protected**

Talk Outline

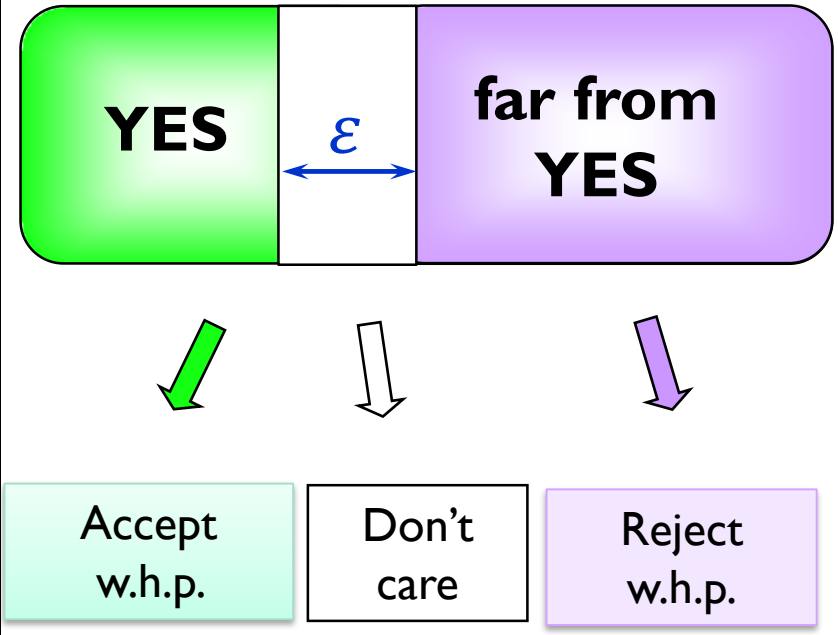


Main Result: Tolerant Property Testing is in general harder than Erasure-Resilient Property Testing

Property Testing

Property Tester

[Rubinfeld Sudan 96,
Goldreich Goldwasser Ron 98]



Property = Set of all YES instances

sorted array

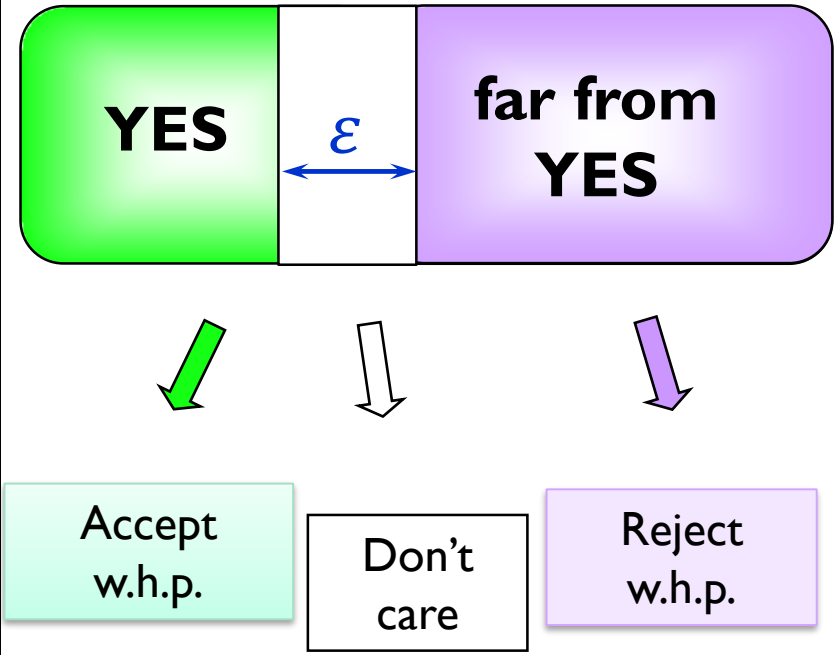
1/2-far from sorted

Two objects are at distance ε = they differ in an ε fraction of places

Property Testing with Erasures

Property Tester

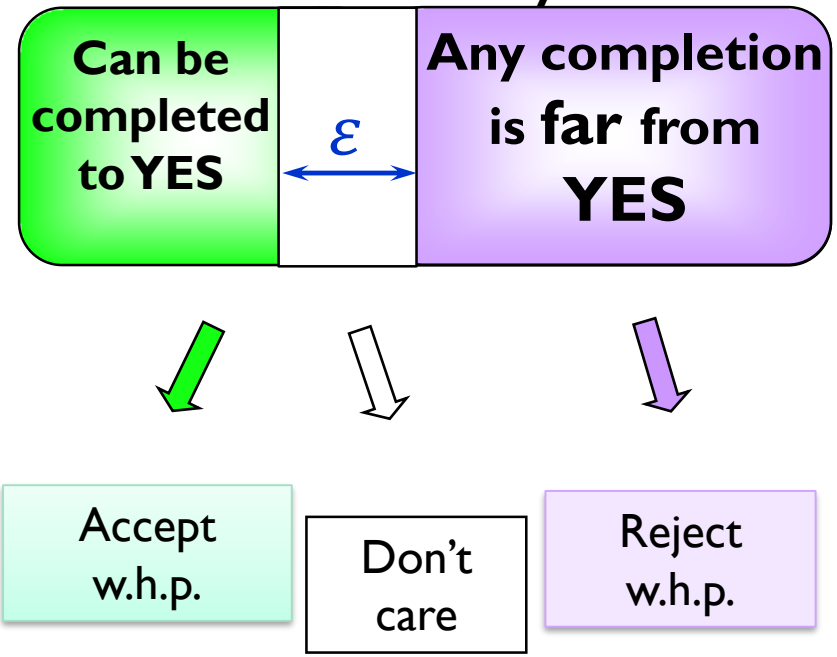
[Rubinfeld Sudan 96,
Goldreich Goldwasser Ron 98]



