Open Problems Column

Edited By William Gasarch
Univ of MD at College Park

1 This Issues Column!

Luca Trevisan passed away on June 19, 2024 at the age of 52, of cancer. He worked on randomness, approximation, and many other topics in theory. This column consists of open problems by Lance Fortnow, Oded Goldreich, Johan Håstad, Salil Vadhan, and David P. Williamson that Luca was interested in.

Open Problems In Honor of Luca Trevisan
By

Lance Fortnow
Oded Goldreich
Johan Håstad
Salil Vadhan
David P. Williamson

2 ZPP and Promise-ZPP by Lance Fortnow

One of Luca Trevisan's early papers with Alexander Andreev, Andrea Clementi and Jose Rolim [1] showed how to simulate **BPP** using weak random sources. That paper also has a discussion of the result that efficient hitting set generators implied that you could derandomize Monte-Carlo algorithms, or equivalently that **Promise-RP** in **P** implies **Promise-BPP** in **P**. Can we do the same for **ZPP**?

Conjecture 2.1 Promise-ZPP *in* **P** *implies* **Promise-BPP** *in* **P**

Let's formally define this problem. We'll consider probabilistic polynomial-time Turing machines (PPTM) M that have three outputs, "Accept", "Reject" and "Don't Know".

BPP is the class of randomized algorithms that could error in either direction.

Def 2.2 A language L is in **BPP** if there is a PPTM M such that for all x

1. If x is in L then $\Pr(M \text{ accepts}) \geq 2/3$.
2. If x is not in L then $\Pr(M \text{ accepts}) \leq 1/3$.

RP is like **BPP** but when it accepts we are guaranteed x is in L.

Def 2.3 A language L is in **RP** if there is a PPTM M such that for all x

1. If x is in L then $\Pr(M \text{ accepts}) \geq 1/2$.
2. If x is not in L then $\Pr(M \text{ accepts}) = 0$.

The class **ZPP**, which captures Las Vegas randomness, guarantees correctness in both directions but we allow the machine to run in expected polynomial-time. To simplify the exposition we use the following equivalent definition.

Def 2.4 A language L is in **ZPP** if there is a PPTM M such that for all x

1. If x is in L then $\Pr(M \text{ accepts}) \geq 1/2$ and $\Pr(M \text{ rejects}) = 0$.

2. If x is not in L then $\Pr(M \text{ rejects}) \geq 1/2$ and $\Pr(M \text{ accepts}) = 0$.

When M doesn't accept or reject it can say "Don't Know".

One can show **ZPP** = **RP** $\cap$ **coRP** [10].

BPP, **RP** and **ZPP** are semantic classes, M is required to fulfill either condition 1 or condition 2 for all inputs. Promise classes relax that requirement. Defining promise classes directly is tricky so let's just define what it means for a promise class to be in P.

Def 2.5 Promise-BPP in P means that for all PPTM M, there exists a language L in **P** such that for all x

1. If $\Pr(M \text{ accepts}) \geq 2/3$ then x is in L.
2. If $\Pr(M \text{ accepts}) \leq 1/3$ then x is not in L.

If $1/3 < \Pr(M \text{ accepts}) < 2/3$ then it doesn't matter whether or not x is in L.

Promise-RP in **P** and **Promise-ZPP** in **P** are defined similarly.

As we mentioned above **Promise-RP** in **P** implies **Promise-BPP** (see [1]) but there is a relativized world where **P** = **RP** $\neq$ **BPP** [31]. **Promise-(NP** $\cap$ **coNP)** in **P** implies **P** = **NP** via self-reduction [8]. One might hope to use the same idea with **Promise-ZPP** but it's hard to maintain the **ZPP**-promise as you do the self-reduction. On the other hand, there is no known relativized world where **Promise-ZPP** is in **P** but **Promise-BPP** is not.

3 Lower Bounds on the Length of Locally Decodable Codes by Oded Goldreich

One of Luca Trevisan's best known papers is the one in which he established, together with Jonathan Katz (who was advised by Luca at the time), a super-linear lower bound on the length of locally decodable codes [23]. The challenge of significantly improving over this lower bound has remained open till this very day, although several improvements has been established since (cf., e.g., [24, 27]). Although this challenge is well-known, I will outline it in the rest of this note. In particular, I will specify the challenge as separating locally decodable codes from relaxed locally decodable codes.

