Does Locality imply Efficient Testability?

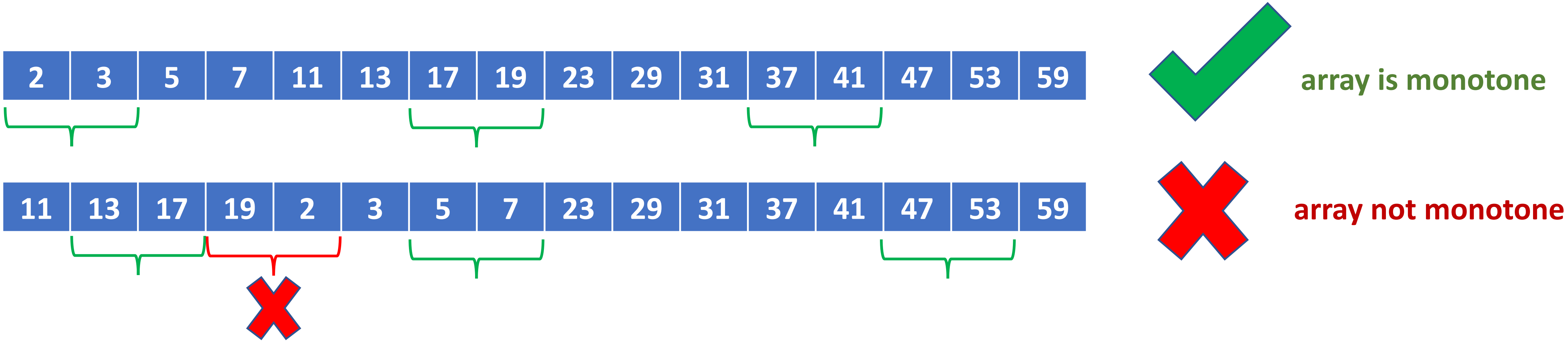
Omri Ben-Eliezer

TEL AVIV UNIVERSITY

WOLA 2019

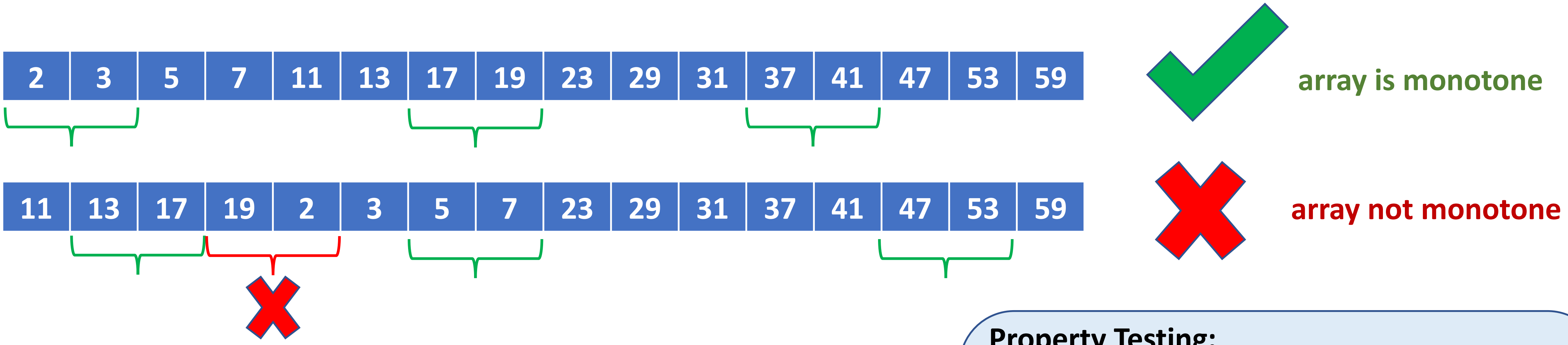
Monotonicity testing: Yet another proof...

Consider an array of numbers. Is the array monotone increasing?



Monotonicity testing: Yet another proof…

Consider an array of numbers. Is the array monotone increasing?



Property Testing:
Given query access to $A: [n] \to \mathbb{R}$ that is ε-far from being monotone increasing, how many queries needed to find (with prob. 2/3) a "proof" that A is not monotone.
ε-far:
Need to change εn entries in A to make it monotone.

Monotonicity testing: Yet another proof…

Consider an array of numbers. Is the array monotone increasing?

2	3	5	7	11	13	17	19	23	29	31	37	41	47	53	59

array is monotone

11	13	17	19	2	3	5	7	23	29	31	37	41	47	53	59

array not monotone

[Ergün, Kannan, Kumar, Rubinfeld, Viswanthan '98:]

Monotonicity is ε-testable with $O(\varepsilon^{-1}\log n)$ queries.