Erasure-Resilient Property Tester

[Dixit Raskhodnikova Thakurta Varma 16]

$\leq \alpha$ fraction of the input is erased **adversarially**

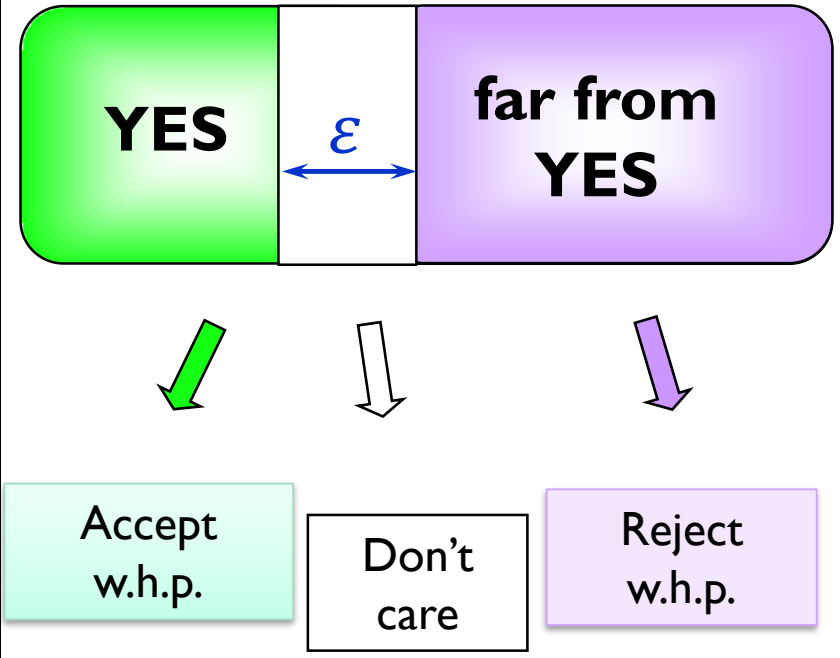


α-erasure-resilient ε-testing

Property Testing with Errors

Property Tester

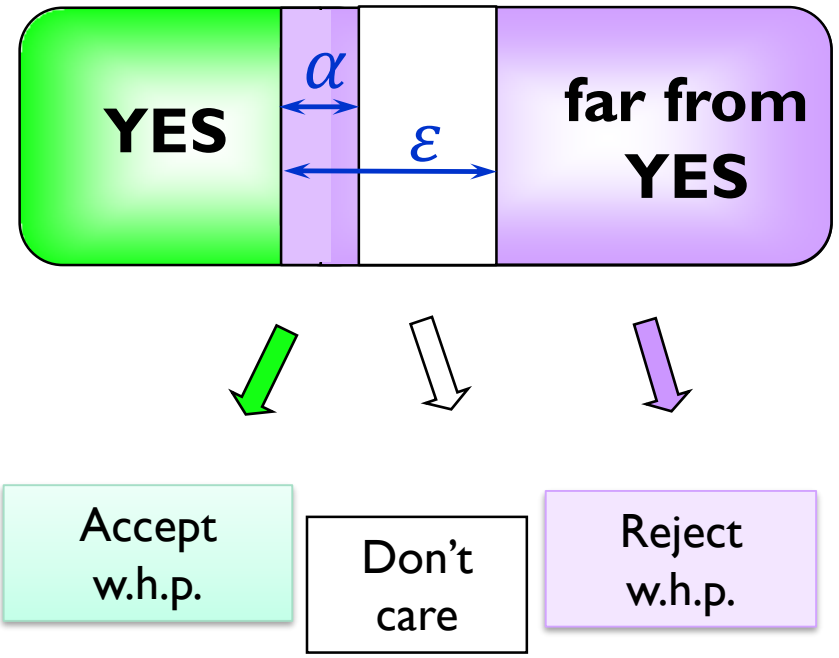
[Rubinfeld Sudan 96,
Goldreich Goldwasser Ron 98]



Tolerant Property Tester

[Parnas Ron Rubinfeld 06]