Background. A locally decodable code (LDC) is a (binary) error correcting code that allows for the recovery of any desired bit in the message based on a constant number of (randomly selected) bits in the possibly corrupted codeword. Locally decodable codes, or rather a family of such codes, have several parameters: The length of the message, denoted k, the length of codewords, denoted n (and viewed as a function of k), the number of queries, denoted q, and the tolerated corruption rate, denoted δ. We shall view $q \in \mathbb{N}$ and $\delta > 0$ as fixed constants, whereas k and $n = n(k)$ are viewed as varying parameters. (This regime is fundamentally different from the one in [26] (and subsequent works), where $n = O(k)$ and q is allowed to be a function of k.)

The conjecture that locally decodable codes require large length (e.g., n must be super-polynomial in k), which was supported by the super-linear lower bound (i.e., $n = \Omega(k^{q/(q-1)})$) of [23], led [2] to suggest a *relaxed notion of LDCs*. In this relaxation, hereafter referred to as relaxed LDCs, the decoder is allowed to announce failure and two conditions are made:

1. When given access to a valid codeword, the local decoder always recovers the desired bit.

2. When given access to a string that is δ-close to a valid codeword (i.e., the relative Hamming distance between the string and the codeword is at most δ), with probability at least 2/3, the local decoder does not err;

that is, with probability at least $2/3$, it outputs either the desired bit or a special failure symbol.

Interestingly, as shown in [2, Sec. 4.2.2] (see [13]), relaxed LDCs of polynomial length exist. Specifically, for every sufficiently large constant q, one can construct a q-query relaxed LDC of length $n = k^{1+O(1/q)}$, where the O-notation hides a universal constant that is independent of q. While these relaxed LDCs are much shorter than the best known (super-polynomial length) constructions of LDCs (of [39, 7]), these relaxed LDCs are not shorter than the known lower bound for LDCs, which are $n = \Omega(k/\log k)^{1+\frac{1}{\lceil q/2\rceil-1}}$ (cf. [24, Thm. 7], improving over [23]). Hence, the known results fail to separate relaxed LDCs from LDCs.

The hope of separating relaxed LDCs from LDCs turned out to be hard to materialize. A formidable difficulty has recently emerged from the work of [16, 6] (see also [12]) which shows that q-query relaxed LDC must have length $n \geq k^{1+\Omega(1/q\log q)^2}$, where the Ω-notation hides a universal constant that is independent of q. Hence, the hope that a separation can be obtained by a construction of a relaxed LDC of length $n = \widetilde{O}(k)$ was shuttered, and the ballpark for improved constructions of relaxed LDCs is (conceptionally) smaller. Still, a q-query relaxed LDC of length $n = k^{1+o(1/q)}$ would yield the desired separation.

The challenge. A seemingly more viable way of separating relaxed LDCs from LDCs is to show that q-query LDC must have length $n \geq k^{1+\omega(1/q)}$, where the ω-notation refers to a super-linear function. Actually, for a universal constant $c > 0$, it suffices to show that, for some sufficiently large q, it holds that q-query LDC must have length $n \geq k^{1+(c/q)}$. (These constants should be set to fit the construction of [2]; that is, enable the construction of a q-query *relaxed* LDC of length $n < k^{1+(c/q)}$.) Recall that we actually conjecture that $O(1)$-query LDC must have length $n \geq k^{\omega(1)}$.

4 An Interesting Graph That Might Be a Small Set Expander by Johan Håstad

Abstract

We define a graph which has many eigenvalues close to one. We hope that this is a small set expander.

4.1 Motivation

Graphs with many (in the best of all worlds n^δ for $\delta > 0$) large (close to one, or at least greater than 1/2) eigenvalues are potential sources for hard instances for unique games and max-cut. We want the graph to be a small set expander to avoid algorithms that split up the graph in to pieces.

This is very much related to Luca's work showing how to relate the top k non-trivial eigenvalues of the adjacency matrix of a graph to the ability to partition the vertices in to k parts with sparse cuts between the parts.

