Monotonicity Testing

Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron and Alex Samorodnitsky
Probabilistic Property Testing

Probabilistic Algorithm

YES

Always accept.

NO

Reject with probability $\geq 1/2$. 
Probabilistic Property Testing

Probabilistic Algorithm

YES

Always accept.

NO

Reject with probability $\geq 1/2$.

Probabilistic Property Tester

YES

Always accept.

Close to YES

Don’t care

Far from YES

Reject with probability $\geq 1/2$. 
Probabilistic Property Tester

Algorithm

Accept if the object has the property.
Reject with Pr ≥ 1/2
if the object is far from any object with the property.
Probabilistic Property Tester

Algorithm

Accept if the object has the property.

Reject with $\Pr \geq 1/2$

if the object is far from any object with the property.
Probabilistic Property Tester

Algorithm

Accept if the object has the property.
Reject with \( \Pr \geq 1/2 \) if the object is far from any object with the property.
**Probabilistic Property Tester**

Express property testing as testing properties of functions.

Algorithm

- **Accept** if the object has the property.
- **Reject** with \(Pr \geq 1/2\)

if the object is far from any object with the property.
Motivation

*Probabilistic Property Tester* can be

- much faster than an exact algorithm;
- the only option when the exact problem is not decidable;
- used for preprocessing;
- good enough in application where some errors are tolerable.
Problem Statement

\[ \text{Algorithm} \]

\[ \varepsilon \]

Accept if \( f \) is monotone.

Reject with \( \Pr \geq 1/2 \) if \( f \) is \( \varepsilon \)-far from monotone.
Definitions for $f : \Sigma^n \to R$

- For two $n$-symbol strings $x$ and $y$ we say $x < y$ if $y$ is formed from $x$ by increasing one or more symbols.

Examples:

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>001001</td>
<td>011001</td>
<td>011111</td>
<td></td>
<td>100000</td>
</tr>
</tbody>
</table>


Definitions for $f: \Sigma^n \rightarrow R$

- For two $n$-symbol strings $x$ and $y$ we say $x \prec y$ if $y$ is formed from $x$ by increasing one or more symbols.

- $f$ is monotone if $f(x) \leq f(y)$ for all $x \prec y$.

Examples:

<table>
<thead>
<tr>
<th>x</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>x</th>
<th>001001</th>
<th>011001</th>
<th>011111</th>
<th>?</th>
<th>100000</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Definitions for $f : \Sigma^n \rightarrow R$

- For two $n$-symbol strings $x$ and $y$ we say $x \prec y$ if $y$ is formed from $x$ by increasing one or more symbols.

- $f$ is monotone if $f(x) \leq f(y)$ for all $x \prec y$.

- $f$ is $\varepsilon$-far from monotone if every monotone function disagrees with $f$ on at least an $\varepsilon$-fraction of the domain.

Examples:

<table>
<thead>
<tr>
<th>$x$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>0</td>
<td>3</td>
<td>7</td>
<td>8</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>0</td>
<td>3</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

1/3-far from monotone
**Results**

$Q = \text{Query Complexity of Monotonicity Tests}$

$Q = O\left(\frac{n^2}{\varepsilon} \cdot |\Sigma|^2 \cdot |R|\right)$

This work

$Q = O\left(\frac{n}{\varepsilon} \cdot \log |\Sigma| \cdot \log |R|\right)$
Algorithm (Reduction to a simpler case)

- **INPUT:**
  - $\epsilon$ and $f : \Sigma^n \rightarrow R$

- Repeat several times:
  - Pick a **line** along the axes of the hyper-grid uniformly at random.
  - Use your favorite algorithm to test if the **line** is monotone [our paper, EKKRV98, Noga Alon].
  - If a pair $(x, y)$ of points on the **line** with $x \prec y$ and $f(x) > f(y)$ is found, then REJECT.

