THEORY OF COMPUTING, Volume 15 (18), 2019, pp. 1–9
www.theoryofcomputing.org

Lower Bounds for Data Structures with Space Close to Maximum Imply Circuit Lower Bounds

Emanuele Viola*

Received February 15, 2019; Revised October 4, 2019; Published December 18, 2019

Abstract: Let $f:\{0,1\}^n \to \{0,1\}^m$ be a function computable by a circuit with unbounded fan-in, arbitrary gates, w wires and depth d. With a very simple argument we show that the m-query problem corresponding to f has data structures with space $s = n + r$ and time $(w/r)^d$, for any r. As a consequence, in the setting where s is close to m a slight improvement on the state of existing data-structure lower bounds would solve long-standing problems in circuit complexity. We also use this connection to obtain a data structure for error-correcting codes which nearly matches the 2007 lower bound by Gál and Miltersen. This data structure can also be made dynamic. Finally we give a problem that requires at least 3 bit probes for $m = n^{O(1)}$ and even $s = m/2 - 1$.

Independent work by Dvir, Golovnev, and Weinstein (2018) and by Corrigan-Gibbs and Kogan (2018) give incomparable connections between data-structure and other types of lower bounds.

ACM Classification: F.2.3

AMS Classification: 68Q17

Key words and phrases: data structure, lower bound, circuit, arbitrary gates, wires, error-correcting codes

*Supported by NSF CCF award 1813930. This work was partly carried out while the author was visiting the Simons Institute for the Theory of Computing in association with the DIMACS/Simons Collaboration on Lower Bounds in Computational Complexity, which is conducted with support from the National Science Foundation.

DOI: 10.4086/toc.2019.v015a018

1 Introduction

Proving data-structure lower bounds is a fundamental research agenda to which much work has been devoted, see for example [21] and the 29 references there. A *static data structure* for a function $f:\{0,1\}^n \to \{0,1\}^m$ is specified by an arbitrary map mapping an input $x \in \{0,1\}^n$ into s memory bits, and m query algorithms running in time t. Here the i-th query algorithm answers query i which is the i-th output bit of f. Time is measured only by the number of queries that are made to the data structure, disregarding the processing time of such queries. The state of time t lower bounds for a given space s can be summarized with the following expression:

$$t \geq \log(m/n)/\log(s/n). \tag{1.1}$$

Specifically, no explicit function for which a bound better than (1.1) is known, for any setting of parameters. This is true even if time is measured in terms of *bit* probes (that is, the word size is 1), and the probes are non-adaptive. The latter means that the locations of the bit probes of the i-th query algorithm only depend on i.

Note that in such a data structure a query is simply answered by reading t bits from the s memory bits at fixed locations that depend only on the query but not on the data. All the data structures in this note will be of this simple form, making our results stronger.

On the other hand, for several settings of parameters we can prove lower bounds that either match or are close to (1.1) for explicit functions. Specifically, for *succinct* data structures using space $s = n + r$ when $r = o(n)$, the expression (1.1) becomes $(n/r)\log(m/n)$. Gál and Miltersen [12], Theorem 5, have proved lower bounds of the form $\Omega(n/r)$.

When $s = n(1+\Omega(1))$, (1.1) is at best logarithmic. Such logarithmic lower bounds were obtained for $m = n^{1+\Omega(1)}$ by Siegel [23], Theorem 3.1, for computing hash functions. For several settings of parameters [23] also shows that the bound is tight. The lower bound was rediscovered in [15]. Their bounds are stated for non-binary, adaptive queries. For a streamlined exposition of this lower bound and matching upper bound, in the case of non-adaptive, binary queries see Lecture 18 in [26]. Remarkably, if $s = n(1+\Omega(1))$ and $m = O(s)$, or if $s = n^{1+\Omega(1)}$ no lower bound is known. Counting arguments (Theorem 8 in [17]) show the existence of functions requiring polynomial time even for space $s = m-1$.

Miltersen, Nisan, Safra, and Wigderson [18] showed that data-structure lower bounds imply lower bounds for branching programs which read each variable few times. However stronger branching-program lower bounds were later proved by Ajtai [1] (see also [3]).

In this note we show that when m is close to s, say $m = O(s)$ or $m = s \cdot \mathrm{poly}\log s$ even a slight improvement over the state of data structure lower bounds implies new circuit lower bounds. When $m = s^{1+\varepsilon}$ for a small enough ε our connection still applies, but one would need to prove polynomial lower bounds to obtain new circuit lower bounds. When say $m = s^2$ we do not obtain anything. It is an interesting open question to link that setting to circuit lower bounds.