Property Testing:
Given query access to $A: [n] \to \mathbb{R}$ that is ε-far from being monotone increasing, how many queries needed to find (with prob. 2/3) a "proof" that A is not monotone.
ε-far:
Need to change εn entries in A to make it monotone.

Monotonicity testing: Yet another proof…

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Monotonicity testing: Yet another proof…

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Monotonicity testing: Yet another proof…

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Monotonicity testing: Yet another proof…

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Monotonicity testing: Yet another proof…

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Consider a *partitioning* of the array into intervals.

In which intervals is the *first* elements larger than the *last?*

Monotonicity testing: Yet another proof...

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Hierarchical partitioning:

each interval in level i is union of two or three intervals from level $i-1$

Monotonicity testing: Yet another proof…

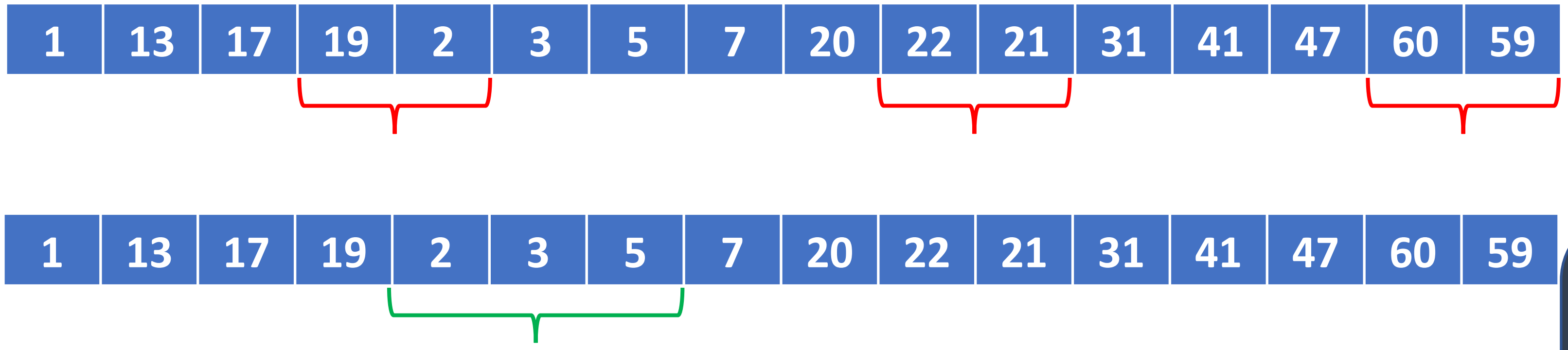
1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

Hierarchical partitioning:

each interval in level i is union of two or three intervals from level $i-1$

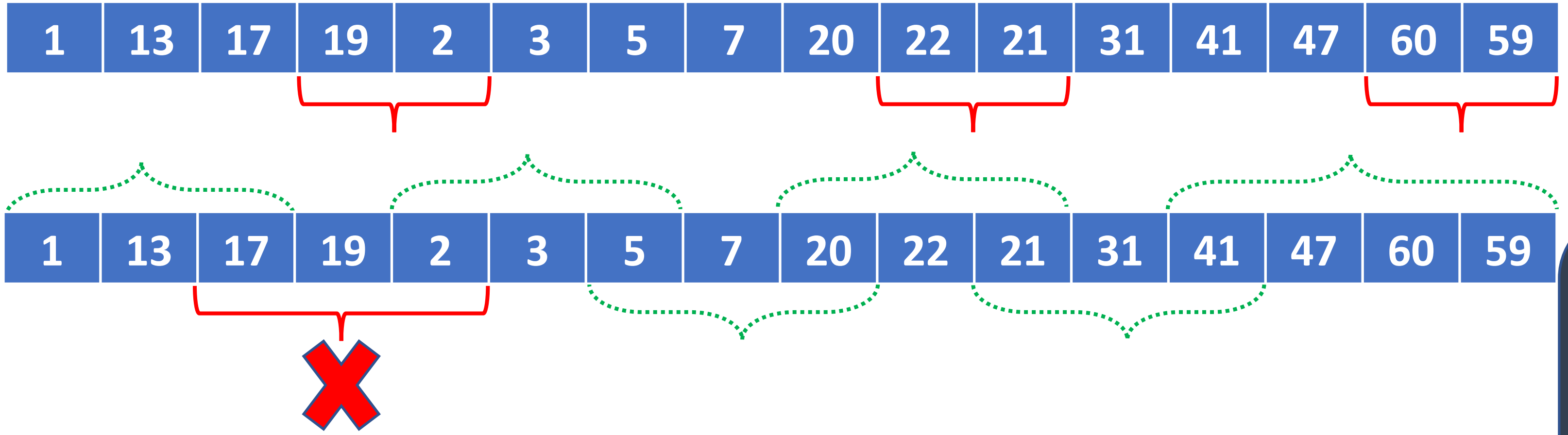
Monotonicity testing: Yet another proof…