- Otherwise, ACCEPT.
Special case: \( f : \{0,1\}^n \rightarrow R \)

• Edge \( x \rightarrow y \) iff \( x < y \) and 
  \( x \) and \( y \) differ in one coordinate

• Edge \( x \rightarrow y \) is a violated edge of \( f \) 
  if \( f(x) > f(y) \).
Algorithm for $f : \{0,1\}^n \mapsto R$

- **INPUT:**
  - $\varepsilon$ and $f : \{0,1\}^n \mapsto R$

- Repeat $Q/2$ times:
  - Pick an edge $x \rightarrow y$ uniformly at random.
  - If $x \rightarrow y$ is violated (i.e. $f(x) > f(y)$), then REJECT.

- Otherwise, ACCEPT.
Intuition for Analysis

• If $f$ is monotone, the algorithm always accepts.

• If $f$ is not monotone:
  – If $f$ has few violated edges, we can make $f$ monotone by changing its value at a few points.
  – If $f$ has many violated edges, the algorithm succeeds with high probability.
Proof Plan

• BINARY RANGE \((f : \{0,1\}^n \mapsto \{0,1\})\)

# altered points \(\leq 2 \cdot \# \text{ violated edges}\)

THEOREM: If \(f\) is \(\varepsilon\)-far from monotone, then a random edge is violated with probability

\[
\frac{\# \text{ violated edges}}{n 2^n} \geq \frac{\# \text{ altered points}}{2 \cdot n 2^n} \geq \frac{\varepsilon 2^n}{2 \cdot n 2^n} \geq \frac{\varepsilon}{2n}.
\]
Proof Plan

- **BINARY RANGE** \(( f : \{0,1\}^n \mapsto \{0,1\})\)
  
  \# altered points \(\leq 2 \cdot \# \text{ violated edges}\)

  **THEOREM:** If \( f \) is \(\varepsilon\)-far from monotone, then a random edge is violated with probability

  \[
  \frac{\# \text{ violated edges}}{n 2^n} \geq \frac{\# \text{ altered points}}{2 \cdot n 2^n} \geq \frac{\varepsilon 2^n}{2 \cdot n 2^n} \geq \frac{\varepsilon}{2n}.
  \]

- **RANGE REDUCTION** \(( f : \{0,1\}^n \mapsto R)\)

  \# altered points \(\leq 2 \cdot \# \text{ violated edges} \cdot \log |R|\)

  **THEOREM:** If \( f \) is \(\varepsilon\)-far from monotone, then a random edge is violated with probability

  \[
  \frac{\# \text{ violated edges}}{n 2^n} \geq \frac{\# \text{ altered points}}{n 2^n \cdot 2 \log |R|} \geq \frac{\varepsilon 2^n}{n 2^n \cdot 2 \log |R|} \geq \frac{\varepsilon}{2n \log |R|}.
  \]
Swap violated edges $1 \rightarrow 0$ in red dimension to $0 \rightarrow 1$. Swapping red dimension.
Swap violated edges $1 \rightarrow 0$ in red dimension to $0 \rightarrow 1$. 

Sort
Swap violated edges $1 \rightarrow 0$ in red dimension to $0 \rightarrow 1$. 

Swapping red dimension.
LEMMA. Swapping violated edges in dimension $i$
1. repairs all violated edges in dimension $i$;
2. does not increase the number of violated edges in dimension $j$, for all $j \neq i$. 

Swap violated edges $1 \rightarrow 0$ in red dimension to $0 \rightarrow 1$. 

Swapping red dimension.
Repairing Violated Edges in One Dimension

Swap violated edges \( l \rightarrow 0 \) in red dimension to \( 0 \rightarrow l \).

LEMMA. Swapping violated edges in dimension \( i \)
1. repairs all violated edges in dimension \( i \);
2. does not increase the number of violated edges in dimension \( j \), for all \( j \neq i \).
Back to the Proof Plan

• BINARY RANGE \((f : \{0,1\}^n \mapsto \{0,1\})\)
  
  \[\text{# altered points} \leq 2 \cdot \# \text{ violated edges}\]

• RANGE REDUCTION \((f : \{0,1\}^n \mapsto R)\)
  
  \[\text{# altered points} \leq 2 \cdot \# \text{ violated edges} \cdot \log |R|\]

W.l.g. assume \(R=\{0, 1, \ldots, 2^s-1\}\).

Prove \(#\text{ altered points} \leq 2 \cdot \# \text{ violated edges} \cdot s\) by induction on \(s\).
How can we make $f$ monotone?