2 Circuits with arbitrary gates

A *circuit* $C:\{0,1\}^n \to \{0,1\}^m$ *with arbitrary gates* is a circuit made of gates with unbounded fan-in, computing arbitrary functions. The complexity measures of interest here are the number w of wires, and the depth d. These circuits have been extensively studied, see for example Chapter 13 in the book [14], titled "Circuits with arbitrary gates." The best available lower bounds are proved in [22, 8, 7]. In the case of depth 2 they are polynomial, but for higher depths they are barely super-linear.

Definition 2.1. The function $\lambda_d : \mathbb{N} \to \mathbb{N}$ is defined as $\lambda_1(n) := \lfloor \sqrt{n} \rfloor$, $\lambda_2(n) := \lceil \log_2 n \rceil$ and for $d \geq 3$ $\lambda_d(n) := \lambda^\star_{d-2}(n)$, where $f^\star(n)$ is the least number of times we need to iterate the function f on input n to reach a value ≤ 1. Note $\lambda_3(n) = O(\log\log n)$ and $\lambda_4(n) = O(\log^\star n)$.

With this notation in hand we can express the best available lower bounds from [22, 8, 7]. They show explicit functions $f : \{0,1\}^n \to \{0,1\}^m$ for which any depth-d circuit with w wires satisfies

$$w \geq \Omega_d(m \cdot \lambda_{d-1}(n)). \tag{2.1}$$

Those papers only consider the setting $m = n$, but we note that no lower bound better than (2.1) is available for $m > n$, because such a lower bound would immediately imply a bound better than $\Omega_d(n \cdot \lambda_{d-1}(n))$ for some subset of n output bits. (Write $f : \{0,1\}^n \to \{0,1\}^m$ as $f = (f_1, f_2, \ldots, f_{m/n})$ where each $f_i : \{0,1\}^n \to \{0,1\}^n$. If every f_i is computable in depth d with w wires then f can be computed in depth d with $(m/n)w$ wires. To make the reduction explicit we can append the index i to the input.)

Problem 2.2. Exhibit an explicit function $f : \{0,1\}^n \to \{0,1\}^m$ that cannot be computed by circuits of depth d with $O(m\lambda_{d-1}(n))$ wires, with unbounded fan-in arbitrary gates, for some d.

We show how to simulate circuits with data structures. While it is natural to view a data structure as a depth-2 circuit, a key feature of our result is that it handles depth-d circuits for any d.

Theorem 2.3. Suppose the function $f : \{0,1\}^n \to \{0,1\}^m$ has a circuit of depth d with w wires, consisting of unbounded fan-in, arbitrary gates. Then f has a data structure with space $s = n + r$ and time $(w/r)^d$, for any r.

Proof. Let R be a set of r gates with largest fan-in in the circuit. Note R may include some of the output gates, but does not include any of the input. Note that any other gate has fan-in $\leq w/r$, for else there are r gates with fan-in $> w/r$, for a total of $> w$ wires. The data structure simply stores in memory the input and the values of the gates in R. This takes space $s = n + r$. It remains to show how to answer queries efficiently.

Group the gates of the circuit in $d + 1$ levels where level 0 contains the n input gates and level d contains the outputs. We prove by induction on i that for every $i = 0, 1, \ldots, d$, a gate at level i can be computed by reading $(w/r)^i$ bits of the data structure. For $i = d$ this gives the desired bound.

The base case $i = 0$ holds as every input gate is stored in the data structure and thus can be computed by reading $(w/r)^0 = 1$ bit.

Fix $i > 0$ and a gate g. If $g \in R$ then again it can be computed by reading 1 bit. Otherwise $g \notin R$. Then g has fan-in $\leq w/r$. Then we can compute g if we know the values of its w/r children. By induction each child can be computed by reading $(w/r)^{i-1}$ bits. Hence we can compute g by reading $(w/r)^i$ bits. □

This theorem shows that if for an explicit function $f : \{0,1\}^n \to \{0,1\}^m$ we have a data-structure lower bound showing that for space $s = n + r$ the time t must be

$$t \geq (\omega(m\lambda_{d-1}(n))/r)^d, \tag{2.2}$$

for some d, then we have new circuit lower bounds and Problem 2.2 is solved. We illustrate this via several settings of parameters.