$\leq \alpha$ fraction of the input is erroneous

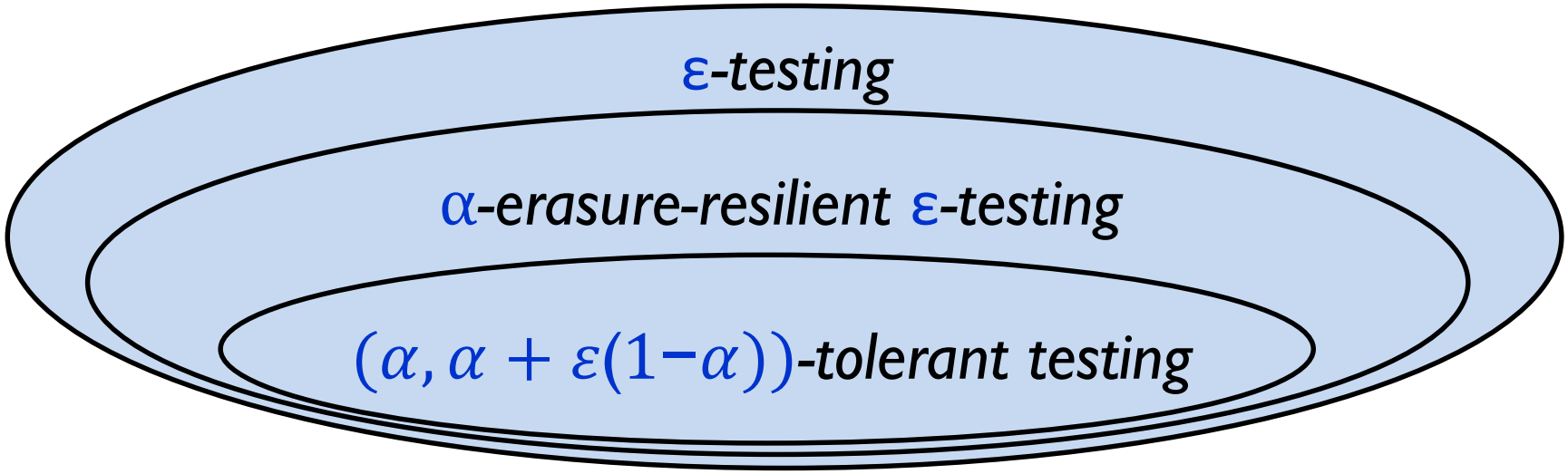


(α, ε)-tolerant testing

Relationships Between Models

Containments are strict:

- [Fischer Fortnow 05]: standard vs. tolerant
- [Dixit Raskhodnikova Thakurta Varma 16]: standard vs. erasure-resilient
- **new**: erasure-resilient vs. tolerant



Our Separation Result

α-erasure-resilient ε-testing vs. $(\alpha, \alpha + \varepsilon(1-\alpha))$-tolerant testing

Theorem: There exists a property R and constants ε, α such that

- α-**erasure-resilient** ε-**testing** R has **constant query complexity**;
- $(\alpha, \alpha + \varepsilon(1-\alpha))$-**tolerant testing** R **needs** $n^{\Omega(1)}$ **queries**.

Today - Vanilla Version of Separation

There exists a property P such that

- **erasure-resilient testing** P has **constant query complexity**;
- **tolerant testing** P **needs non-constant number of queries**.

Main Tool: Locally List Erasure-Decodable Codes

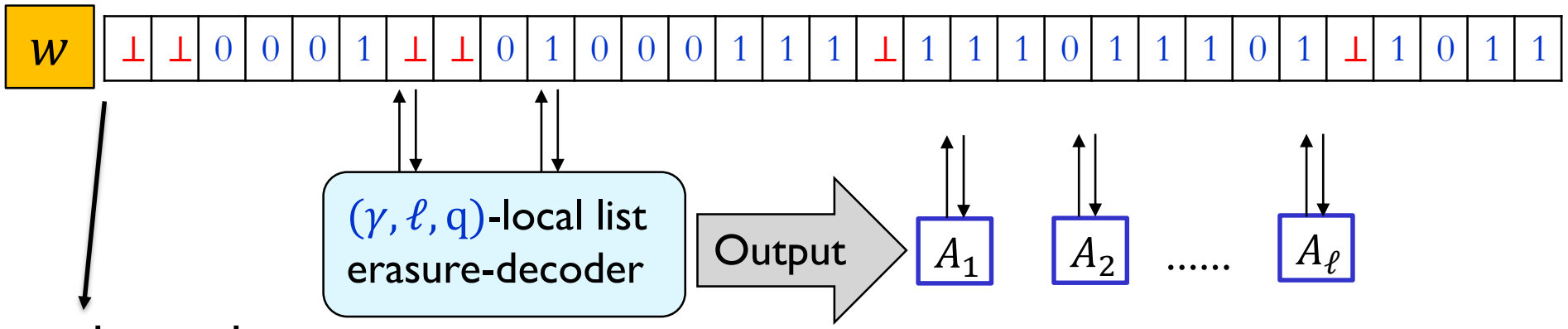
- Locally list decodable codes have been extensively studied
  [Goldreich Levin 89, Sudan Trevisan Vadhan 01, Gutfreund Rothblum 08, Gopalan Klivans Zuckerman 08, Ben-Aroya Efremenko Ta-Shma 10, Kopparty Saraf 13, Kopparty 15, Hemenway Ron-Zewi Wootters 17, Goi Kopparty Oliveira Ron-Zewi Saraf 17, Kopparty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

- Only errors, not erasures were previously considered
  - Not the case without the locality restriction
    [Guruswami 03, Guruswami Indyk 05]
  - Not the case in the approximate setting
    [Bogdanov Safra 07, Watson 15]

Can locally list decodable codes perform better with erasures than with errors?