4.2 The Graph

Let C be a binary linear code of minimal distance d. The construction works for any linear code but a likely choice is to make C a random linear code of co-dimension t where $2^t \approx \binom{n}{d}$. Any d in the range $\omega(1)$ to $o(n)$ might be interesting but a good value to think about is $d = n^c$ for $c < 1$.

We have a weighted graph on the hypercube, $\{0,1\}^n$ and we connect x to $x + c + y$ where c is a random vector in $C^\perp$ (the dual code of C) and y is a random vector where each coordinate is one with probability ϵ/d. An alternative description is to have the vertex set $\{0,1\}^n/C^\perp$ and just use the noise vector y but I find it slightly more convenient to talk about the hypercube. We can observe that if C is the entire space then $d = 1$ and c is always 0 and we have the noisy hypercube.

4.3 Eigenvectors and Eigenvalues

As this is a Cayley graph we know that the eigenvectors are the characters

$$\chi_\alpha(x) = (-1)^{\sum_{i\in\alpha} x_i}.$$

Calculating the expected value at a random edge leaving x we get

$$E_{y,c}[\chi_\alpha(x+y+c)] = \chi_\alpha(x)E_y[\chi_\alpha(y)]E_c[\chi_\alpha(c)].$$

We have two not very difficult claims that are left to the reader to verify.
Claim 1 $E_y[\chi_\alpha(y)] = (1-\frac{2\epsilon}{d})^{|\alpha|}$.
Claim 2 $E_c[\chi_\alpha(c)] = 1$ iff $\alpha \in C$ while it is 0 otherwise.

The two claims imply that any $c \in C$ with $|c| \leq kd$ gives an eigenvalue of at least

$$(1-\frac{2\epsilon}{d})^{kd} \geq (1-2k\epsilon).$$

If C is a random code of co-dimension t the expected number of such codewords is $2^{-t}\binom{n}{kd}$ and with the chosen t this is about $\binom{n}{kd}\binom{n}{d}^{-1}$ which for constant k is about

$$\left(\frac{n}{ekd}\right)^{kd}\left(\frac{n}{ed}\right)^{-d}.$$

Ignoring factors of the form k^d is around $(n/d)^{(k-1)d}$. As the number of vertices is $N = 2^n$, for $d = n^c$ this is at least $2^{(\log N)^c}$ which would be more than previous constructions. The open problem now is

Is this graph a small set expander?

A graph is a small set expander we if for any $\delta > 0$ there is a $\gamma > 0$ such that for any set, S, of size at most $\gamma 2^n$ at least a fraction $1-\delta$ of all edges with one endpoint in S has its other end-point outside S.

5 Efficiency of Pseudorandom Generators from One-Way Functions by Salil Vadhan

In his seminal paper with Gennaro, Gertner, and Katz [9], Luca Trevisan studied the efficiency of constructing pseudorandom generators from one-way functions.

Recall that one of the fundamental results in the foundations of cryptography is that pseudorandom generators exist if (and only if) one-way functions exist, as proven by Håstad, Impagliazzo, Levin, and Luby [20]. Unfortunately, this construction is very inefficient. Given a one-way function $f : \{0,1\}^n \to \{0,1\}^n$, Håstad et al. construct a pseudorandom generator

$$G^f : \{0,1\}^{\ell(n)} \to \{0,1\}^{m(n)}$$

whose seed length is $\ell(n)$ for a fairly large polynomial $\ell(n)$ and, when evaluating G^f once, the algorithm G^f makes $q = q(n)$ queries to f for a fairly large polynomial q.

Note that this description also specifies that G^f is a *black-box construction*, given by a polynomial-time algorithm G that uses an arbitrary function $f : \{0,1\}^n \to \{0,1\}^n$ as an oracle. Furthermore, its security is proved by a polynomial-time *black-box reduction*, R, whereby if $D : \{0,1\}^m \to \{0,1\}$ is any function that distinguishes $G(U_\ell)$ from U_m with nonnegligible advantage, then $R^{D,f}$ inverts f with nonnegligible probability. Again R is constrained to use D and f as oracles. If we instantiate such a black-box construction with a polynomial-time computable one-way function, it follows that G^f is a polynomial-time computable pseudorandom generator.