Setting $s = 1.01n$ and $m = 100n$. As mentioned earlier in this setting no data structure lower bound is available: (1.1) gives nothing. We obtain that even proving, say, a $t \geq \log^{\star\star\star}(n)$ lower bound would solve Problem 2.2. Indeed, pick $d = 100$ and $r = 0.01n$. By Inequality (2.2) to solve Problem 2.2 it suffices to prove a lower bound of $t \geq \omega(\lambda_{99}(n))^{100}$, which is implied by $t \geq \log^{\star\star\star}(n)$.

The succinct setting $s = n + r$ with $r = o(n)$, $m = O(n)$. In this setting the best available lower bound is $t \geq \Omega(n/r)$, see [12], Theorem 5. For say $d = 5$ and $r \leq n/\log^\star n$, the right-hand side of (2.2) is within a polynomial of n/r. Hence for any setting of r the lower bound in [12] is within a polynomial of the best possible that one can obtain without solving Problem 2.2. In particular, for redundancy $r = n/\log^c(n)$, we have lower bounds $\Omega(\log^c n)$, and proving $\log^{5c}(n)$ would solve Problem 2.2. We note that moreover the data-structure given by Theorem 2.3 is *systematic*: the input is copied in n of the $n + r$ memory bits. Thus the connection to Problem 2.2 holds even for lower bounds for systematic data structures.

The setting $m = n^{1+\varepsilon}$ *and* $s = n(1 + \Theta(1))$. As mentioned earlier, the best known lower bound is $\Omega(\log m)$. We obtain that proving a data-structure lower bound of the form $t \geq n^{3\varepsilon} \log^4 n$ would solve Problem 2.2. (Pick $r = n$ and $d = 3$.)

The setting $s = n^{1+\Omega(1)}$ *and* $m = s^{1+\varepsilon}$. Here no data-structure lower bounds are known. We get that a lower bound of $t \geq s^{3\varepsilon} \log^4 n$ would solve Problem 2.2.

3 Bounded fan-in circuits

We also get a connection with bounded fan-in circuits (over the usual basis And, Or, Not). Recall that it is not known if every explicit function $f : \{0,1\}^n \to \{0,1\}^m$ has circuits of size $O(m)$ and depth $O(\log m)$. Using Valiant's well-known connection [24] (see [25, Chapter 3] for an exposition) we obtain the following.

Theorem 3.1. Let $f : \{0,1\}^n \to \{0,1\}^m$ be a function computable by bounded fan-in circuits with $O(m)$ wires and depth $O(\log m)$. Suppose $m = O(n)$. Then f has a data structure with space $n + o(n)$ and time $n^{o(1)}$.

Proof. Suppose the circuit has $w = cm$ wires and depth $d = c \log m$. It is known [24] (see Lemma 28 in [25]) that we can halve the depth by removing $cm/\log d$ wires. Repeating this process say $\log\log\log n$ times the depth becomes $O(\log n)/\log\log n = o(\log n)$, and we have removed $o(m)$ wires. Let R be the set of wires we removed. The data structure consists of the input and the values of R. Thus the redundancy is $|R| = o(n)$.

It remains to see how to answer queries fast. Because the depth is $o(\log n)$, the value at every output gate depends on at most $n^{o(1)}$ wires that were removed, and at most $n^{o(1)}$ input bits. Thus reading the corresponding bits we know the value of the output gate. □

In the other uses of Valiant's result, for depth-3 circuits and matrix rigidity, it is not important that each gate depends on few bits of R. However it is essential for us. For this reason it is not clear if the corresponding depth-reduction for Valiant's series-parallel circuits [6] yields data structures.

For completeness we discuss briefly lower bounds for *dynamic* data structures. These data structures are not populated by an arbitrary map as in the static case above. Instead they start blank and are populated via additional *insertion* (or *update*) algorithms whose time is also taken into account. Here the best known lower bounds are $\Omega(\log^{1.5} n)$ [16]. We note that for an important class of problems, known as *decomposable* problems, it is known since [4] how to turn a static data structure with query time t into a data structure that supports queries in time $O(t \log n)$ as well as insertions in time $O(\log n)$, see Theorem 7.3.2.5 in the book [20]. Hence a strong lower bound for such "half-dynamic" data structures would imply a lower bound for static data structures, and one can apply the above theorems to get a consequence for circuits.

