THE RISE AND FALL OF BOOLEAN FUNCTIONS

Testing k-Monotonicity

Clément Canonne
Joint work with Elena Grigorescu, Siyao Guo, Akash Kumar, and Karl Wimmer.

May 16, 2016

Columbia University

“PROPERTY TESTING?”

WHY?

Property testing of Boolean functions:

WHY?

Property testing of Boolean functions: sublinear, Property testing of Boolean functions: sublinear, approximate,

WHY?

Property testing of Boolean functions: sublinear, approximate, randomized

Property testing of Boolean functions: sublinear, approximate, randomized algorithms that make membership queries

WHY?

Property testing of Boolean functions: sublinear, approximate, randomized algorithms that make membership queries

- Big Dataset: too big

WHY?

Property testing of Boolean functions: sublinear, approximate, randomized algorithms that make membership queries

- Big Dataset: too big
- Expensive access: pricey data

WHY?

Property testing of Boolean functions: sublinear, approximate, randomized algorithms that make membership queries

- Big Dataset: too big
- Expensive access: pricey data
- Preliminary check: before deciding, learning, reconstructing...

WHY?

Property testing of Boolean functions: sublinear, approximate, randomized algorithms that make membership queries

- Big Dataset: too big
- Expensive access: pricey data
- Preliminary check: before deciding, learning, reconstructing...

Need to infer information – one bit – from the data: fast, or with very few queries.

HOW?

Property Testing:

Property Testing:

Property Testing:

in an (egg)shell.

HOW?

Known domain (here $[n]^d = \{1, \ldots, n\}^d$) and range (here $\{0, 1\}$)

Property (or class) $\mathcal{C} \subseteq \{0, 1\}^{[n]^d}$

Blackbox access to unknown $f \in \{0, 1\}^{[n]^d}$

Distance parameter $\varepsilon \in (0, 1]$

HOW?

Known domain (here $[n]^d = \{1, \ldots, n\}^d$) and range (here $\{0, 1\}$)

Property (or class) $\mathcal{C} \subseteq \{0, 1\}^{[n]^d}$

Blackbox access to unknown $f \in \{0, 1\}^{[n]^d}$

Distance parameter $\varepsilon \in (0, 1]$

Must decide:

$$f \in \mathcal{C}$$

HOW?

Known domain (here $[n]^d = \{1, \ldots, n\}^d$) and range (here $\{0, 1\}$)

Property (or class) $\mathcal{C} \subseteq \{0, 1\}^{[n]^d}$

Blackbox access to unknown $f \in \{0, 1\}^{[n]^d}$

Distance parameter $\varepsilon \in (0, 1]$

Must decide:

$$f \in \mathcal{C}, \text{ or } d(f, \mathcal{C}) > \varepsilon?$$

HOW?

Known domain (here $[n]^d = \{1, \ldots, n\}^d$) and range (here $\{0, 1\}$)

Property (or class) $\mathcal{C} \subseteq \{0, 1\}^{[n]^d}$

Blackbox access to unknown $f \in \{0, 1\}^{[n]^d}$

Distance parameter $\varepsilon \in (0, 1]$

Must decide:

$$f \in \mathcal{C}, \text{ or } d(f, \mathcal{C}) > \varepsilon?$$

(and be correct on any f with probability at least 2/3)

WHAT?

one-sided vs. two-sided

adaptive vs. non-adaptive

regular vs. tolerant

For circuit complexity theorists:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is monotone if it is computed by a Boolean circuit with no negations (only AND and OR gates).

For circuit complexity theorists:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is monotone if it is computed by a Boolean circuit with no negations (only AND and OR gates).

For analysis of Boolean functions enthusiasts:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is monotone if for any $x \preceq y$ in $\{0,1\}^d$, $f(x) \leq f(y)$.

For circuit complexity theorists:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is monotone if it is computed by a Boolean circuit with no negations (only AND and OR gates).

For analysis of Boolean functions enthusiasts:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is monotone if for any $x \preceq y$ in $\{0,1\}^d$, $f(x) \leq f(y)$.

For people with a twisted mind:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is monotone if $f(0^n) \leq f(1^n)$ and f changes value at most once on any ascending chain.

(These definitions are equivalent.)

Examples.

The majority function (1 iff at least half the votes are positive): more votes cannot make a candidate lose.

The s-clique function (1 iff the input graph contains a clique of size s): more edges cannot remove a clique.

The dictator function (1 iff $x_1 = 1$): more voters have no influence anyway.

Can we learn them?

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(\sqrt{d}/\varepsilon)}$. [BT96]

Can we learn them?