Hierarchical partitioning:

each interval in level i is union of two or three intervals from level $i-1$

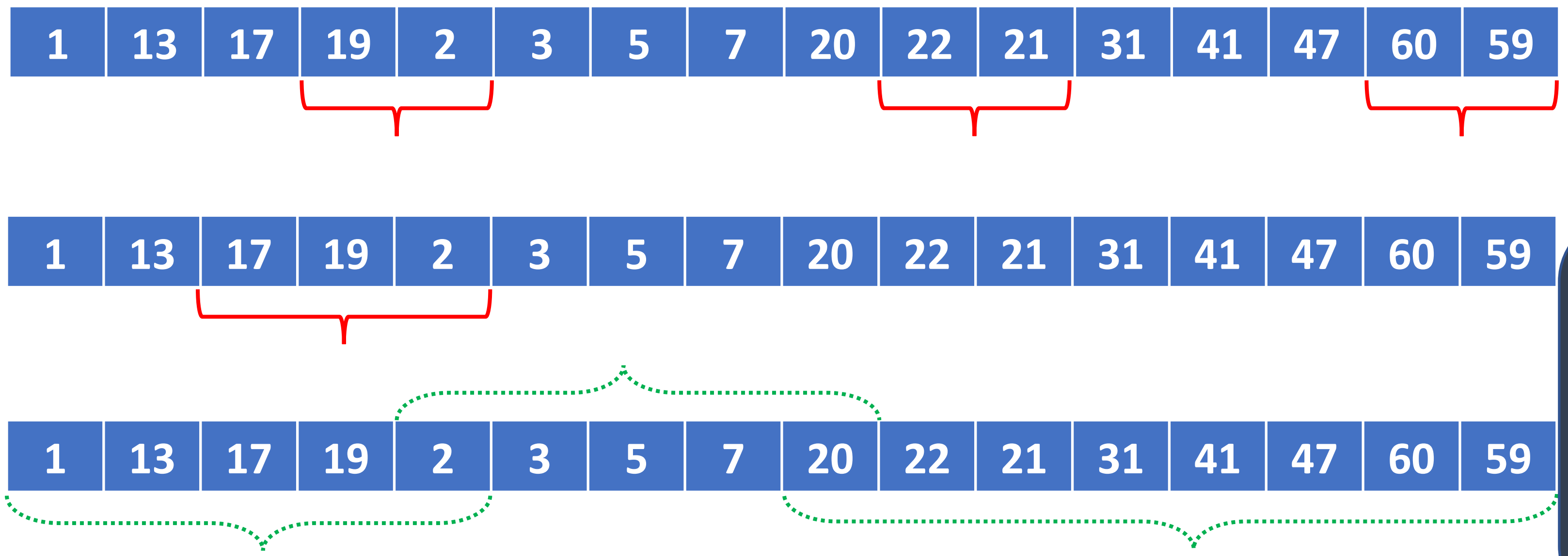
Monotonicity testing: Yet another proof…



<u>Hierarchical partitioning:</u>

each interval in level i is union of two or three intervals from level $i-1$

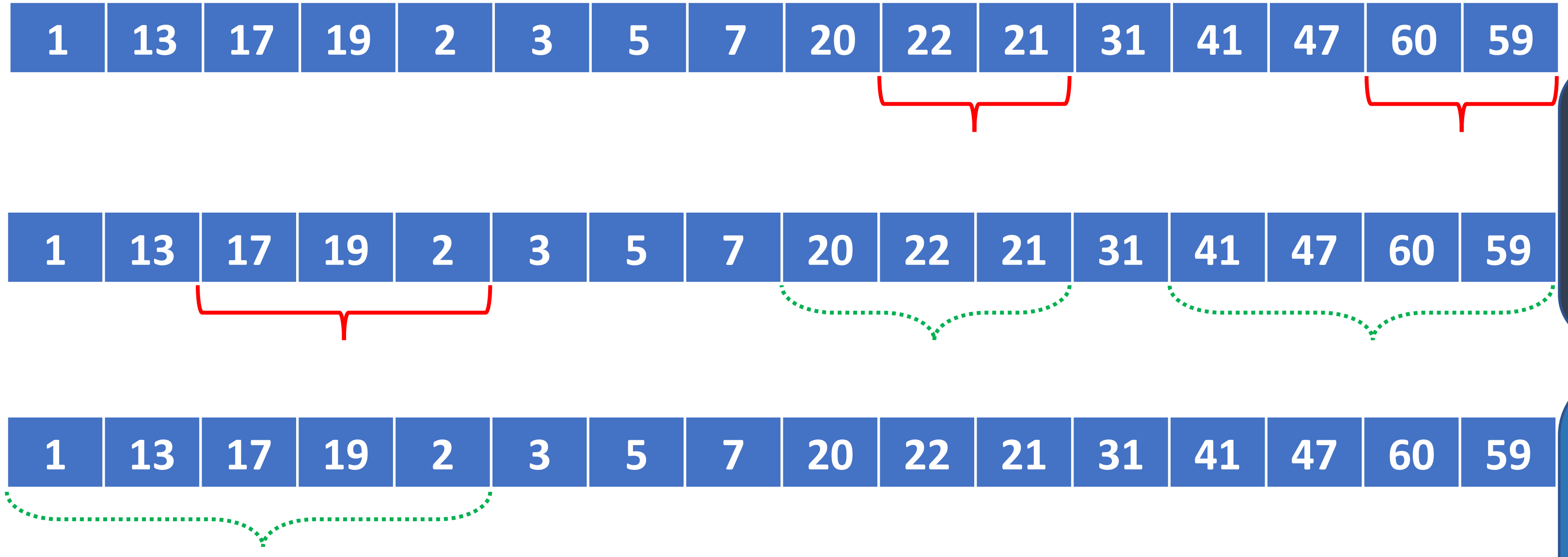
Monotonicity testing: Yet another proof…