4 Data structures for error-correcting codes

We can use Theorem 2.3 to obtain new data structures for any problem which has efficient circuits. Let $f : \{0,1\}^n \to \{0,1\}^m$ be the encoding map of a binary error-correcting code which is *asymptotically good*, that is $m = O(n)$ and the minimum distance is $\Omega(m)$. Gál and Miltersen show [12], Theorem 5, that any data structure for this problem requires time $\geq \Omega(n/r)$ if the space is $s = n + r$. Combining a slight extension of Theorem 2.3 together with a circuit construction in [11] we obtain data structures with time $O(n/r)\log^3 n$.

We first state the result from [11] that we need.

Theorem 4.1. [11] There exists a map $f : \{0,1\}^n \to \{0,1\}^m$ which is the encoding map of an asymptotically good, binary error-correcting code and which is computable by a circuit with the following properties:

(1) The circuit has depth two and is made of XOR gates,

(2) it has $O(n \log^2 n)$ wires,

(3) the fan-in of the output gates is $O(\log n)$, and

(4) the fan-out of the input gates is $O(\log^2 n)$.

Remark 1. Actually the bounds in [11] are slightly stronger. (They prove that the minimum number of wires is $\Theta(n(\log n/ \log\log n)^2)$.) We focus on the above parameters for simplicity. Properties (1)-(3) are explicit in [11], Section 6. Property (4) is only needed here for the dynamic data structure in Theorem (4.3). It holds because the gates in the middle layer are grouped in $O(\log m)$ *range detectors*. And each range detector is constructed with a unique-neighbor expander where the degree in the input nodes is $O(\log m)$.

Next is the new data structure.

Theorem 4.2. There exists an asymptotically good, binary code whose encoding map $f : \{0,1\}^n \to \{0,1\}^m$ has data structures with space $n + r$ and time $O(n/r)\log^3 n$, for every r.

Proof. First note that if in Theorem 2.3 we start with a circuit where the output gates have fan-in k, then we can have a data structure with time $k(w/r)^{d-1}$. The proof is the same as before, using the bound of k instead of w/r for the output gates. Using Theorem 4.1 the result follows. □

We also obtain a dynamic data structure for the encoding map.

Theorem 4.3. There exists an asymptotically good, binary code whose encoding map $f : \{0,1\}^n \to \{0,1\}^m$ has dynamic data structure with space $O(n \log n)$ supporting updating an input bit in time $O(\log^2 n)$, and computing one bit of the codeword in time $O(\log n)$.

Proof. Suppose f has a depth-2 circuit where the output gates have fan-in k and the input gates have fan-out ℓ. Then this gives a data structure where the memory consists of the middle layer of gates. To compute one bit of the codeword we simply read the k corresponding bits in the data structure, and to update one message bit we simply update the ℓ corresponding bits. Essentially this observation already appears in [5], except they do not parameterize it by the fan-in and fan-out, and so do not get a worst-case data structure. Using Theorem 4.1 the result follows. □

In both data structures, the query algorithms are explicit, but the preprocessing and updates are not (because the corresponding layer in the circuits in [11] is not explicit).

5 A lower bound for large s

As remarked earlier we have no lower bounds when s is much larger than n. Next we prove a lower bound of $t \geq 3$ for any $m = n^{O(1)}$ even for $s = m/2 - 1$. This also appeared in [26], Lecture 18. The lower bound is established for a *small-bias generator* [19]. A function $f : \{0,1\}^n \to \{0,1\}^m$ is an ε-biased generator if the XOR of any subset of the output bits is equal to one with probability p such that $|p - 1/2| \leq \varepsilon$, over a uniform input. There are explicit constructions with $n = O(\log m/\varepsilon)$ [19, 2].

Theorem 5.1. Let $f : \{0,1\}^n \to \{0,1\}^m$ be a $o(1)$-biased generator. Suppose f has a data structure with time $t = 2$. Then $s \geq m/2$.

Proof. Suppose for a contradiction that $s < m/2$. By inspection, any function g on 2 bits is either *affine*, or else is *biased*, that is either $g^{-1}(1)$ or $g^{-1}(0)$ contains at most one input.

Suppose $\geq m/2$ of the queries are answered with affine functions. Then because $s < m/2$ some linear combination of these affine queries is fixed. This is a contradiction.

Otherwise $\geq m/2$ of the queries are answered with biased functions. We claim that there exists one biased query whose (set of two) probes are covered by the probes of other two biased queries. To show this we can keep collecting biased queries whose probes are not covered. We must stop eventually, for $s < m/2$. Hence assume that the probes of f_3 are covered by those of f_1 and f_2.