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(\sqrt{d}/\varepsilon)}$. [BT96]

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions with membership queries (to error ε) requires $2^{\Omega(\sqrt{d}/\varepsilon)}$ queries. [BT96, BCO+15]

Can we learn them?

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(\sqrt{d}/\varepsilon)}$. [BT96]

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions with membership queries (to error ε) requires $2^{\Omega(\sqrt{d}/\varepsilon)}$ queries. [BT96, BCO+15]

Can we learn them?

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(\sqrt{d}/\varepsilon)}$. [BT96]

Theorem

Learning the class $\mathcal{M}$ of monotone Boolean functions with membership queries (to error ε) requires $2^{\Omega(\sqrt{d}/\varepsilon)}$ queries. [BT96, BCO+15]

So...

Can we test them?

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions can be done with $\tilde{O}(\sqrt{d}/\varepsilon)$, non-adaptively, with one-sided error. [GGL+00, KMS15]

Can we test them?

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions can be done with $\tilde{O}(\sqrt{d}/\varepsilon)$, non-adaptively, with one-sided error. [GGL^{+}00, KMS15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(\sqrt{d})$ queries*, non-adaptively, with two-sided error. [CST14, CDST15]

Can we test them?

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions can be done with $\tilde{O}(\sqrt{d}/\varepsilon)$, non-adaptively, with one-sided error. [GGL+00, KMS15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(\sqrt{d})$ queries*, non-adaptively, with two-sided error. [CST14, CDST15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(d^{1/4})$ queries, adaptively, with two-sided error. [BB15]

Can we test them?

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions can be done with $\tilde{O}(\sqrt{d}/\varepsilon)$, non-adaptively, with one-sided error. [GGL⁺00, KMS15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(\sqrt{d})$ queries*, non-adaptively, with two-sided error. [CST14, CDST15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(d^{1/4})$ queries, adaptively, with two-sided error. [BB15]

Can we test them?

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions can be done with $\tilde{O}(\sqrt{d}/\varepsilon)$, non-adaptively, with one-sided error. [GGL+00, KMS15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(\sqrt{d})$ queries*, non-adaptively, with two-sided error. [CST14, CDST15]

Theorem

Testing the class $\mathcal{M}$ of monotone Boolean functions requires $\Omega(d^{1/4})$ queries, adaptively, with two-sided error. [BB15]

So...

Also...

Many results on testing monotonicity over different domains, ranges [DGL$^+$99, FR10, CS13, FLN$^+$02], or in different distances [BRY14].

So...

Let's forget about monotonicity.

For circuit complexity theorists:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ has inversion complexity t if it can be computed by a Boolean circuit with t negations (besides AND and OR gates), but no less.

For circuit complexity theorists:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ has inversion complexity t if it can be computed by a Boolean circuit with t negations (besides AND and OR gates), but no less.

For people with a twisted mind:

Definition

A Boolean function $f\colon \{0,1\}^d \to \{0,1\}$ is k-alternating if f changes value at most k times on any increasing chain from 0^n to 1^n.

(Analysis of Boolean functions enthusiasts, don't go yet?)

(These definitions are equivalent [Mar57].)

Examples.

The "not-too-many" function (1 iff between 40% and 60% of the votes are positive): more votes can harm a candidate.

The s-clique-but-no-Hamiltonian function (1 iff the input graph contains a clique of size s, but no Hamiltonian cycle): more edges can make things worse.

The Highlander function (1 iff exactly one of the x_i's is 1): there shall be only one.

Why should we care?

Circuit lower bounds [Mar57, Juk12],

Why should we care?

Circuit lower bounds [Mar57, Juk12], learning theory [BCO+15],

Why should we care?

Circuit lower bounds [Mar57, Juk12], learning theory [BCO+15], cryptography [GMOR15], ...

Can we learn them?

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(k\sqrt{d}/\varepsilon)}$. [BCO+15]

Can we learn them?

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(k\sqrt{d}/\varepsilon)}$. [BCO+15]

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions with membership queries (to error ε) requires $2^{\Omega(k\sqrt{d}/\varepsilon)}$ queries. [BCO+15]

Can we learn them?

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(k\sqrt{d}/\varepsilon)}$. [BCO+15]

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions with membership queries (to error ε) requires $2^{\Omega(k\sqrt{d}/\varepsilon)}$ queries. [BCO+15]

Can we learn them?