Hierarchical partitioning:

each interval in level i is union of two or three intervals from level $i-1$

Monotonicity testing: Yet another proof…



Consider only "bad" intervals that are maximal: not contained in any other "bad" one.

Claim: Suffices to edit elements within "good" intervals that are one level above maximal "bad" ones, to make array monotone

Monotonicity testing: Yet another proof…

1	13	17	19	2	3	5	7	20	22	21	31	41	47	60	59

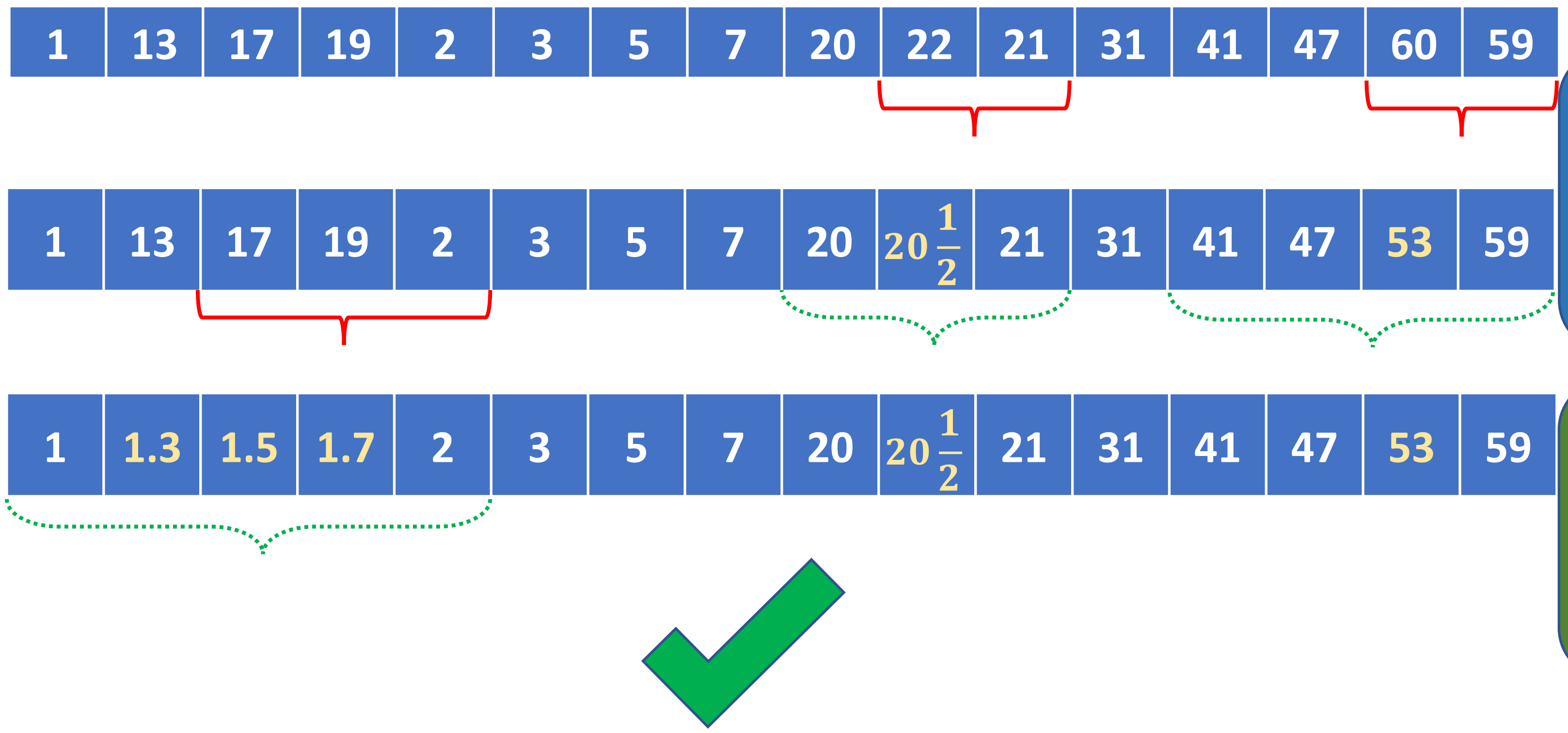
1	13	17	19	2	3	5	7	20	$20\frac{1}{2}$	21	31	41	47	53	59