Because f_1 and f_2 are biased, for each of these two functions there exists an output value that determines the values of both its input bits. If these values occur for both f_1 and f_2 then the value of f_3 is fixed as well. Hence, there exists an output combination of $(f_1, f_2, f_3) \in \{0,1\}^3$ which never occurs. This means that the distribution of (f_1, f_2, f_3) over a uniform input has statistical distance $\Omega(1)$ from uniform. But this contradicts the small-bias property, because by the so-called Vazirani XOR lemma, see for example [13], the distribution of (f_1, f_2, f_3) is $\sqrt{8} \cdot o(1) = o(1)$ close to uniform in statistical distance. $\square$

Problem 5.2. Exhibit an explicit function $f : \{0,1\}^n \to \{0,1\}^m$ that does not have a data structure with space $s = n + m/10$, and time $t = 3$.

6 Comparison with independent work

Comparison with [10]. Independent work by Dvir, Golovnev, and Weinstein [10] connects data-structure lower bounds and *matrix rigidity*. Both [10] and this paper aim to link data-structure lower bounds to other challenges in computational complexity. However the works are incomparable. [10] is concerned with linear data structures, and links them to rigidity or linear circuits. This paper is concerned with arbitrary data structures, and links them to circuits with arbitrary gates. [10] shows that a polylogarithmic data-structure lower bound, even against space $s = 1.001n$ and $m = n^{100}$ would give new rigid matrices. This work gives nothing in this regime. The main results in [10] and this work are proved via different techniques. But each paper also has a result whose proof uses Valiant's depth reduction [24].

Comparison with [9]. Independent work by Corrigan-Gibbs and Kogan [9] shows that better data-structure lower bounds for the *function-inversion* problem (and related problems) yield new circuit lower bounds for depth-2 circuits with advice bits.

Acknowledgment. I am grateful to Omri Weinstein for stimulating discussions during his visit at Northeastern University and at the Simons institute. I also thank the anonymous referees for helpful comments.

References

[1] MIKLÓS AJTAI: A non-linear time lower bound for Boolean branching programs. *Theory of Computing*, 1(8):149–176, 2005. Preliminary version in FOCS'99. [doi:10.4086/toc.2005.v001a008] 2

[2] NOGA ALON, ODED GOLDREICH, JOHAN HÅSTAD, AND RENÉ PERALTA: Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992. Preliminary version in FOCS'90. [doi:10.1002/rsa.3240030308] 6

[3] PAUL BEAME, MICHAEL SAKS, XIAODONG SUN, AND ERIK VEE: Time-space trade-off lower bounds for randomized computation of decision problems. *J. ACM*, 50(2):154–195, 2003. Preliminary version in FOCS'00. [doi:10.1145/636865.636867] 2

[4] JON LOUIS BENTLEY: Decomposable searching problems. *Inf. Process. Lett.*, 8(5):244–251, 1979. [doi:10.1016/0020-0190(79)90117-0] 5

[5] JOSHUA BRODY AND KASPER GREEN LARSEN: Adapt or die: Polynomial lower bounds for non-adaptive dynamic data structures. *Theory of Computing*, 11(19):471–489, 2015. [doi:10.4086/toc.2015.v011a019, arXiv:1208.2846] 6

[6] CHRIS CALABRO: A lower bound on the size of series-parallel graphs dense in long paths. *ECCC,* TR08-110, 2008. 5

[7] DMITRIY YU. CHERUKHIN: Lower bounds for Boolean circuits with finite depth and arbitrary gates. *ECCC,* TR08-032, 2008. 3

[8] DMITRIY YU. CHERUKHIN: Lower bounds for depth-2 and depth-3 Boolean circuits with arbitrary gates. In *Proc. 3nd Internat. Comp. Sci. Symp. in Russia (CSR'08)*, pp. 122–133. Springer, 2008. [doi:10.1007/978-3-540-79709-8_15] 3

[9] HENRY CORRIGAN-GIBBS AND DMITRY KOGAN: The function-inversion problem: Barriers and opportunities. *ECCC,* TR18-182, 2019. 7

[10] ZEEV DVIR, ALEXANDER GOLOVNEV, AND OMRI WEINSTEIN: Static data structure lower bounds imply rigidity, 2019. Preliminary version ECCC, TR18-188. [doi:10.1145/3313276.3316348, arXiv:1811.02725] 7