Swap violated edges in red dimension?
How can we make $f$ monotone?

violated edges of $f$

Swap violated edges in red dimension?
Operator SQUASH

\[ SQUASH[f, a, b](x) = \begin{cases} 
  a & \text{if } f(x) \leq a \\
  b & \text{if } f(x) \geq b \\
  f(x) & \text{otherwise}
\end{cases} \]

PROPERTY: does not introduce new violated edges.
**Operator CLEAR**

**Squash:**

**Repair** (switch to closest monotone function with range \{a, ..., b\}): 

**Unsquash** values not altered by repair:

**PROPERTIES:**
1. does not introduce new violated edges.
2. **clears** interval \([a, b]\).
3. leaves clear intervals clear.
Making $f$ monotone

$\downarrow$

Clear $[2^{s-1}-1, 2^s-1]$  

$\downarrow$

Clear $[0, 2^s-1]$  

$\downarrow$

Clear $[2^{s-1}, 2^{s-1}-1]$
Making $f$ monotone

$\begin{align*}
&f \\
&\downarrow \\
&\text{Clear } [2^{s-1}-1, 2^s-1] \\
&\downarrow \\
&\text{Clear } [0, 2^{s-1}-1] \\
&\downarrow \\
&\text{Clear } [2^{s-1}, 2^s-1-1]
\end{align*}$

- $2v$
- $2(l + r) \leq 2v$
- $2r$
- $0$

#violated edges
We are proving (by induction on $S$) that for functions with a range of size $2^S$,

\[
\text{# altered points} \leq 2 \cdot \text{# violated edges} \cdot S.
\]

Base case [functions with a range of size 2]:

\[
\text{# altered points} \leq 2 \cdot \text{# violated edges}.
\]

Induction hypothesis [functions with a range of size $2^{S-1}$]:

\[
\text{# altered points} \leq 2 \cdot \text{# violated edges} \cdot (S - 1).
\]
Making $f$ monotone

$\begin{array}{c}
\text{Clear } [2^{s-1}-1, 2^{s-1}] \\
\text{Clear } [0, 2^{s-1}-1] \\
\text{Clear } [2^{s-1}, 2^{s-1}-1]
\end{array}$

$\begin{array}{c}
\text{#violated} \\
\text{edges} \\
\text{points} \\
\text{changed}
\end{array}$

$\begin{array}{c}
2v \\
\leq 2v \\
2r \\
\leq 2l(s-1) \\
0 \\
\leq 2r(s-1) \\
\text{total} \\
\leq 2v \cdot s
\end{array}$
We are done

- **BINARY RANGE** \((f : \{ 0,1 \}^n \mapsto \{ 0,1 \})\)

  \# altered points \(\leq 2 \cdot \# \text{ violated edges} \) 

- **RANGE REDUCTION** \((f : \{ 0,1 \}^n \mapsto R)\)

  \# altered points \(\leq 2 \cdot \# \text{ violated edges} \cdot \log |R| \)

W.l.g. assume \(R=\{0, 1,\ldots, 2^s-1 \}\).

Proof by induction on \(s\).
Conclusions

- **SUMMARY OF RESULTS IN THIS TALK**
  - Monotonicity test for $f : \{0,1\}^n \mapsto \{0,1\}$ [switching argument].
  - Monotonicity test for $f : \{0,1\}^n \mapsto R$ [SQUASH and CLEAR argument].

- **SUMMARY OF RESULTS IN THE PAPER**
  - Designed good monotonicity tests for $f : \Sigma \mapsto \{0,1\}$.
  - Reduced testing monotonicity of $f : \Sigma^n \mapsto \{0,1\}$ to the case $n = 1$ [sorting argument (a generalization of the switching argument)].
  - Reduced testing monotonicity of $f : \Sigma^n \mapsto R$ to the case $f : \Sigma^n \mapsto \{0,1\}$ [SQUASH and CLEAR argument].

- **OPEN PROBLEMS**
  - Query complexity independent of the size of the range?
  - $f : D \mapsto R$, where $D$ is any partially ordered set.
  - Tests for other properties.