1	1.3	1.5	1.7	2	3	5	7	20	$20\frac{1}{2}$	21	31	41	47	53	59

Consider only “bad” intervals that are maximal: not contained in any other “bad” one.

Claim: Suffices to edit elements within “good” intervals that are one level above maximal “bad” ones, to make array monotone

Monotonicity testing: Yet another proof…



<u>Corollary</u>: If array is ε-far from monotonicity, then set of "maximal bad intervals" has size st least $\approx \varepsilon n$

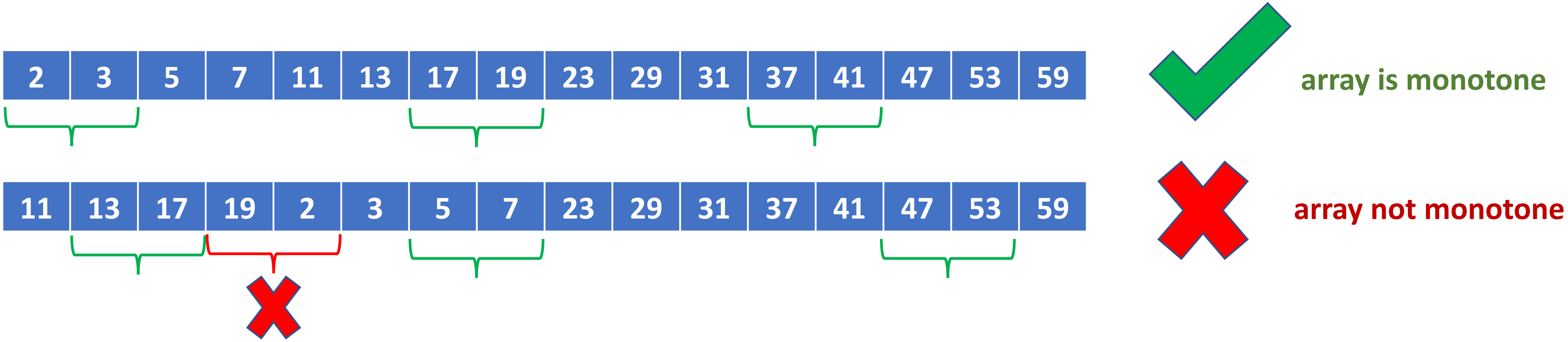
<u>The test</u>: Pick $\approx 1/\varepsilon$ intervals from each level, query their endpoints. Reject if any of them is bad. Total query complexity $\approx \varepsilon^{-1} \log n$.

Local properties

A property of arrays $A\colon [n] \to \Sigma$ is **k-local** if it can be defined by a family of **forbidden consecutive patterns** of size $\leq k$.

Examples:

Monotonicity is **2-local**. Forbidden patterns: "$A(i) > A(i+1)$"



Local properties

A property of arrays $A: [n] \to \Sigma$ is **k-local** if it can be defined by a family of **forbidden consecutive patterns** of size $\leq k$.

Examples:

Monotonicity is **2-local**. Forbidden patterns: "$A(i) > A(i+1)$"

Lipschitz-continuity is **2-local**

Convexity is **3-local**

Properties of first k discrete derivatives are **$(k+1)$-local**

Pattern matching and **computational biology** problems are **k-local for small k**

Local properties

A property of arrays $\mathrm{A}: [\boldsymbol{n}]^{\boldsymbol{d}} \rightarrow \Sigma$ is **$\boldsymbol{k}$-local** if it can be defined by a family of **forbidden consecutive patterns** of size $\leq \boldsymbol{k} \times \cdots \times \boldsymbol{k}$.

Examples:

Monotonicity is **2-local**. Forbidden patterns: "$A(i) > A(i+1)$"

Lipschitz-continuity is **2-local**

Convexity is **3-local**

Submodularity is **2-local**

Properties of first k discrete derivatives are **$(k+1)$-local**

Pattern matching problems in **computer vision** are **k-local for small k**

Local properties $\Leftrightarrow$ Local algorithms

The LOCAL model in distributed computing [Linial'87]:

Which graph properties are "locally decidable" by balls of radius k?