In special cases, such as when f is a one-way *permutation*, there are more efficient constructions. In particular, the classic construction of pseudorandom generators from one-way permutations, due to Blum and Micali [3], Yao [38], and Goldreich and Levin [15] has seed length $\ell(n) = O(n)$ and query complexity $q(n) = \lceil (m(n) - \ell(n))/\log n \rceil$ to achieve output length $m(n)$. In particular, to have a pseudorandom generator that stretches by up to $\log n$ bits (i.e., $m(n) \leq \ell(n) + \log n$) requires only $q(n) = 1$ queries to f, and in general with $q(n)$ queries, we can have a stretch of $q(n) \cdot \log n$.

Gennaro, Gertner, Katz, and Trevisan [9] proved that this construction is the best possible: *every* black-box construction of pseudorandom generators from one-way permutations has stretch of at most $O(q(n) \cdot \log n)$. We remark that their lower bound holds even for a more general notion of black-box reduction than we defined above, where the reduction R only needs to work for distinguishers D that are (nonuniform) polynomial-time algorithms, rather than arbitrary oracles.

However, it remained open to explain the much greater inefficiency of the construction of pseudorandom generators from general one-way functions (rather than one-way permutations). Indeed, the construction of Håstad et al. incurs a large polynomial seed length $\ell(n)$ and query complexity $q(n)$ even to obtain a pseudorandom generator that stretches by one bit (i.e. when $m(n) = \ell(n)+1$). Holenstein and Sinha [22] proved that every black-box construction, even stretching by only one bit, requires $q(n) = \Omega(n/\log n)$ queries. Their lower bound holds even for the special case of *regular* one-way functions, where $|f^{-1}(y)|$ is the same size for all $y \in \{0,1\}^n$. (Crucially, however, this size is unknown to the pseudorandom generator construction.) Their lower bound is tight in this special case, where it matches the query complexity of a pseudorandom generator construction of Goldreich, Krawczyk, and Luby [14]. Furthermore, lower bounds that apply to regular one-way functions cannot explain the large seed length incurred in the case of general one-way functions, since there are constructions of pseudorandom generators from regular one-way functions that have a seed length of $\ell(n) = \tilde{O}(n)$ [18, 40]. Note that, since $q(n) = \Omega(n/\log n)$, such a seed length of $\ell(n) = \tilde{O}(n)$ is not sufficient to specify $q(n)$ independent queries to the function f (which would require seed length $n \cdot q(n)$), so the queries to f are necessarily dependent in such constructions.

For the case of general one-way functions, there has been substantial progress on improving the efficiency of the construction of pseudorandom generators [21, 17, 19, 37, 29]. The current state of the art has a seed length of $\ell(n) = \tilde{O}(n^3)$ and makes $q(n) = \tilde{O}(n^3)$ queries to the one-way function. Thus, we ask:

Open Problem 1: What is the smallest constant c such that there is a (black-box) construction of pseudorandom generators from general one-way functions that has query complexity $q(n) = \tilde{O}(n^c)$?

Open Problem 2: What is the smallest constant c such that there is a (black-box) construction of pseudorandom generators from general one-way functions that has seed length $\ell(n) = \tilde{O}(n^c)$?

In both cases, all we know is that $c \in [1, 3]$ and any shrinking of this interval would be interesting. As discussed in the survey [36], the main source of inefficiency in these constructions, as compared to the case of regular one-way functions, is making $\tilde{O}(n^2)$ independent evaluations of the one-way function in order to convert the conditional Shannon entropy $H(U_n|f(U_n))$ into (smoothed) min-entropy. An information-theoretic analogue of this problem (not involving one-wayness or pseudorandomness) was studied in [5], and in that setting, a query complexity lower bound of $q(n) = \tilde{\Omega}(n^2)$ was proved. Perhaps those ideas can be ported to the computational setting to yield an analogous query complexity lower bound for pseudorandom generator constructions. If, in addition, it can be proved that the queries must be nearly independent, then a seed length lower bound of $\Omega(q(n) \cdot n) = \tilde{\Omega}(n^3)$ would follow. We remark that it was shown in [33] any such lower bound on query complexity or seed length requires working with a more constrained notion of black-box reduction than done in [9].