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions from uniform examples (to error ε) can be done in time $2^{\tilde{O}(k\sqrt{d}/\varepsilon)}$. [BCO+15]

Theorem

Learning the class $\mathcal{M}_k$ of k-monotone Boolean functions with membership queries (to error ε) requires $2^{\Omega(k\sqrt{d}/\varepsilon)}$ queries. [BCO+15]

But can we test?

OUR RESULTS

	General k	$k = 2$	$k = 1$ (monotonicity)
$d = 1$	$\Theta(\frac{k}{\varepsilon})$ 1.s.-n.a., $\tilde{O}(\frac{1}{\varepsilon^7})$ 2.s.-n.a.	$O(\frac{1}{\varepsilon})$ 1.s.-n.a.	$\Theta(\frac{1}{\varepsilon})$ 1.s-n.a.
$d = 2$	$\tilde{O}(\frac{k^2}{\varepsilon^3})$ 2.s.-n.a. (from below)	$\Theta(\frac{1}{\varepsilon})$ 2.s.-a.	$\Theta(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon})$ 1.s-n.a., $\Theta(\frac{1}{\varepsilon})$ 1.s-a.
$d \geq 3$	$\tilde{O}(\frac{1}{\varepsilon^2}\left(\frac{5kd}{\varepsilon}\right)^d)$ 2.s.-n.a., $2^{\tilde{O}(k\sqrt{d}/\varepsilon^2)}$ 2.s.-n.a.	$\tilde{O}(\frac{1}{\varepsilon^2}\left(\frac{10d}{\varepsilon}\right)^d)$ 2.s.-n.a. $2^{\tilde{O}(\sqrt{d}/\varepsilon^2)}$ 2.s.-n.a.	$O(\frac{d}{\varepsilon}\log\frac{d}{\varepsilon})$ 1.s-n.a.

Table: Testing k-monotonicity of $f\colon [n]^d \to \{0,1\}$. (Last column: not us.)

OUR RESULTS

	General k	$k = 2$	$k = 1$ (monotonicity)
$d = 1$	$\Theta(\frac{k}{\varepsilon})$ 1.s.-n.a., $\tilde{O}(\frac{1}{\varepsilon^7})$ 2.s.-n.a.	$O(\frac{1}{\varepsilon})$ 1.s.-n.a.	$\Theta(\frac{1}{\varepsilon})$ 1.s-n.a.
$d = 2$	$\tilde{O}(\frac{k^2}{\varepsilon^3})$ 2.s.-n.a. (from below)	$\Theta(\frac{1}{\varepsilon})$ 2.s.-a.	$\Theta(\frac{1}{\varepsilon}\log\frac{1}{\varepsilon})$ 1.s-n.a., $\Theta(\frac{1}{\varepsilon})$ 1.s-a.
$d \geq 3$	$\tilde{O}(\frac{1}{\varepsilon^2}\left(\frac{5kd}{\varepsilon}\right)^d)$ 2.s.-n.a., $2^{\tilde{O}(k\sqrt{d}/\varepsilon^2)}$ 2.s.-n.a.	$\tilde{O}(\frac{1}{\varepsilon^2}\left(\frac{10d}{\varepsilon}\right)^d)$ 2.s.-n.a. $2^{\tilde{O}(\sqrt{d}/\varepsilon^2)}$ 2.s.-n.a.	$O(\frac{d}{\varepsilon}\log\frac{d}{\varepsilon})$ 1.s-n.a.

Table: Testing k-monotonicity of $f\colon [n]^d \to \{0, 1\}$. (Last column: not us.)

Sorry...

... about your eyes.

REST OF THE TALK

OUTLINE

1. The case of the line $[n]$: “Two-sidedness, magic, and k”
2. The case of the grid $[n]^2$: “L_1 Testing: an unexpected journey”
3. The case of the hypergrid $[n]^d$: “There be Fourier.”
4. Discussion: the hypercube $[2]^d$.

Theorem

There exists a one-sided non-adaptive tester for k-monotonicity of $f\colon [n] \to \{0,1\}$ with query complexity $O(\frac{k}{\varepsilon})$.

… and this is tight.*

Proof.

1. Blocks of size $\frac{\varepsilon n}{k}$: block coarsening $g\colon [n] \to \{0,1\}$ of f.
   - f k-monotone $\rightsquigarrow$ g k-monotone and close to f
   - f far from it $\rightsquigarrow$ g (i) far from k-monotone, or (ii) far from f
   - g "simple."
2. Learn g: cheap.
3. Test if g is close to f: very cheap...

$\square$

Proof.