A Locally List Erasure-Decodable Code

- An error-correcting code $\mathcal{C}_n: \Sigma^n \to \Sigma^N$, usually $N \gg n$
- Parameters: γ fraction of erasures, list size ℓ and q queries.



– w.h.p., for every $x \in \Sigma^n$ with encoding $\mathcal{C}_n(x)$ that agrees with w on all non-erased bits, one of the algorithms A_j, given oracle access to w, simulates oracle access to x;

– each algorithm A_j makes at most q queries to w.

Hadamard Code

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

Type of Corruptions	Corruption Tolerance γ	Number of Queries	List Size	Reference
Errors	$0 \le \gamma < 1/2$	$\Theta\left(\frac{1}{(1/2-\gamma)^2}\right)$	$\Theta\left(\frac{1}{(1/2-\gamma)^2}\right)$	[Goldreich Levin89, Blinovsky86, Guruswami Vadhan10, GrinbergShaltiel Viola18]
Erasures*	$0 \le \gamma < 1$	$\Theta\left(\frac{1}{1-\gamma}\right)$	$O\left(\frac{1}{1-\gamma}\right)$	[**new**, GrinbergShaltiel Viola18]

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

*An improvement in dependence on γ 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 ϕ_n : 3CNF formula on n variables, $\Theta(n)$ clauses
- Property $R_{\phi_n} \subseteq \{0,1\}^n$: set of satisfying assignments to ϕ_n

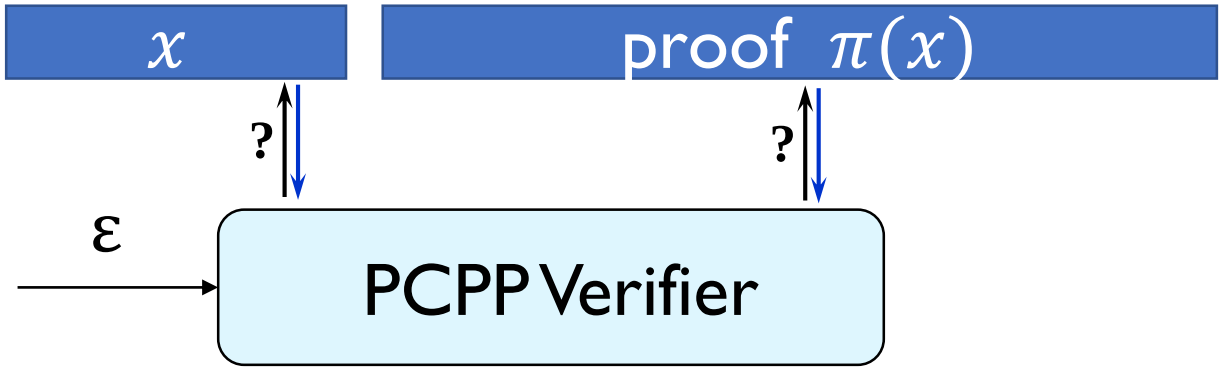
Theorem [Ben-Sasson Harsha Raskhodnikova 05]

There exists sufficiently small $\boldsymbol{\varepsilon^\star}$,
$\boldsymbol{\varepsilon^\star}$ -testing R_{ϕ_n} requires $\boldsymbol{\Omega(n)}$ queries.

- R_{ϕ_n} decidable by a $\boldsymbol{O(n)}$-size circuit.

Testing with Advice: PCPs of Proximity (PCPPs)

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



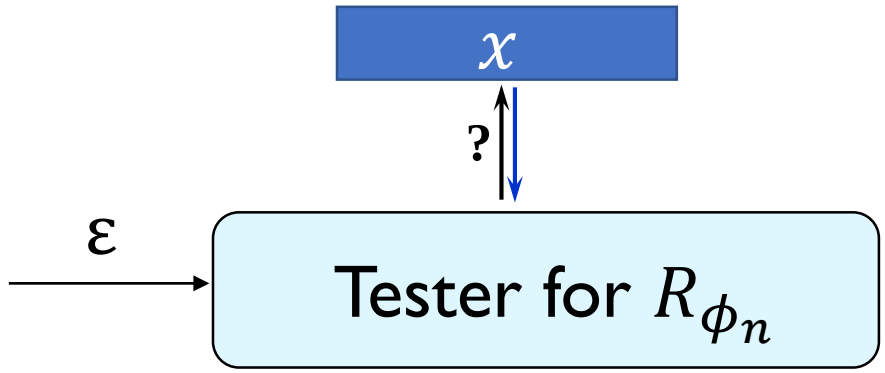
- If x has the property, then $\exists \pi(x)$ for which verifier accepts.
- If x is ε-far, then $\forall \pi(x)$ verifier rejects with probability $\geq 2/3$.