6 Some open problems based on Luca's work on MAX CUT by David P. Williamson

In 2012, Luca [35] gave an interesting algorithm that used repeated eigenvalue calculation to get a .531-approximation algorithm for the MAX CUT problem. Soto [34] later improved the analysis of the algorithm to a .614-approximation algorithm. One perspective on the algorithm is that it calculates the eigenvector y associated with the smallest eigenvalue of $2I - \mathcal{L}$ (normalized so that $\|y\|_\infty = 1$), where $\mathcal{L}$ is the normalized Laplacian of the graph, picks a random value $t \in (0, 1]$, then puts a vertex i on one side of the cut if $y(i)^2 \geq t$ and $y(i) > 0$, and places i on the other side of the cut if $y(i)^2 \geq t$ and $y(i) < 0$. The algorithm recurses on the vertices for which $y(i)^2 < t$, and joins the resulting sets of vertices with the current ones in whatever way maximizes the size of the cut.

Given that we already know a .878-approximation algorithm for the MAX CUT problem based on semidefinite programming (Goemans & Williamson [11]), why is this algorithm of interest? First, from a pragmatic standpoint, eigenvector computation is easier than solving a semidefinite program. In particular, the algorithm of [11] does not scale well to large graphs because most SDP solvers must maintain a dense $n \times n$ semidefinite matrix, whereas eigenvector computation can be done quickly in $O(n)$ space. Although Luca's algorithm requires repeated eigenvector computation, some experimental work done over the years by my PhD students and some undergrads, finally written up in a paper with PhD student Renee Mirka [30], shows that Luca's algorithm is significantly faster (often by an order of magnitude or more) and nearly as good in terms of quality of solution obtained. Second, Luca's result shows that eigenvector computation outperforms linear programming based algorithms for the MAX CUT problem: a result of Kothari, Meka, and Raghavendra [28] (improving a previous result of Chan, Lee, Rahghavendra, and Steurer [4]) shows that subexponentially sized linear programs cannot have an integrality gap for the MAX CUT problem of more than .5.

There's no particular reason to think that the Soto analysis is tight. This leads to the following question.

Open Question: Find a tight analysis for Luca's algorithm or a variant.

In [30], we observed that simply partitioning the graph based on whether the eigenvalue entry was positive or negative, or taking the best 'sweep cut' (that is, looking at taking the smallest k entries for one side of the cut and the remaining $n-k$ entries for the other side of the cut for each value of $k = 1, \ldots, n-1$) gave computation times that were significantly faster than Luca's algorithm, and, surprisingly, frequently just as good if not better solution quality. It would be interesting to know if there are approximation guarantees better than .5 for these algorithms.

Open Question: Determine whether or not there is an approximation guarantee better than 0.5 for simple spectral partitioning or sweep cuts for the MAX CUT problem.

Finally, we'd like to know the following: in what other circumstances can we use an eigenvector calculation to derive an approximation algorithm that does better than simple random algorithms, or LP integrality gaps? Natural candidates for this question would include other two-variable constraint satisfaction problems (2CSP), including the maximum cut problem in directed graphs (MAX DICUT) and the maximum satisfiability problem in which there are at most (or exactly) two variables per clause (MAX 2SAT or MAX E2SAT). Some of my group did work in this direction. Together with Paul and Poloczek [32], we gave an improvement on the $\frac{3}{4}$-approximation algorithm obtained by simple randomization for MAX E2SAT when such instances are *balanced*; that is, the number of clauses (or the weight of clauses) in which the literal x_i appears is the same as that in which $\bar{x}_i$ appears. In this case, via a reduction to Luca's algorithm, we can obtain a .81-approximation algorithm for the problem (a .943-approximation algorithm using SDP for these instances is known, due to Khot, Kindler, Mossel, and O'Donnell [25]). Furthermore, our work shows that as in the case of MAX CUT, the spectral algorithm outperforms the SDP-based algorithm both in running time and quality of solution. Still, it remains an open question of whether any of this work can be extended to general instances of MAX E2SAT or MAX 2SAT, or even MAX DICUT.

Open Question: Find an approximation algorithms using eigenvector calculations that outperforms the simple randomized algorithms for MAX 2SAT and MAX DICUT.