[11] ANNA GÁL, KRISTOFFER ARNSFELT HANSEN, MICHAL KOUCKÝ, PAVEL PUDLÁK, AND EMANUELE VIOLA: Tight bounds on computing error-correcting codes by bounded-depth circuits with arbitrary gates. *IEEE Trans. Inform. Theory*, 59(10):6611–6627, 2013. Preliminary version in STOC'12. [doi:10.1109/TIT.2013.2270275] 5, 6

[12] ANNA GÁL AND PETER BRO MILTERSEN: The cell probe complexity of succinct data structures. *Theoret. Comput. Sci.*, 379(3):405–417, 2007. Preliminary version in ICALP'13. [doi:10.1016/j.tcs.2007.02.047] 2, 4, 5

[13] ODED GOLDREICH: Three XOR-lemmas - An exposition. In *Studies in Complexity and Cryptography*, pp. 248–272. Springer, 2011. [doi:10.1007/978-3-642-22670-0_22] 7

[14] STASYS JUKNA: *Boolean Function Complexity: Advances and Frontiers*. Springer, 2012. [doi:10.1007/978-3-642-24508-4] 3

[15] KASPER GREEN LARSEN: Higher cell probe lower bounds for evaluating polynomials. In *Proc. 53rd FOCS*, pp. 293–301. IEEE Comp. Soc. Press, 2012. [doi:10.1109/FOCS.2012.21] 2

[16] KASPER GREEN LARSEN, OMRI WEINSTEIN, AND HUACHENG YU: Crossing the logarithmic barrier for dynamic Boolean data structure lower bounds. In *Proc. 50th STOC*, pp. 978–989. ACM Press, 2018. [doi:10.1145/3188745.3188790, arXiv:1703.03575] 5

[17] PETER BRO MILTERSEN: The bit probe complexity measure revisited. In *Proc. 10th Symp. Theoretical Aspects of Comp. Sci. (STACS'93)*, pp. 662–671. Springer, 1993. [doi:10.1007/3-540-56503-5_65] 2

[18] PETER BRO MILTERSEN, NOAM NISAN, SHMUEL SAFRA, AND AVI WIGDERSON: On data structures and asymmetric communication complexity. *J. Comput. System Sci.*, 57(1):37–49, 1998. Preliminary version in STOC'95. [doi:10.1006/jcss.1998.1577] 2

[19] JOSEPH NAOR AND MONI NAOR: Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. Preliminary version in STOC'90. [doi:10.1137/0222053] 6

[20] MARK H. OVERMARS: *The Design of Dynamic Data Structures*. Springer, 1983. [doi:10.1007/BFb0014927] 5

[21] MIHAI PĂTRAŞCU: Unifying the landscape of cell-probe lower bounds. *SIAM J. Comput.*, 40(3):827–847, 2011. Preliminary version in FOCS'08. [doi:10.1137/09075336X, arXiv:1010.3783] 2

[22] PAVEL PUDLÁK: Communication in bounded depth circuits. *Combinatorica*, 14(2):203–216, 1994. [doi:10.1007/BF01215351] 3

[23] ALAN SIEGEL: On universal classes of extremely random constant-time hash functions. *SIAM J. Comput.*, 33(3):505–543, 2004. [doi:10.1137/S0097539701386216] 2

[24] LESLIE G. VALIANT: Graph-theoretic arguments in low-level complexity. In *Proc. 6th Internat. Symp. on Mathematical Foundations of Computer Science (MFCS'77)*, pp. 162–176. Springer, 1977. [doi:10.1007/3-540-08353-7_135] 4, 7

[25] EMANUELE VIOLA: On the power of small-depth computation. *Foundations and Trends in Theoretical Computer Science*, 5(1):1–72, 2009. [doi:10.1561/0400000033] 4

[26] EMANUELE VIOLA: Special topics in complexity theory, 2017. Available on author's webpage. 2, 6

AUTHOR

Emanuele Viola
Associate professor
Khoury College of Computer Sciences
Northeastern University
Boston, MA
viola@ccs.neu.edu
http://www.ccs.neu.edu/~viola

ABOUT THE AUTHOR

EMANUELE VIOLA, after worrying about it for half his life, has decided that he believes $\mathsf{P} = \mathsf{NP}$ and that he'd like to use the nickname "Manu." In the picture that appears on the website you can see him in his natural habitat.