Theorem [Dinur 07]

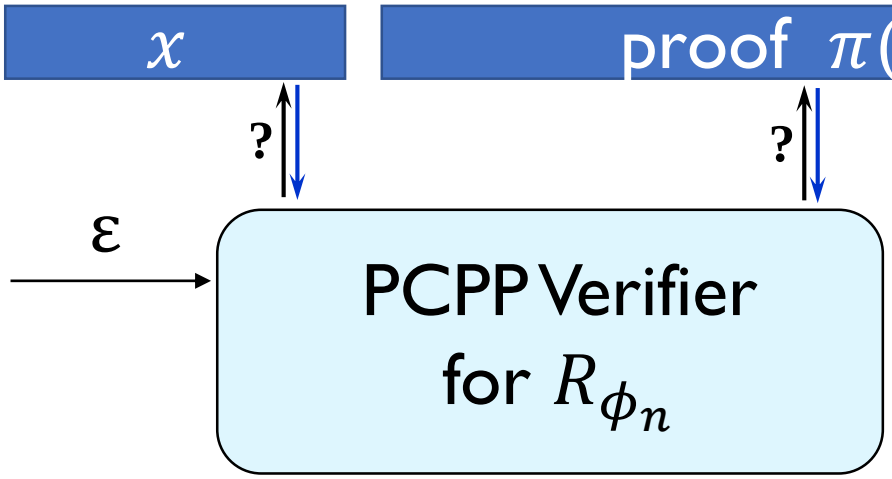
Every property decidable with a circuit of size m
has PCPP with proof length $\tilde{O}(m)$ and constant query complexity.

3CNF properties have efficient PCPPs

3CNF properties have efficient PCPPs

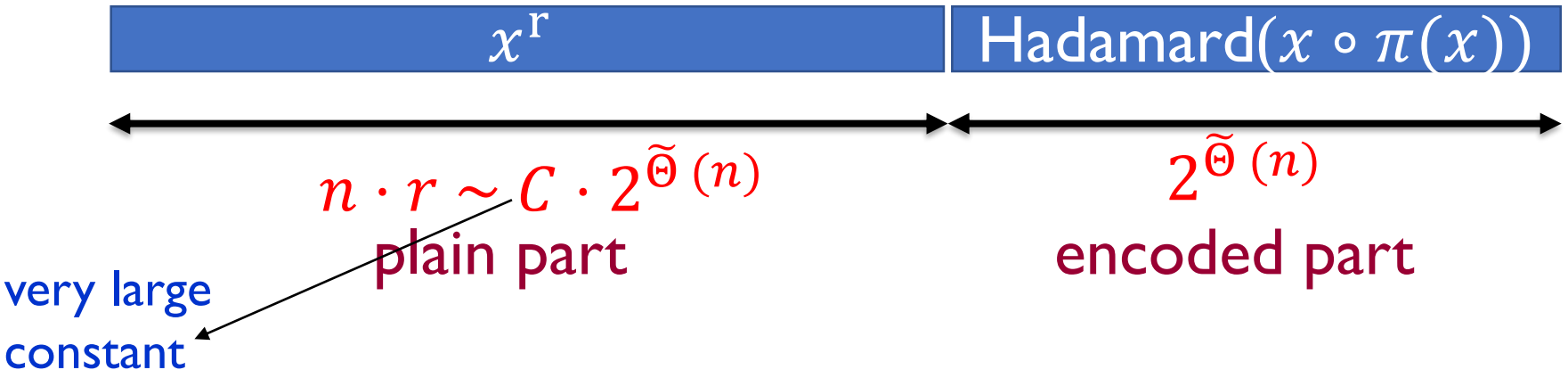


Needs $\Omega(n)$ queries to test without proof.



Constant query complexity with proof.

Separating Property *P*



- x satisfies the hard 3CNF property R_{ϕ_n}
- $\pi(x)$ is the proof on which the PCPP verifier accepts x
- r is the number of repetitions to make the length of plain part a large multiple of the length of encoded part

Separating Property: Erasure-Resilient Testing

Theorem

There is an α-erasure-resilient ε-tester for P that makes $O(\frac{1}{\varepsilon(1-\alpha)})$ queries and works for all $\alpha \in (0, 3/4C)$ and $\varepsilon \in (3/C, 1)$.

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

If a string satisfies *P*, some decoding is a `valid proof' for PCPP verifier.

If a string is far from *P*, no decoding gives a `valid proof'.

Erasure-Resilient Tester for P

Inputs: $\varepsilon \in (0,1), \alpha \in [0,1)$, oracle access to $y \in \{0,1\}^N$

1. Locally list erasure-decode Hadamard to get a list of algorithms.
2. For each algorithm:
   - **repeat** $\Theta(\frac{1}{\varepsilon(1-\alpha)})$ times: // repetition check
     - Pick a uniformly random bit in a uniformly random input block in the plain part x^r and compare it with the corresponding bit of the decoding.
     - **Reject** if both bits are nonerased and different.
   - **repeat** $\Theta(1)$ times: // PCPP check
     - Check whether PCPP verifier accepts (decoded) $x \circ \pi(x)$.
3. **Accept** if, for some algorithm on the list, both checks pass.