My former PhD student Alice Paul and I made some preliminary efforts to find an algorithm for MAX 2SAT. The main issue appears to be that for MAX 2SAT and MAX DICUT there are terms in the objective function that can either be viewed as linear or as quadratic with a special variable x_0 that stands for whether 1 or -1 represents TRUE (for DICUT it stands for whether 1 or -1 represents being inside the set S). Luca's algorithm works by setting a subset of the variables, then recurses on the remainder. The issue for MAX 2SAT and MAX DICUT is that we could not figure out what to do in the remaining recursive calls once x_0 was set. Balanced MAX E2SAT instances remove the problem since all terms in the objective function involving x_0 cancel out. One approach to adapting the algorithm for MAX 2SAT and MAX DICUT would be to find a variant of the algorithm that sets all variables at once without recursive calls.

References

[1] A. Andreev, A. Clementi, J. Rolim, and L. Trevisan. Weak random sources, hitting sets, and BPP simulations. *SIAM Journal on Computing*, 28(6):2103–2116, 1999.

[2] E. Ben-Sasson, O. Goldreich, P. Harsha, M. Sudan, and S. P. Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM J. Comput.*, 36(4):889–974, 2006.

[3] M. Blum and S. Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13, 1984.

[4] S. O. Chan, J. Lee, P. Raghavendra, and D. Steurer. Approximate constraint satisfaction requires large LP relaxations. *Journal of the Association of Computing Machinery (JACM)*, 63:Article 34, 2016.

[5] Y.-H. Chen, M. Göös, S. P. Vadhan, and J. Zhang. A tight lower bound for entropy flattening. In *33rd Computational Complexity Conference*, volume 102 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 23, 28. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018.

[6] M. de Sena Dall'Agnol, T. Gur, and O. Lachish. A structural theorem for local algorithms with applications to coding, testing, and privacy. In D. Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1651–1665. SIAM, 2021. `https://doi.org/10.1137/1.9781611976465.100`.

[7] K. Efremenko. 3-query locally decodable codes of subexponential length. *SIAM J. Comput.*, 41(6):1694–1703, 2012. `https://doi.org/10.1137/090772721`.

[8] S. Even, A. Selman, and Y. Yacobi. The complexity of promise problems with applications to public-key cryptography. *Information and Computation*, 61(2):159–173, May 1984.

[9] R. Gennaro, Y. Gertner, J. Katz, and L. Trevisan. Bounds on the efficiency of generic cryptographic constructions. *SIAM Journal on Computing*, 35(1):217–246, 2005.

[10] J. Gill. Computational complexity of probabilistic Turing machines. *SIAM Journal on Computing*, 6:675–695, 1977.

[11] M. X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

[12] O. Goldreich. On the lower bound on the length of relaxed locally decodable codes. *Electron. Colloquium Comput. Complex.*, TR23-064, 2023. `https://eccc.weizmann.ac.il/report/2023/064`.

[13] O. Goldreich. On the relaxed LDC of BGHSV: A survey that corrects the record. *Electron. Colloquium Comput. Complex.*, TR24-078, 2024. `https://eccc.weizmann.ac.il/report/2024/078`.

[14] O. Goldreich, H. Krawczyk, and M. Luby. On the existence of pseudorandom generators. *SIAM Journal on Computing*, 22(6):1163–1175, 1993.

[15] O. Goldreich and L. A. Levin. A hard-core predicate for all one-way functions. In D. S. Johnson, editor, *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washington, USA*, pages 25–32. ACM, 1989.

[16] T. Gur and O. Lachish. On the power of relaxed local decoding algorithms. *SIAM J. Comput.*, 50(2):788–813, 2021. https://doi.org/10.1137/19M1307834.

[17] I. Haitner, D. Harnik, and O. Reingold. Efficient pseudorandom generators from exponentially hard one-way functions. In *Automata, languages and programming. Part II*, volume 4052 of *Lecture Notes in Comput. Sci.*, pages 228–239. Springer, Berlin, 2006.

[18] I. Haitner, D. Harnik, and O. Reingold. On the power of the randomized iterate. *SIAM Journal on Computing*, 40(6):1486–1528, 2011.