1. Blocks of size $\frac{\varepsilon n}{k}$: block coarsening $g\colon [n] \to \{0,1\}$ of f.
    - f k-monotone $\rightsquigarrow$ g k-monotone and close to f
    - f far from it $\rightsquigarrow$ g (i) far from k-monotone, or (ii) far from f
    - g "simple."
2. Learn g: cheap.
3. Test if g is close to f: very cheap...

$\square$

But two-sided.

Need another small trick to get one-sidedness.

Theorem

There exists a two-sided non-adaptive tester for k-monotonicity of $f\colon [n] \to \{0,1\}$ with query complexity $O(\frac{1}{\varepsilon^7})$, independent of k.

... and did not see that coming.

Proof.

1. Blocks of size $\frac{\varepsilon n}{k}$: block coarsening $g \colon [K] \to \{0,1\}$ of f.
2. Function $g \rightsquigarrow$ distribution D_g over $[s]$ (blocks), where "$s = \mathrm{kmonotonicity}(f)$"
3. Have sample access to D_g and query access to its pmf

Idea

Do support size estimation on D_g in the extended access model of [CR14].

$\square$

Proof.

1. Blocks of size $\frac{\varepsilon n}{k}$: block coarsening $g\colon [K] \to \{0,1\}$ of f.
2. Function $g \rightsquigarrow$ distribution D_g over [s] (blocks), where "$s = \mathrm{kmonotonicity}(f)$"
3. Have sample access to D_g and query access to its pmf

Idea

Do support size estimation on D_g in the extended access model of [CR14].

$\square$

Challenges

- Do not have (efficient) sample access to D_g
- Do not have (efficient) query access to its pmf
- Support size estimation is a promise problem

Solutions

- Have (efficient) (sample + query) access to D_g+(capped) pmf
- The block coarsening is helping us keeping the promise
- Getting our hands (a bit) dirty.

Theorem ($k = 2$)

There exists a two-sided adaptive tester for 2-monotonicity of $f\colon [n]^2 \to \{0,1\}$ with query complexity $O(\frac{1}{\varepsilon})$.

Proof.

1. Blocks of size $\frac{\varepsilon n}{k}$: block coarsening $g\colon [n] \times [K] \to \{0,1\}$ of f. ("as usual")
2. Assume g is 2-column-wise monotone.
   - Can actually provide access to a "fixed" version of g that is.
   - This helps: reduces the problem to tolerant monotonicity testing in L_1 distance of two functions $f\colon [n] \to [0,1]$
   - … and we know how to do that*: [BRY14]

$\square$

"Overall"

$f\colon [n]^2 \to \{0,1\} \rightsquigarrow g\colon [n]\times[K] \to \{0,1\} \rightsquigarrow \tilde{g}\colon [n]\times[K] \to \{0,1\}$ with $\tilde{g}$ 2-column-wise monotone, and

- f 2-monotone $\rightsquigarrow \tilde{g}$ 2-monotone and close to f;
- f far from it $\rightsquigarrow \tilde{g}$ either far from 2-monotone or far from f; if the former, then L_1 testing will find out.

Challenges

- Provide efficient access to $\tilde{g}$;
- This does not work.*

  *(namely, only leads to $O(1/\varepsilon^2)$ query complexity. Damn.)

Solution

Add yet another layer: $f \rightsquigarrow g \rightsquigarrow \tilde{g} \rightsquigarrow \dot{g}$... with $\dot{g}$ defined so that we can amortize the queries.

Theorem

There exists a two-sided non-adaptive tester for k-monotonicity of $f\colon [n]^d \to \{0,1\}$ with query complexity $\min(\tilde{O}(\frac{1}{\varepsilon^2}\left(\frac{5kd}{\varepsilon}\right)^d), 2^{\tilde{O}(k\sqrt{d}/\varepsilon^2)})$.

Proof.

Actually, two different (tolerant) testers.

1. A block-based one:
    - Partition the domain into (hyper)blocks.
    - Learn the block coarsening of f on this partition.
    - Hope for the best.
2. A Fourier-based one:
    - Fourier analysis on $[n]^d$
    - Generalize the influence lemma of [BCO+15]
    - Agnostic learning via [KKMS08]
    - Connection between agnostic learning and tolerant testing.

□

Proof.

Actually, two different (tolerant) testers.

1. A block-based one:
   - Partition the domain into (hyper)blocks.
   - Learn the block coarsening of f on this partition.
   - Hope for the best.
2. A Fourier-based one:
   - Fourier analysis on $[n]^d$
   - Generalize the influence lemma of [BCO+15]
   - Agnostic learning via [KKMS08]
   - Connection between agnostic learning and tolerant testing.

□

Upshot

Who cares about n? (and also... connections!)