Erasure-Resilient Tester for P: Analysis

- $y \in P$
  - W.h.p., there is an algorithm in the list that correctly decodes $x \circ \pi(x)$. Tester accepts.
- **y is ε-far from P**
  - **Claim 1**: Plain part of y is $2\varepsilon/3$-far from being repetitions of a string satisfying the hard 3CNF property.
    - Reason: Encoded part is `too short' compared to plain part.
  - **Claim 2**: W.h.p. either repetition check or PCPP check rejects each decoding.
    - Fix a decoding $x' \circ \pi'$.
    - If plain part is far from repetitions of x', repetition check rejects w.h.p.
    - If plain part is `close' to repetitions of $x'$, then $x'$ has to be `far' from 3CNF hard property (else contradiction to Claim 1). In this case, PCPP check rejects w.h.p.

Hardness of Tolerant Testing P

Theorem

For every $\alpha \in (1/2C, 1), \varepsilon' \in (\alpha, \varepsilon^\star)$, the query complexity of (α, ε')-tolerant testing P needs $\widetilde{\Omega}(\log N)$ queries.

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

- Given x, simulate oracle access to:

x^r	00000 … 00000

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

Hardness of Tolerant Testing P

- Want to $\varepsilon^\star$-test string $x \in \{0,1\}^n$ for the hard 3CNF property. Simulate oracle access to

- If x satisfies the 3CNF property, y is $1/2C$-close to P.
- If x is $\varepsilon^\star$-far from the 3CNF property, y is $\sim\varepsilon^\star$-far from P.
- $(1/2C, \varepsilon^\star)$-tolerant testing of $P \Rightarrow \varepsilon^\star$-testing of the hard 3CNF property.
- Testing 3CNF property requires $\Omega(n)$ queries, where $n = |x|$.

  Input length for separating property is $N \sim C \cdot 2^{\tilde{\Theta}(n)}$.

$\Omega(n) \approx \tilde{\Omega}(\log N)$ queries are needed to tolerant test P.

What we proved

α-erasure-resilient ε-testing vs. $(\alpha, \alpha + \varepsilon(1-\alpha))$-tolerant testing

Theorem: There exists C, ε, α such that

- α-erasure-resilient ε-testing P has constant query complexity;
- $(\alpha, \alpha + \varepsilon(1-\alpha))$-tolerant testing P needs $\widetilde{\Omega}(\log N)$ queries.

$(C = 4/\varepsilon^{\star}, \alpha = 2/3C, \varepsilon = 16/5C)$

Error-tolerant testing is harder than erasure-resilient testing in general.

Strengthened Separation: Challenges

- Strength of lower bound related to the **rate** of the code used to encode
  - Hadamard has low (inverse exponential) rate

- To get a lower bound of $N^{\Omega(1)}$ with our construction, need locally list erasure-decodable code of inverse polynomial rate
  - (γ, q, L)-locally list erasure-decodable codes with inverse polynomial rate (with constant γ, q, L) are not known
  - Corresponding question in the case of errors is the holy grail of research on local decoding

Rate = Message length/Codeword length

Strengthened Separation

- **Helpful Observation:** Queries of the PCPP verifier can be made 'nearly uniform' over proof indices
  [Dinur 07] + [Ben-Sasson Goldreich Harsha Sudan Vadhan 06, Guruswami Rudra 05]
  - No need to decode the every proof bit

- **Idea**: Encode the proof with approximate locally list decodable codes that decode a constant fraction of proof bits correctly.
  - Approximate locally list decodable codes of inverse-polynomial rate known [Impagliazzo Jaiswal Kabanets Wigderson 10]

Strengthened Separation

There exists a property R on string of length N that is

- erasure-resiliently testable with a constant number of queries,
- but requires $N^{\Omega(1)}$ queries to tolerantly test.

Error-tolerant testing is much harder than erasure-resilient testing in general.