Our setting a bit different:

1. **Graph topology known in advance**: graph is the line (for d=1) / hypergrid (d>1).
2. However, each **vertex** holds a **value** (**not known in advance**).

Claim: Property is k-local $\Leftrightarrow$ has local algorithm (known topology, unknown values) with $\Theta(k)$ rounds

Generic test for local properties

Theorem [B., 2019]:

Any k-local property $\mathcal{P}$ of $[n]^d$-arrays over any finite alphabet Σ is ε-**testable** using

$$O\left(\frac{k \log n}{\varepsilon}\right) \text{ queries for } d = 1$$

$$O_d\left(\frac{kn^{d-1}}{\varepsilon^{1/d}}\right) \text{ queries for } d > 1$$

Property Testing:
Given property $\mathcal{P}$, parameter ε, and query access to A: $[n]^d \to \Sigma$, distinguish with prob. 2/3 between the cases:

- A satisfies $\mathcal{P}$
- A is ε-far from $\mathcal{P}$: need to change εn^d values in A to satisfy $\mathcal{P}$

Generic test for local properties

Theorem [B., 2019]:

Any k-local property $\mathcal{P}$ of $[n]^d$-arrays over any finite alphabet Σ is ε-**testable** using

$$O\left(\frac{\boldsymbol{k \log n}}{\varepsilon}\right) \text{ non-adaptive queries for } d = 1$$

$$O_d\left(\frac{\boldsymbol{kn^{d-1}}}{\varepsilon^{1/d}}\right) \text{ non-adaptive queries for } d > 1$$

The good news: Test is canonical (queries depend on d, k, ε, n, but not on $\mathcal{P}$, Σ);
proximity oblivious (repetitive iterations of the same "basic" test);
non-adaptive (makes all queries in advance); and has
one-sided error.

Allows "**sketching for testing**".

The bad news: linear running time for $d = 1$; exponential for $d > 1$ ☹

The main idea: Unrepairability

$\mathcal{P}$: a 2-local property of 1D arrays $A: [n] \to \Sigma$.

An interval $I = \{a, a+1, \ldots, b\} \subseteq [n]$ is **unrepairable** (w.r.t A, $\mathcal{P}$) if, no matter how we modify $A(a+1), \ldots, A(b-1)$, the sub-array of A between a and b will **never** satisfy $\mathcal{P}$.

Observation: Enough to query only $f(a)$ and $f(b)$ to know if I is unrepairable.

Example: unrepairable interval for **monotonicity**.

11	13	17	19	2	3	5	7	23	29	31	37	41	47	53	59

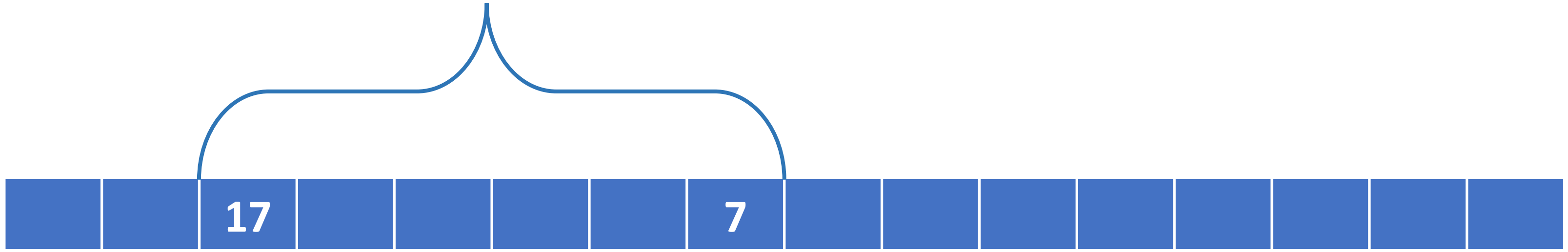
The main idea: Unrepairability

$\mathcal{P}$: a 2-local property of 1D arrays $A: [n] \to \Sigma$.

An interval $I = \{a, a+1, \dots, b\} \subseteq [n]$ is **unrepairable** (w.r.t A, $\mathcal{P}$) if, no matter how we modify $A(a+1), \dots, A(b-1)$, the sub-array of A between a and b will **never** satisfy $\mathcal{P}$.

Observation: Enough to query only $f(a)$ and $f(b)$ to know if I is unrepairable.

Example: unrepairable interval for **monotonicity**.