[19] I. Haitner, O. Reingold, and S. Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. *SIAM Journal on Computing*, 42(3):1405–1430, 2013.

[20] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[21] T. Holenstein. Pseudorandom generators from one-way functions: a simple construction for any hardness. In *Theory of cryptography*, volume 3876 of *Lecture Notes in Comput. Sci.*, pages 443–461. Springer, Berlin, 2006.

[22] T. Holenstein and M. Sinha. Constructing a pseudorandom generator requires an almost linear number of calls. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science—FOCS 2012*, pages 698–707. IEEE Computer Soc., Los Alamitos, CA, 2012.

[23] J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proceedings of the Thirty-second Annual*

ACM Symposium on the Theory of Computing, Portland OR, New York, 2000. ACM.

[24] I. Kerenidis and R. de Wolf. Exponential lower bound for 2-query locally decodable codes. *Journal of Computer and System Sciences*, pages 395–420, 2004.
`http://arxiv.org/abs/quant-ph/0208062`.

[25] S. Khot, G. Kindler, E. Mossel, and R. O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.
`https://www.cs.cmu.edu/~odonnell/papers/maxcut.pdf`.

[26] S. Kopparty, O. Meir, N. Ron-Zewi, and S. Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *J. ACM*, 64(2):11:1–11:42, 2017.
`https://doi.org/10.1145/3051093`.

[27] P. Kothari and P. Manohar. Superpolynomial lower bounds for smooth 3-lccs and sharp bounds for designs. *Electron. Colloquium Comput. Complex.*, TR24-068, 2024.
`https://eccc.weizmann.ac.il/report/2024/068`.

[28] P. K. Kothari, R. Meka, and P. Raghavendra. Approximating rectangles by juntas and weakly-exponential lower bounds for LP relaxations of CSPs. *SIAM Journal on Computing*, 51, 2021. Special Issue For STOC 2017.

[29] N. Mazor and R. Pass. Counting unpredictable bits: a simple PRG from one-way functions. In *Theory of cryptography. Part I*, volume 14369 of *Lecture Notes in Comput. Sci.*, pages 191–218. Springer, Cham, [2023] ©2023.

[30] R. Mirka and D. P. Williamson. An experimental evaluation of semidefinite programming and spectral algorithms for MAX CUT. *ACM Journal of Experimental Algorithmics*, 28:1–18, 2023. Article 2.1.

[31] A. Muchnik and N. Vereshchagin. A general method to construct oracles realizing given relationships. *Theoretical Computer Science*, 157:227–258, 1996.

[32] A. Paul, M. Poloczek, and D. P. Williamson. Simple approximation algorithms for balanced MAX 2SAT. *Algorithmica*, 80:995–1012, 2018.

[33] O. Reingold, L. Trevisan, and S. Vadhan. Notions of reducibility between cryptographic primitives. In *Theory of cryptography*, volume 2951 of *Lecture Notes in Comput. Sci.*, pages 1–20. Springer, Berlin, 2004.

[34] J. Soto. Improved analysis of a max-cut algorithm based on spectral partitioning. *SIAM Journal on Discrete Mathematics*, 29:259–2682, 2015.

[35] L. Trevisan. Max Cut and the smallest eigenvalue. *SIAM Journal on Computing*, 41:1769–1786, 2012.

[36] S. Vadhan. Computational entropy. In O. Goldreich, editor, *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, pages 693–726. ACM, 2019.

[37] S. Vadhan and C. J. Zheng. Characterizing pseudoentropy and simplifying pseudorandom generator constructions. In *STOC'12—Proceedings of the 2012 ACM Symposium on Theory of Computing*, pages 817–836. ACM, New York, 2012.

[38] A. Yao. Theory and application of trapdoor functions. In *Proceedings of the 23rd Annual IEEE Symposium on Foundations of Computer Science,* Chicago IL, pages 80–91, 1982.

[39] S. Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. ACM*, 55(1):1:1–1:16, 2008.
`https://doi.org/10.1145/1326554.1326555`.

[40] Y. Yu, X. Li, and J. Weng. Pseudorandom generators from regular one-way functions: New constructions with improved parameters. *Theoretical Computer Science*, 569:58–69, 2015.