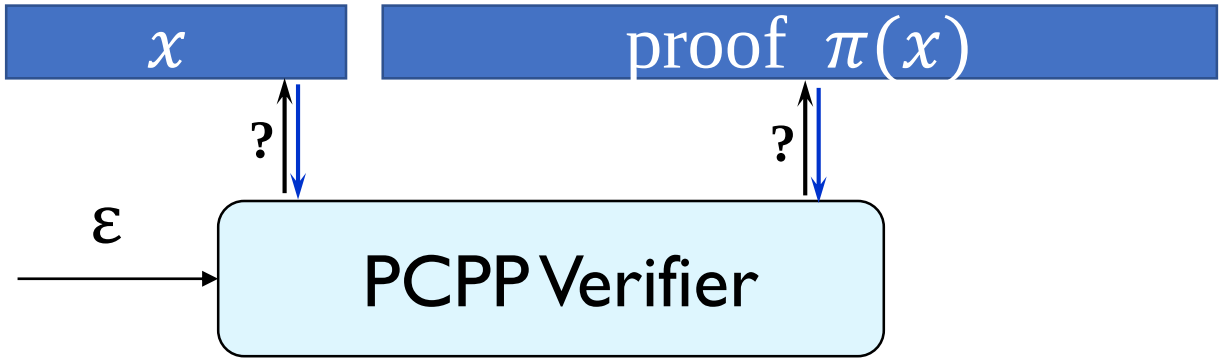
Open Questions and Directions

- Even stronger separation -- constant vs. linear ?
- Separation between errors and erasures for a "natural" property?
- Erasure-resilience versus error tolerance in other models of sublinear algorithms.
- Are locally list erasure-decodable codes provably better than locally list decodable codes ?
  - Same question in the approximate case also.
- Constant-query, constant list size, local list erasure-decodable codes with inverse polynomial rate ?

Thank you!

Proof of Strengthened Separation

Observation: PCPPs can be `smoothened'



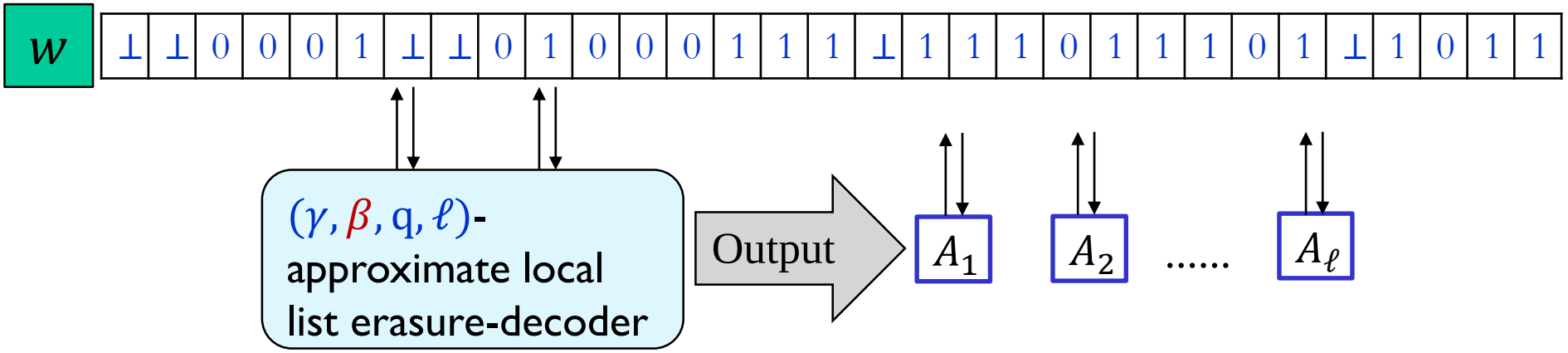
- If x has the property, then $\exists \pi(x)$ for which verifier accepts.
- If x is ε-far, then $\forall \pi(x)$ verifier rejects with probability $\geq 2/3$.
- **Smoothness:**
  - Each input query goes to an input position w.p. $\leq 2/|x|$.

Theorem [Dinur 07] + [Ben-Sasson Goldreich Harsha Sudan Vadhan 06, Guruswami Rudra 05]

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

Tool: Approximate Locally List Erasure-Decodable Code

- An error-correcting code $\mathcal{C}_n: \Sigma^n \to \Sigma^N$
- Parameters: γ fraction of erasures in codeword, β fraction of errors allowed in decoding, q queries, list size ℓ.



  - algorithm A_j's simulate oracle access to strings x' that are β-close to the (potential) original messages x;
  - each algorithm A_j makes at most q queries to w.

Approx. Locally List Erasure-Decodable Codes

Theorem [Impagliazzo Jaiswal Kabanets Wigderson 10]

For every constants $\beta \in (0,1), \gamma \in (0,1/2)$, there is a code family with inverse polynomial rate that is $(\gamma, \beta$, constant-query, constant-list-size)-approximate locally list decodable.