A very cube problem.

Can we get a $2^{o_\varepsilon(\sqrt{d})}$ tester for k-monotonicity of $f\colon \{0,1\}^d \to \{0,1\}$?

A very cube problem.

Can we get a $2^{o_\varepsilon(\sqrt{d})}$ tester for k-monotonicity of $f\colon \{0,1\}^d \to \{0,1\}$?

How hard can it be?

Monotonicity is local. k-monotonicity is not.

DISCUSSION: $\{0,1\}^d$

A very cube problem.

Can we get a $2^{o_\varepsilon(\sqrt{d})}$ tester for k-monotonicity of $f\colon \{0,1\}^d \to \{0,1\}$?

How hard can it be?

Monotonicity is local. k-monotonicity is not.

Lower bounds, ideas, and hopes.

Some of each.

CURRENT STATE: $\{0,1\}^d$

Constant k	upper bound	1.s.-n.a. l.b.	2.s.-n.a. l.b.	2.s.-a. l.b.
$k = 1$	$O(\sqrt{d})$ [KMS15]	$\Omega(d^{1/2})$ [FLN+02]	$\Omega(d^{1/2-o(1)})$ [CDST15]	$\Omega(d^{1/4})$ [BB15]
$k \geq 2$	$O(d^{k\sqrt{d}})$ [BCO+15]	$\Omega(d^{k/4})$	$\Omega(d^{1/2-o(1)})$	$\Omega(d^{1/4})$

Table: Testing k-monotonicity of a function $f\colon \{0,1\}^d \to \{0,1\}$

QUESTIONS?

Aleksandrs Belovs and Eric Blais.
A polynomial lower bound for testing monotonicity.
CoRR, abs/1511.05053, 2015.
To appear in STOC'16.

Eric Blais, Clément L. Canonne, Igor Carboni Oliveira, Rocco A. Servedio, and Li-Yang Tan.
Learning circuits with few negations.
In APPROX-RANDOM, volume 40 of LIPIcs, pages 512–527. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

Piotr Berman, Sofya Raskhodnikova, and Grigory Yaroslavtsev.
L_p-testing.
In STOC, pages 164–173. ACM, 2014.

N. Bshouty and C. Tamon.
On the Fourier spectrum of monotone functions.
Journal of the ACM, 43(4):747–770, 1996.

Xi Chen, Anindya De, Rocco A. Servedio, and Li-Yang Tan.
Boolean function monotonicity testing requires (almost) $n^{1/2}$ non-adaptive queries.
In STOC, pages 519–528. ACM, 2015.

Clément L. Canonne and Ronitt Rubinfeld.
Testing probability distributions underlying aggregated data.
In Proceedings of ICALP, pages 283–295, 2014.

Deeparnab Chakrabarty and C. Seshadhri.
Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids.
In STOC, pages 419–428, 2013.

Xi Chen, Rocco A. Servedio, and Li-Yang Tan.
New algorithms and lower bounds for monotonicity testing.
In FOCS, pages 286–295. IEEE Computer Society, 2014.

Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky.
Improved testing algorithms for monotonicity.
In RANDOM-APPROX, volume 1671 of Lecture Notes in Computer Science, pages 97–108. Springer, 1999.

Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky.
Monotonicity testing over general poset domains.
In Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada, pages 474–483, 2002.

Shahar Fattal and Dana Ron.
Approximating the distance to monotonicity in high dimensions.
ACM Trans. Algorithms, 6(3), 2010.

Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky.
Testing monotonicity.
Combinatorica, 20(3):301–337, 2000.

Siyao Guo, Tal Malkin, Igor Carboni Oliveira, and Alon Rosen.
The power of negations in cryptography.
In TCC (1), volume 9014 of Lecture Notes in Computer Science, pages 36–65. Springer, 2015.

Stasys Jukna.
Boolean Function Complexity.
Springer, 2012.

Adam Tauman Kalai, Adam R. Klivans, Yishay Mansour, and Rocco A. Servedio.
Agnostically learning halfspaces.
SIAM J. Comput., 37(6):1777–1805, 2008.

Subhash Khot, Dor Minzer, and Muli Safra.
On monotonicity testing and Boolean isoperimetric type theorems.

In FOCS, pages 52–58. IEEE Computer Society, 2015.

A. A. Markov.
On the inversion complexity of systems of functions.
Doklady Akademii Nauk SSSR, 116:917–919, 1957.
English translation in [Mar58].

A. A. Markov.
On the inversion complexity of a system of functions.
Journal of the ACM, 5(4):331–334, October 1958.