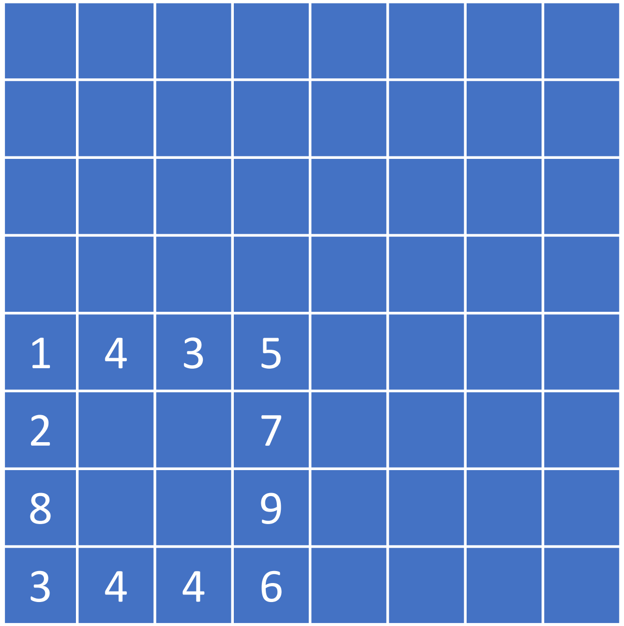
The main idea: Unrepairability

$\mathcal{P}$: a 2-local property of 1D arrays $A: [n] \to \Sigma$.

Proof idea:

Structural result: Suppose that A is ε-far from $\mathcal{P}$. Then there is a set of "canonical" unrepairable intervals covering $\geq \varepsilon n$ of the entries.

Algorithm: For any $i = 0,1,\ldots,\log n$, pick $\approx 1/\epsilon$ "canonical" intervals of length $\approx 2^i$ and query their endpoints.
With good probability, one of the intervals will be unrepairable.



Extension to multiple dimensions:
Replace "intervals" by "d-dimensional consecutive boxes" and "endpoints" with "$(d-1)$-dimensional boundaries".

Non-adaptive Lower bounds

The upper bound is tight for non-adaptive algorithms, for any fixed $d \geq 1$

For $\boldsymbol{d = 1}$, matches $\Theta(\log n)$ bounds for monotonicity [EKKRV'98, F'04, CS'13], convexity [PRR'04], and Lipschitz [JR'11]. Tight for monotonicity even among adaptive two-sided tests.

For $\boldsymbol{d > 1}$,

Theorem [B., 2019]:

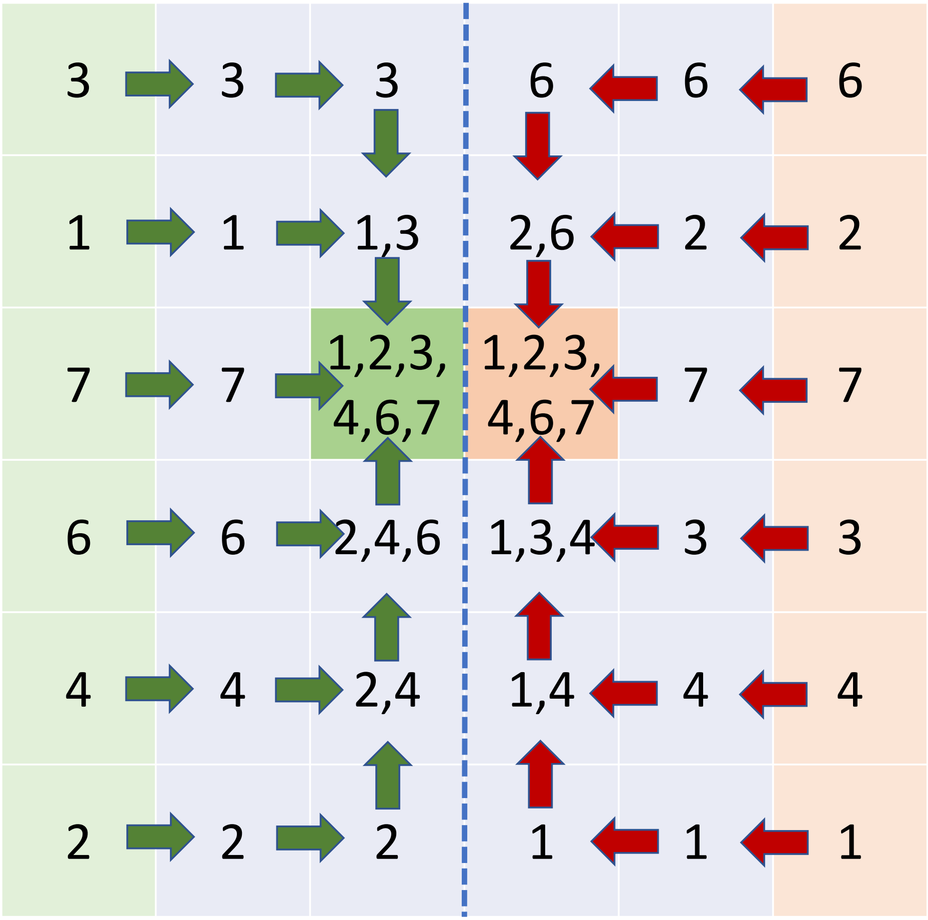
There exists a k-local property of $[n]^d$-arrays over alphabet of size $n^{O(d)}$, whose non-adaptive one-sided query complexity is $\Omega_d(k\varepsilon^{-\frac{1}{d}}n^{d-1})$.