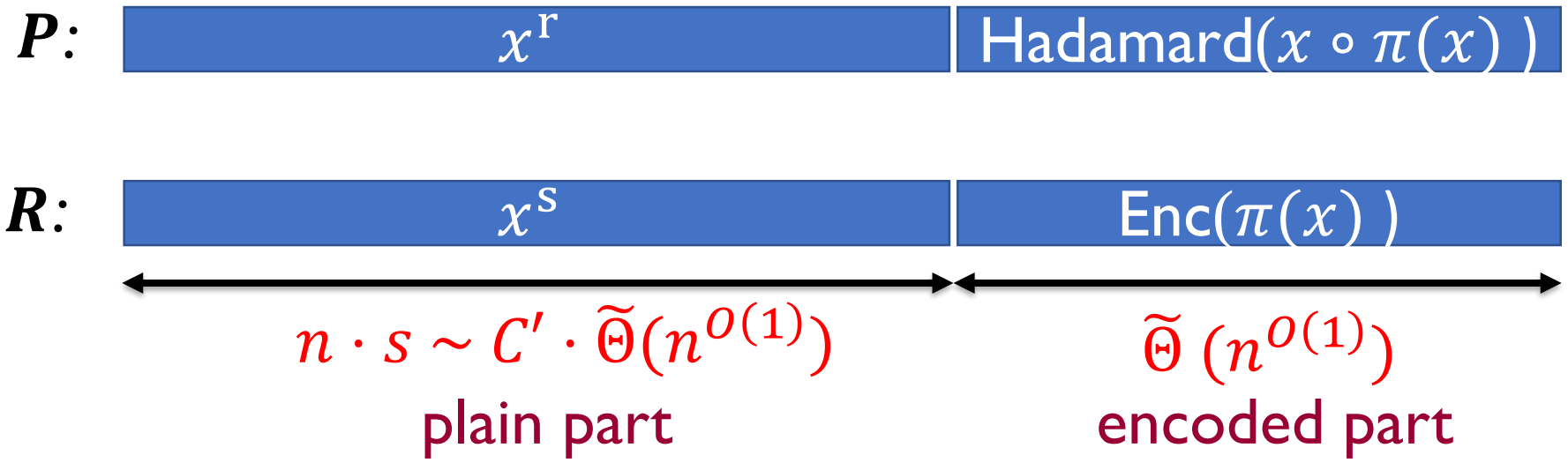
Observation*: (γ, β, q, L)-approximate locally list decodable code $\Rightarrow$ $(2\gamma, \beta, 4q, 4L)$-approximate locally list erasure-decodable code.

Theorem

For every constants $\beta, \gamma \in (0,1)$, there is a code family with inverse polynomial rate that is $(\gamma, \beta$, constant-query, constant-list-size)-approximate locally list erasure-decodable.

*Suggested by Venkat Guruswami

Separating Property R



- x satisfies the hard 3CNF property R_{ϕ_n}
- $\pi(x)$ is the proof on which the PCPP verifier accepts x
- Enc is a $(\frac{3}{4}, \beta,$ constant, constant)-approximate locally list erasure-decodable code (β - very small constant).
- s is the number of repetitions (to make the length of plain part a large multiple of the length of encoded part)

Erasure-Resilient Tester for R

Inputs: $\varepsilon \in (0,1), \alpha \in [0,1)$, oracle access to $y \in \{0,1\}^N$

1. **repeat** $\Theta(\frac{1}{\varepsilon})$ times: // repetition check
   - Sample $a \in [n]$, $i \in [s]$ u.r. until $y[(i-1)n+a]$ is nonerased.
   - Sample $j \in [s]$ u.r. until $y[(j-1)n+a]$ is nonerased.
   - **Reject** if the bits are different.
2. Locally list erasure-decode Enc to get a list of algorithms.
3. For each algorithm:
   - **repeat** $\Theta(1)$ times: // PCPP check
     - Sample $a \in [n]$, $i \in [s]$ u.r. until $y[(i-1)n+a]$ is nonerased.
     - Check whether PCPP verifier accepts $x' \circ \pi$, where x' is the i-th input repetition in y, and π is the decoded proof.
4. **Accept** if, for some algorithm on the list, both checks pass.

Erasure-Resilient Tester for R: Analysis

- $y \in R$
  - W.h.p., there is an algorithm in the list that decodes a string close to $\pi(x)$.
  - Since PCPP verifier queries are smooth, tester accepts.

- **y is ε-far from R**
  - **Case 1**: plain part is $\varepsilon/100$-far from being repetitions of the same string
    - repetition check rejects w.h.p.

Erasure-Resilient Tester for P': Analysis

- **y is ε-far from R**
  - **Case 2**: plain part is $\varepsilon/100$-close to repetitions of x^*.
    - **Claim 1:** x^* is $\varepsilon/2$-far from 3CNF property.
    - **Claim 2**: W.h.p. PCPP check rejects each decoding.
      - Fix a decoding π'.
      - Tester samples an input repetition x' with probability proportional to the number of nonerased points in the block.
      - W.h.p., x' is close to x^* (therefore far from CNF property) and has only a `small' fraction of erasures.
      - W.h.p., the PCPP verifier rejects $x' \circ \pi'$.

Constant number of queries to erasure-resilient test R.

Hardness of Tolerant Testing R

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

- Given x, simulate oracle access to:

x^S	00000 … 00000

- Every codeword of Enc has equal number of 0's and 1's.
- All-zero string is Enc$(\pi(x))$ with $1/2$ of the encoding bits being erroneous!
- Testing 3CNF property requires $\Omega(n)$ queries, where $n = |x|$.

  The input length for separating property is $N \sim C' \cdot \widetilde{\Theta}(n^{O(1)})$.

$\Omega(n) \approx N^{\Omega(1)}$ queries are needed to tolerant test R.

Strengthened Separation

There exists a property R on string of length N that is

- erasure-resiliently testable with a constant number of queries,
- but requires $N^{\Omega(1)}$ queries to tolerantly test.

Error-tolerant testing is much harder than erasure-resilient testing in general.