Non-adaptive Lower bounds

The upper bound … or any fixed $d \geq 1$

For $\boldsymbol{d = 1}$, matches $\Theta(\log n)$ … 04, CS'13], convexity
[PRR'04], and Lipschitz [JR … daptive two-sided tests.

For $\boldsymbol{d > 1}$,



Theorem [B., 2019]:

There exists a k-loc … alphabet of size
$n^{O(d)}$, whose non-ad … ity is $\Omega_d(k\varepsilon^{-\frac{1}{d}}n^{d-1})$.

Adaptive Lower bounds

What about the adaptive case for $\boldsymbol{d > 1}$?

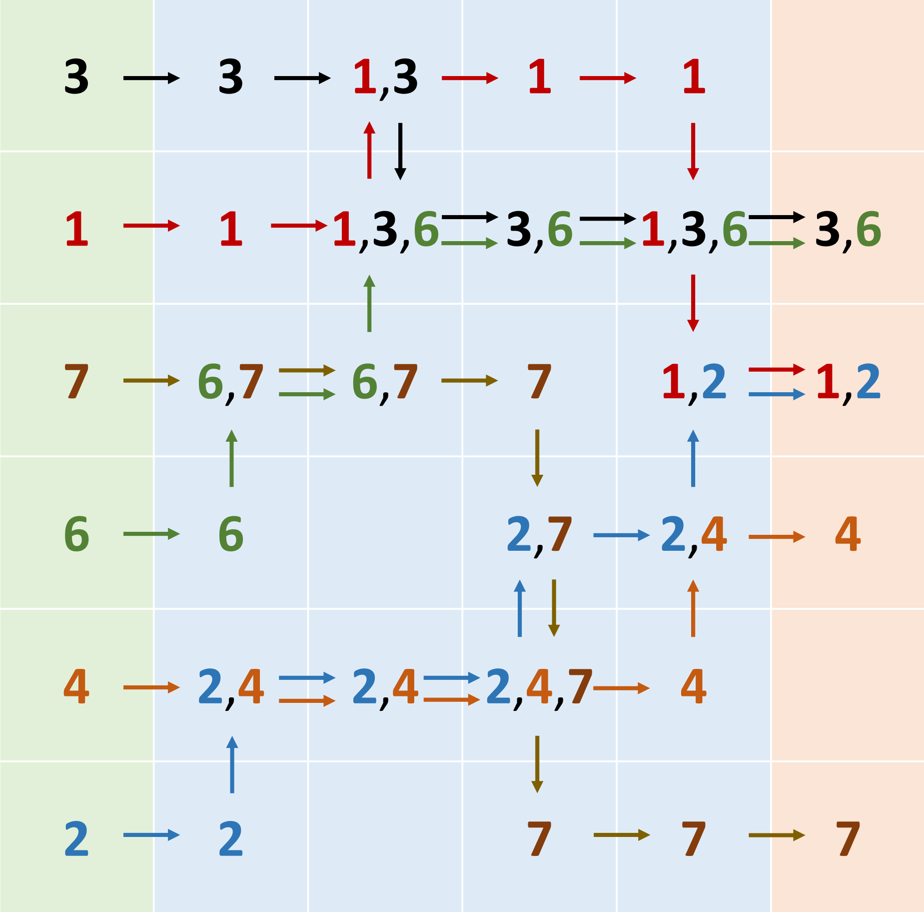
Theorem [B., 2019+]:

There exists a 2-local property of $[n]^d$-arrays, whose adaptive two-sided query complexity is $n^{\Omega(1)}$.

Open question: close the gaps – no known lower bounds depending on d.

Adaptive Lower bounds

What about the adaptive



Theorem [B., 2019+]:

There exists a 2-loc ose adaptive two-
sided query complexi

Open question: close the ending on d.

Questions

1. Exponential running time is undesirable.
   [S. Raskhodnikova, C. Seshadhri:] For which subclasses of local properties can we also get sublinear running time?
   [Chakrabarty, Seshadhri '12]: "bounded derivative" properties.

2. On which graph does "locality $\Longrightarrow$ sublinear testability" hold?
   Bounded-degree graphs? Hyperfinite graphs?

3. How powerful is adaptivity?

Thank you!